Laboratoire VERIMAG

UMR 5104

Directeur: Nicolas Halbwachs

Campagne d'évaluation 2011–2015

Unité de recherche : dossier unique

PROJET

Contents

1 Self Assessment **3**

1.1 Strengths . . . 3

1.2 Weaknesses . . . 3

1.3 Opportunities . . . 4

1.4 Threats . . . 4

2 General Perspectives **4**

3 Perspectives Synchrone **6**

3.1 Transversal Trends . . . 6

3.1.1 Components . . . 6

3.1.2 Energy . . . 6

3.1.3 Algorithms and models for networks of embedded systems . . . 7

3.1.4 Operating systems for embedded applications . . . 7

3.2 Research Themes . . . 8

3.2.1 Programming Languages . . . 8

3.2.2 Verification . . . 8

3.2.2.1 Foundational work in abstract interpretation . . . 8

3.2.2.2 Model-Driven approaches and abstract interpretation . . . 8

3.2.2.3 The SystemC/TLM Verification Chain . . . 8

3.2.2.4 Models and verification methods for energy-related properties . 8

3.2.3 Implementation . . . 9

3.2.3.1 Energy-aware Implementation . . . 9

3.2.3.2 Multi-thread implementations via Lustre++ . . . 9

3.2.3.3 Abstract Interpretation for Compiler optimization . . . 9

3.2.4 Models, Algorithms, Analysis . . . 10

3.2.4.1 Virtual Prototyping . . . 10

3.2.4.2 Between Computational and Analytical Models for Non-Functional Properties of Embedded Systems . . . 10

3.2.4.3 Component models . . . 10

3.2.4.4 Algorithms for distributed systems - foundations . . . 10

3.2.4.5 Algorithms for distributed systems - applications . . . 11

3.3 Projects, Industrial Relations and Contracts . . . 11

3.4 Teaching . . . 12

4 Perspectives DCS **12**

4.1 BIP . . . 12

4.1.1 Encompassing heterogeneity . . . 13

4.1.2 Achieving Constructivity . . . 13

4.1.3 Correct-by-construction model transformations . . . 14

4.1.4 Applications and Tools . . . 16

4.2 Software verification . . . 16

4.3 Security 19
4.3.1 Computational security proofs 19
4.3.1.1 Approach 19
4.3.2 Components for end-to-end secure distributed systems 20

5 Perspectives TEMPO **23**
5.1 Hybrid Systems 23
5.1.1 Toolset Framework Development 23
5.1.2 New Problems and Algorithms 23
5.1.2.1 Nonlinear Systems 23
5.1.2.2 Abstraction Refinement 24
5.1.2.3 Connection with SMT 24
5.1.3 Applications 25
5.1.3.1 Systems Biology 25
5.1.3.2 Analog Circuits 25
5.1.3.3 Embedded Control Systems 25
5.2 Timed Systems and the Multicore Anxiety 26
5.2.1 High-Level Analysis Tools 26
5.2.2 Mapping and Scheduling 26
5.2.3 Satisfaction and Multi-Criteria Optimization 27
5.2.4 Foundations 27
5.2.5 Experimental Validation 27
5.3 Summary for the TEMPO team 28

6 References **28**

1 Self Assessment

1.1 Strengths

The main strength of Verimag is the quality of its production (in term of quality of public actions, number and interest of projects) and its international visibility. There are no recognized "indices" for measuring these criteria, but if we look at the most recent editions of two of the main international conferences in our research domain, EMSOFT'08[1], and TACAS'09[2], we find in each conference 30% papers that cite at least one publication from Verimag. Moreover, 75% of these citations concern publications from the last 10 years, meaning that this significant impact is not due to old historical papers.

Other strong points of the laboratory concern:

- its strong scientific consistency: Verimag concentrates an important strike force on topics which are all related to a unique broad domain. So, there are fruitful cooperations between teams.
- the good balance and synergy between fundamental research and applications. The industrial cooperations are significant within projects.
- its efficiency in finding projects and funding: for instance, our acceptance rate in ANR projects is around 88%.
- we think that the quality of our supervision of PhD students is good: there were very few failures, during the period, and our doctors are rapidly hired, often in academic positions.

1.2 Weaknesses

Our participation in the Institut Carnot LSI highlighted a weakness in direct industrial partnership, which is the type of cooperation encouraged by Carnot institutes. Most direct contracts we have with industry come with "CIFRE contracts", where a company funds a PhD thesis and its supervision by the laboratory. Otherwise, most of our industrial cooperations take place within projects with public funding. In our opinion, projects funded by the ANR or the "poles de compétitivité" strongly compete with direct partnership: companies prefer to be funded for their academic cooperations.

The number of supervised theses is strongly varying, and sometimes low. The supervision task is not evenly shared. A effort was made to increase the number of HDR, which must be sustained.

In the past, one strength of the laboratory was the development and distribution of big software toolboxes, like IF or LUSTRE; these are still maintained, but no longer in the core of our activity. Verimag still develops many tools and prototypes, but no such big platforms dedicated to transfer and distribution.

[1] "International Conference on Embedded Software", Atlanta, Oct. 2008

[2] "Tools and algorithms for the construction and analysis of systems", York, UK, March 2009

There are still 4 non-publishing persons in the laboratory, but two of them are getting involved in research.

1.3 Opportunities

The setting up of "Grenoble Université de l'Innovation" (GUI), the "plan campus" and the PILSI pole are of course very important for us, as for all laboratories in computer science in Grenoble. Verimag is strongly involved in this process, since Yassine Lakhnech is in charge of PILSI, and the laboratory participated in several proposals of projects for PILSI. The installation of the integration center of PILSI in the CTL building will have strong consequences for us, since it should become soon a valuable tool for industrial cooperation, and its proximity will probably be an advantage. Concerning premises, the laboratory will to gather again in the same building, first within the Equation building — hopefully extended with a part of the building presently occupied by the continuing education service of the UJF. The medium-term plan is the gathering of most computer science laboratories within a new PILSI building, which would solve for long the problem of premises, and favor the scientific cooperation with other laboratories and teams.

1.4 Threats

During the last period, two researchers left Verimag, and moved to foreign countries. These departures, and those which could happen in the future, significantly weaken the laboratory. The foreseeable disinvolvement of Joseph Sifakis during the next period is likely to reduce the visibility of Verimag and specifically its presence in European instances. Due to the departure of Sergio Yovine, the ISE team will be terminated (see below). We will pay attention to preserving the scientific consistency of the laboratory, the topics of which should be extended without going into dispersion.

As other laboratories, we suffer from the dissensions between the universities, and we strongly hope and demand that the progress towards a unique University of Grenoble be continued and hastened.

We are very anxious about the current reform of the CNRS, which assigns us to an institute restricted to software, and separated from control theory, architecture and embedded systems.

2 General Perspectives

Verimag has reached such a size that the essential of the scientific policy is decided within the teams. The detailed scientific perspectives are mainly described in the sections devoted to each team.

However, we will continue to take care that the overall consistency of the addressed scientific domain be preserved. In spite of the increasing size of the laboratory and the extension of the research topics, the communication and the cooperation among the teams must be continued and encouraged. An attempt of "large internal seminars" had mitigated results in the preceding period, but should be experienced again. The role of the laboratory is to create opportunities of research and projects, especially concerning transverse topics.

We list below some foreseeable events and tendencies for the near future.

New research trends. Modelling, evaluating and optimizing energy consumption in embedded systems is likely to become a major topic in the next period. Programming multicore architectures will be attacked on several sides, (1) by using component-based programming and the BIP framework, (2) by developing specific program analyzes dedicated to compilation towards these architectures, and (3) by applying scheduling techniques based on timed system analysis. Proof of cryptographic schemes is a promising topic in security. Our work on early simulation and virtual prototyping will be continued and applied not only to the design of systems on chip and sensor networks, but also to complex critical software, in particular in nuclear control systems (COMON project). Concerning tool development, the BIP toolset is likely to become a major platform of the laboratory, and the analysis tools for hybrid systems should be integrated and applied to various application domains.

PILSI projects. Verimag will strongly participate in projects within the PILSI pole. Teams contributed in several projects proposals, in particular those concerning programming multicore processors, the predictable implementation of embedded systems, and wireless sensor networks. More generally, these projects should be an opportunity for increasing the local cooperation with other teams and laboratories.

Termination ISE. Created in 2006, the ISE team will be closed in 2009, because of the departure of its leader, Sergio Yovine, who went back to Argentina for personal reasons. It corresponds to the successful termination of a set of projects involving the team. It does not mean that the research theme of implementation of embedded software will be abandoned: it is developing in other teams, and new projects are started or considered (programming for multicore, static analysis for compilation, . . .).

Relations with Inria and CEA. There are encouraging perspectives of tighter cooperation with Inria: a common team proposal has just been deposited; Verimag is the only non-Inria partner in the Synchronics "action d'envergure".

We are already cooperating with the CEA in several projects, and the planned venue of CEA teams in the PILSI integration center should favor the strengthening of this cooperation.

Premises. As mentioned before, Verimag should leave the CTL building by the end of 2009, to leave place to the PILSI integration center. This is an opportunity to gather again in the same place. In a first step, this will take place within the Equation Center premises that should be extended in the second part of the building. Afterwards, we should move in the new PILSI building, together with two other laboratories (LIG and LJK).

3 Detailed Perspectives: Synchrone team

The reorganization of the group into 4 main research themes being quite recent, we will keep this internal structure for the following years. However, some new trends are emerging, which are quite orthogonal to this structure, and will give birth to new research activities in two or more of the four research themes. They are: *energy-aware* modeling and implementation; *components* in programming or modeling languages, and in verification methods; algorithms and models for *networks of embedded systems*; *operating systems* for embedded applications; compiler optimization and static analysis methods.

We first describe these new trends. Then we detail the research activities of the four main themes.

3.1 Transversal Trends

3.1.1 Components

We have been studying components for some time, now, but this research direction has become transversal. The beginning work on objects for synchronous languages (in the context of the Synchronics project, see below), converges with the work on the model 42 we have been developing for 4 years now. 42 itself converges with the notion of components we need at the transaction level for the virtual prototyping of systems-on-a-chip. In the HELP project (see below), we will be studying *component-based* high-level models of low-power systems. Moreover, the work on verification methods and applications will try and exploit the natural components of the system descriptions we have to take into account (TLM designs, Lustre programs, etc.).

The notion of a component for embedded systems will thus be quite central in all our activities. Our point of view on *components* is that the FAMAPASAP principle (*Forget As Much As Possible As Soon As Possible*) should always be the priority: a component is a piece of development that can be encapsulated, so that other developers can use it without knowing its internal details. Of course, when a component is encapsulated, the *directions for use* should be made explicit in a precise specification, or *contract*.

3.1.2 Energy

The need for low-power systems is now well admitted, in the domain of embedded systems in general. This is particularly true for sensor networks or consumer electronics (mobile phones and all kinds of portable devices), because of lifetime constraints. But this is also true for other (non autonomous) embedded systems, in a world concerned with sustainable development.

Real-time programming exists, in the sense that there are languages, compiler optimizations, implementation methods and execution environments that are specifically designed to meet timing requirements. By analogy, *real-energy programming* does not exist, yet. We will study specific implementation methods.

The other important point concerns *high-level models*. Embedded systems are more and more complex, and time-to-market demands impose very strong constraints on the development methods. To meet these constraints, the design of an embedded system can start from a virtual prototype (i.e., an executable high level model, which can be available early in the design cycle for early decisions). For instance, in the specific domain of System-on-Chip (SoC) design, transaction-level modeling (TLM) has started to make its way into industrial-strength development methods. While TLM models do respect the global abstract functionality, sub-levels have been defined to take into account *extra-functional* aspects. Simpler models are easier to develop and simulate, but less accurate for analysis (for instance timing may go from untimed/causal to approximate-timed to cycle-accurate). The virtual prototype platform has to be relevant for software validation, i.e., be functionally faithful. But its faithfulness with respect to non-functional features of the final chip is also an important question. Low-Power energy saving is now such a key non-functional aspect. We will study high-level computational models for low-power systems, in the context of the HELP project (see below).

Finally, energy consumption is the main property we would like to take into account when trying to bridge the gap between analytical and computational models for embedded systems.

3.1.3 Algorithms and models for networks of embedded systems

Our work on the modeling and analysis of sensor networks has led to the following conclusions:

- One node of a sensor network is an embedded system, with very scarce CPU, energy and memory resources; however, it is very seldom the case that real-time problems occur.
- The whole network can be seen as an embedded system, because its design has to take the physical environment into account. This environment determines the quality of the radio link, of course. But it also determines the activity of the whole network, because of the values obtained by sensing it.
- The interactions between the quite traditional problems of embedded system design (for the nodes), and the problems due to network design (for the whole system) cannot be solved by a simple juxtaposition of the traditional solutions of each domain. This is often called "*cross-layer*" design by the network community, but this should be understood in a wider sense.

The group has a long experience in models for embedded systems, mixing hardware, software and operating systems elements; on the other hand, S. Devismes (who joined us in September 2008) is an expert in distributed systems and fault-tolerant algorithms. We have started exploiting these complementary competences in the ARESA project, but many interesting things are still to be done. In the ARESA2 project, we will study models able to represent the trade-off between energy and security in networks of embedded systems.

3.1.4 Operating systems for embedded applications

After the very simple implementations of synchronous languages on naked machines, we have been studying the introduction of some operating system primitives in the implementations of synchronous programs, to allow multi-cycle programs, distributed and multi-thread execution platforms, and urgent events. This has been done by "importing" some notions from the operating systems community.

We will now look at complete solutions, in which both the application and the needed operating system elements are described at a high level of abstraction. For instance, we can try to describe device drivers in some formal language like Lustre. Then, the application and the necessary drivers are compiled together to form the executable system. In the operating system community this is related to the notion of an *exo-kernel.* The approach is particularly well adapted to embedded systems, and being able to specify both the application and the drivers in a formally defined formalism would be an interesting contribution.

3.2 Research Themes

We now list more specific activities, organized according to the four research themes.

3.2.1 Programming Languages

The main activities will be devoted to the introduction of *objects* in synchronous languages, and to efficient modular compilation methods. This is closely related to the work on components.

This work will be performed in the context of the Synchronics project, with important collaborations with Marc Pouzet (Orsay).

3.2.2 Verification

3.2.2.1 Foundational work in abstract interpretation We will work on two main points:

- Modular static analysis, for numerical properties and their interactions with discrete properties
- Decision procedures for logical theories, quantifier elimination, use of STA/SMT techniques, use of numerical techniques imported from operation research

3.2.2.2 Model-Driven approaches and abstract interpretation This foundational work will be applied to high-level models of embedded (control) systems in Lustre or Simulink. This means applying verification techniques very early in the design flow, long before the sequential code is produced.

3.2.2.3 The SystemC/TLM Verification Chain In the context of the openTLM project (2006-2010), we will continue working in collaboration with B. Jeannet (INRIA Rhône-Alpes) on the application of abstract interpretation to SystemC/TLM models. The idea is to exploit the inter-thread and inter-procedural approaches proposed by B. Jeannet for the formal models extracted from SystemC/TLM design.

On the other hand, we will redefine the internal formal models used to connect SystemC/TLM models to the input of verification tools, by exploiting the SSA form of the C family compilers. First results on the use of the SSA form in this context are published in [BGM+09].

3.2.2.4 Models and verification methods for energy-related properties In the context of a PhD starting september 2009 (Laurie Lugrin), we will continue the fruitful collaboration with L. Mounier (DCS Team) on the definition of formal models for energy consumption. We use the same idea as a lot of other approaches for the modeling of energy consumption, namely *power-state* modeling. A power-state model is a degenerated version of a hybrid automaton, where the states are labeled by equations of the form $de/dt = k$, where k is a constant. The definition of LPTA (Linear Priced Timed Automata) [RLS06] is quite similar to ours; the main difference is that we consider discrete time, and we plan to use symbolic verification techniques, in which time is not treated in a particular way.

However, we will focus on a problem that does not seem to have received much attention for the moment, at least in the formal methods community: we will try to define a formal framework in which the energy models of the various components of a sensor network can be combined, simplified modularly, and analyzed in a per-component basis, so that global estimations of energy consumption for the whole network can be derived. The priority will be on *modular abstractions for non-functional properties.*

3.2.3 Implementation

3.2.3.1 Energy-aware Implementation The group has started to work on the implementations methods that can be defined for a global managing of energy consumption between the embedded software and the various device drivers in the nodes of a sensor network. Existing work in systems like TinyOS allow for a quite ad-hoc managing of energy consumption, in the driver of each hardware element. When a global policy has to be implemented, the global state of the hardware elements has to be taken into account, but this is done in a per-case basis. In fact, some of the global optimizations problems can be formulated as a controller synthesis problem [RW87]. Moreover, in other application domains like systems-on-a-chip for multimedia applications, people have developed solutions based on the notion of a power manager component. Our approach will be to adopt the idea of power managers, and to try and generate them from the specifications of energy policies, by techniques similar to controller synthesis.

3.2.3.2 Multi-thread implementations via Lustre++ The various works done so far on the implementation of Lustre for multi-thread and/or distributed execution platforms have led to the idea of using an intermediate formal language for the description of implementation constraints. For instance, a Lustre programmer can control the placement of sub-programs on processors or threads, by giving some *annotations* in the high level program itself. The idea of a formally-defined intermediate language is to formalize the annotations. We defined Lustre++ for that purpose. Lustre++ can be translted into pure Lustre for all the validation activities (formal verification, simulations, automatic test generation, etc.), and is the input of the compilation chain.

We will continue working in that direction, generalizing the type of execution platforms that can be addressed.

3.2.3.3 Abstract Interpretation for Compiler optimization A lot of compilers are based on the SSA form (Single-Static-Assignment) used as an intermediate form. Optimizations are performed on the SSA form, and are based on simple static analysis techniques. On the other hand, abstract interpretation for proving properties of programs relies on quite sophisticated static analysis techniques. The idea is to import the sophisticated techniques of formal verification into the optimization phases of modern compilers.

A CIFRE thesis with STMicroelectronics has started in 2009, on applying abstract interpretation to the analysis of programs in the SSA form, at the low levels of a real compiler, to prepare the code generation on dedicated architectures. The properties of interest may concern invariant equations among variables, or alignment of addresses (for applying vectorial statements).

3.2.4 Models, Algorithms, Analysis

3.2.4.1 Virtual Prototyping We will continue working on virtual prototyping methods for various embedded systems (systems-on-a-chip, sensor networks, space or avionics systems, etc.). The COMON project (see below) will offer the opportunity to look more closely at nuclear plant applications. For this class of applications, we will concentrate on automatic testing methods.

3.2.4.2 Between Computational and Analytical Models for Non-Functional Properties of Embedded Systems The global objective is to find the right measure for the performances of an embedded system in terms of: time, energy, temperature. This should be supported by formal models, expressive enough to include state-dependent behaviors, and for which abstraction relations and composition operators are formally defined.

We have worked on Real Time Calculus, which is not easy to extend with the notion of state-dependent behavior. In order to augment the expressiveness with states, RTC has been interfaced with timed automata [LPT09]. We proposed to use several *granularities* in order to reduce state explosion [LAM09]. We also worked on the connection to Lustre, to allow the use of abstract interpretation tools connected to Lustre.

All these approaches are based on the idea that real-time-calculus, and the notion of an *arrival curve* that is used as the main abstraction of event flows, are the "main" structure of the model; some of the components may be described in more details with timed-automata or Lustre programs, but they are equipped with wrappers that convert input arrival curves into state-based formalisms, and output state-based expressions back into arrival curves. Another idea is to change the abstraction used for event flows (connections between components) in such a way that it accepts *states*.

3.2.4.3 Component models We will continue the work on the model 42. Several directions will be investigated, among which:

- The use of 42 together with SystemC-TLM (see first results in [BMF09]), to get the benefits of both worlds: the expressiveness and flexibility of SystemC-TLM for large and heterogeneous models of systems-on-a-chip, and the formal definition of 42 that allows a natural connection with various verification and runtime verification tools.
- The formalization of web services in terms of 42 components and contracts
- The extension and use of 42 for the modeling of time and energy consumption (in the context of the HELP project, ANR 2009-12)

3.2.4.4 Algorithms for distributed systems - foundations In the context of the ANR SHAMAN project, we will work in collaboration with Carole Delporte and Hugues Fauconnier

(LIAFA), and Sebastien Tixeuil (LIP6), on:

- Necessary and sufficient properties for fail-stop fault tolerance in limited memory networks.
- Auto-stabilization mechanisms

We will also work on fail-stop fault tolerance for robots in a plan, in collaboration with Franck Petit (LIP) and Francois Bonnet (IRISA).

3.2.4.5 Algorithms for distributed systems - applications We will try and exploit the idea that a network of embedded devices can be considered as a particular type of execution platform for all the embedded software that runs on the devices. This execution platform has to be modeled precisely before embedded software can be developed and evaluated. For instance, the fact that a sensor network is highly redundant can be exploited by using techniques from fault-tolerant distributed systems in order to design the protocols. In other words, designing a "perfect" MAC protocol might be useless and too costly. It would be enough to design a not-perfect MAC protocol, provided the upper levels take this imperfection into account, as fault-tolerant systems do.

The main application will be sensor networks, in the context of the project ARESA2 (ANR 2009-12), where will will study the trade-off between energy consumption and security, with the approach of distributed fault-tolerant systems.

3.3 Projects, Industrial Relations and Contracts

The following projects have been accepted recently:

- ANR Arpège HELP: the project has just been accepted in June 2009. The partners are Verimag/Synchrone, INRIA-AOSTE, the University of Nice, STMicroelectronics Grenoble, and DOCEA Power. The HELP project focuses on functional and non-functional high-level models for the design of low-power embedded systems. The challenge of the HeLP proposal is to study means to provide a component-based, virtual prototype platform approach which relates and combines various modeling levels, with their inherent abstraction levels and the efficient simulation techniques that are associated with each. A key point is the ability to model the intrinsic coupling between functionality and energy (a power manager takes decisions depending on functional information). We plan to do so by capitalizing on expertise by Docea Power and LEAT on Energy/Power modeling, by ST on varying TLM levels with different timing accuracy, by Verimag and INRIA-Aoste on high-level modeling and semantic issues (underlying efficient simulation), and by LEAT on scheduling. The practical relevance of the approach, and its validation against previous, lower-level results shall be asserted through a common case study provided by industrial partners.
- ANR VERSO ARESA2: in this project we will maintain collaborations with FT R&D and other partners on sensor networks. We we look at the tradeoff between energy and security in such systems.
- ANR ASOPT: ASOPT is a fundamental research project proposal, involving software development for experimental and dissemination purposes. The purpose of this project is to develop new abstract domains and new resolution techniques to improve the quality

of program analysis, especially for embedded control programs, and in the longer run, for numerical simulations programs.

- Minalogic COMON (model-based design for nuclear systems): started in 2009, for a duration of 30 months. The partners are local industries of nuclear power plants — ATOS Origin, Corys TESS, and Rolls-Royce Civil Nuclear — together with Verimag. The goal of the project is to define and implement an engineering process for the design of control systems for nuclear plants. The process is intended to encompass all the steps from the global design of the plant to the construction of the control system; it should integrate, complement, and automate the various methodologies currently in use. In this project, the role of Verimag is to contribute to the formalization of concepts, to propose a methodology for early prototyping and testing of the applications, and to develop a prototype tool supporting this methodology.
- INRIA Synchronics: This project, started Jan 1st 2008, is supported by INRIA. It capitalizes on recent extensions of data-flow synchronous languages (mode automata, Lucid Synchrone, Signal, Lustre, Reactive ML, relaxed forms of synchronous composition or compilation techniques for various platforms). We aim to address the main challenges of embedded system design, starting from a single, semantically well founded programming language.
- CIFRE STMicroelectronics (J. LeGuen, PhD 2009-2012): the PhD will study the application of static analysis methods to the optimization of compilers based on the SSA form (Single-Static-Assignment).

Moreover, we should start defining a project on implementation methods and operating systems.

3.4 Teaching

The master curriculum of Grenoble INP entitled "Embedded Software and Systems" has just started (september 2008). Starting sep 2009, P. Raymond, together with Thao Dang (Tempo Team) will teach embedded system implementation, from control problems to multi-thread implementations. F. Maraninchi will continue teaching verification methods (modeling, model-checking and abstract interpretation). M. Moy will continue teaching transaction-level-modeling.

4 Detailed Perspectives: DCS team

4.1 BIP

We will develop a system design methodology based on BIP characterized by the following: Correctness: Guarantee correctness of the application software and its implementation by using two kinds of scalable methods: 1) methods based on constructivity results allowing to infer global properties of a system from properties of its components; 2) methods based on model transformations which preserve functional properties. Productivity: Allow enhanced productivity, especially for programming parallel applications. This can be achieved by offering programmers domain specific languages allowing in particular natural expression of parallelism, both data or functional parallelism. The semantics of these languages will be defined through translation, in BIP. Performance: Allow analysis and evaluation of efficiency in using resources. For this BIP is equipped with notions of resources such as memory, time and energy. Parsimony: Using BIP does not enforce any particular programming or execution model. Designers can use degrees of freedom in the design process, e.g. parallelism or non-determinism, for choosing amongst possible implementations guided only by requirements.

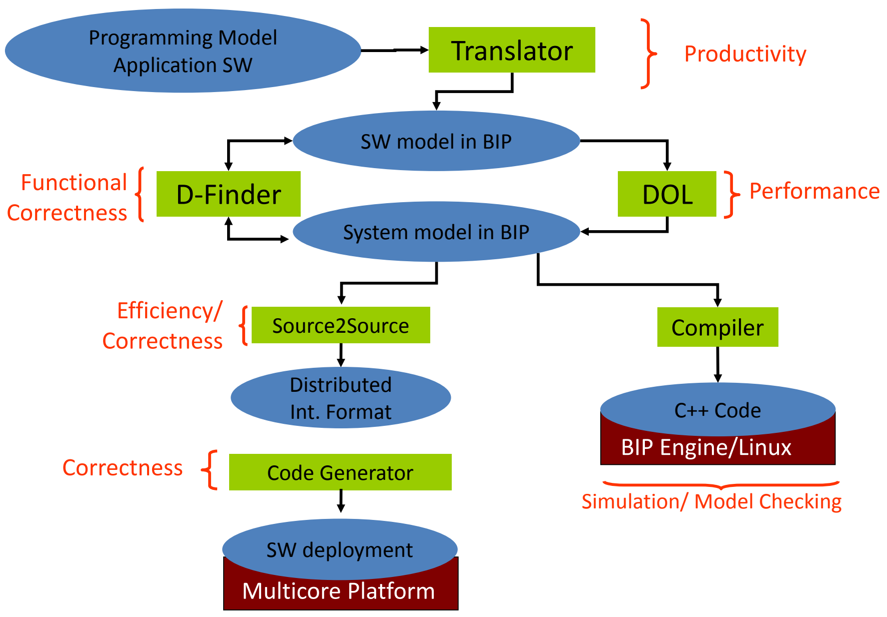

Figure 1: Design Flow Methodology

4.1.1 Encompassing heterogeneity

BIP will be the unifying semantic model for the various programming models used for writing application software. We will study translations into BIP of domain specific languages, including synchronous and data flow languages. Theoretical work deals with further formalization of BIP and of the associated system construction space Behavior $\times$ Interaction $\times$ Priority to study relations between different classes of systems e.g. asynchronous/ synchronous, event-triggered/data-triggered.

4.1.2 Achieving Constructivity

We will extend and adapt existing work [GS05, GGMC$^+$07b] and we will develop new composability and compositionality techniques for classes of properties such as deadlock-freedom, liveness and invariance. These are based on the separation of concerns underlying the layered

BIP model. They will use structural analysis techniques. For deadlock-free atomic components, they will provide sufficient conditions on the interaction models for global deadlock-freedom or for preserving deadlock-freedom of an integrated component. For interaction models, deadlock-freedom preservation is checked by analysis of a dependency graph relating the ports of the components. The dependency relation associates with a port the set of the ports with which synchronization is needed in some interaction. A circuit in the dependency graph characterizes a potential deadlock situation. More detailed analyzes of the behavior atomic components allow deciding deadlock-freedom. For priorities, deadlock-freedom preservation will be checked by composing the priority orders applied in the BIP model. The composition will consist in computing the transitive closure of the union of the priority orders. If the resulting relation is a priority order, then global deadlock-freedom is preserved. We will enhance and extend the existing structural analysis techniques in several directions: We will find sufficient conditions for individual deadlock-freedom of components or clusters of components. These techniques will be applied to other classes of properties such as liveness. Finally, we will use the system construction space Behavior $\times$ Interaction $\times$ Priority to study property preserving transformations. We will study in particular, transformations preserving deadlock-freedom of an untimed system when it is transformed into a timed one by adding timing constraints.

4.1.3 Correct-by-construction model transformations

A key idea in our methodology is to generate from a model of the application software and a model of the target platform, an implementation by using a set of correct-by-construction model transformations. These transformations should preserve functional properties. Furthermore, they should take into account extra-functional requirements. We will study four different types of transformations. Source-to-source BIP transformations: These transformations take BIP models and transform them into functionally equivalent BIP models with different architectures. They have been implemented in the BIP2BIP tool. They allow in particular to generate from a model a single atomic component by composing the behavior of the constituent components. From atomic components monolithic C code can be generated. This code is much more efficient than the componentized code.

From BIP to distributed BIP: We have studied the principles of a distributed implementation method for BIP. The method consists of three steps:

- It starts from a global state model of the system to be implemented described in BIP. The model represents the system behavior as a transition system where transitions are atomic. The BIP execution platform uses an Engine which coordinates the execution of the components. Atomicity of transitions implies a strict alternation between the execution of components and the Engine: no interaction is possible when some component is performing a computation.

- From the global state model, a partial state model is derived where we distinguish between states from which components are ready for interaction and states where components are busy by executing some internal computation. For this model partial state knowledge may suffice for executing interactions. We study conditions for the partial state model to be equivalent to the global state model. The conditions are in the form of an oracle used by the BIP Engine to safely execute interactions in the presence of uncertainty about the global state.

- From the partial state model, a distributed model is obtained where atomic multiparty interactions of the partial state models are replaced by communication protocols. In this model, components exchange messages to communicate with the Engine represented by an additional component.

We have shown that the three models are not in general, observationally equivalent, by considering as silent the actions corresponding to internal computations of the initial global state model. We will investigate conditions under which observational equivalence is achieved. We anticipate the following work directions:

- Study different distributed implementations from fully decentralized to fully centralized ones. The implementation of a multiparty interaction as a protocol can be done: either in a decentralized manner by adding to each one of the components involved in the interaction a controller; or in a centralized manner by using a single controller coordinating the behavior of the components.

- Use existing distributed algorithms for multiparty interaction and conflict resolution e.g. maximal matching algorithm.

- Prove correctness by using composability techniques - non interference of features of the composed algorithms

- Study performance of distributed implementation by considering two criteria: 1) degree of parallelism; 2) overhead for coordination.

- Implementation tools and case studies

Integrating data and memory management policies, This is a very recent work direction. BIP adopts a private memory model which is safe for programming but may lead to inefficient implementations. The aim is to study memory transformation from private to shared memory and conversely. We are also interested in transformations leading to mixed solutions combining private and shared memory and determining tradeoffs.

Integrating architecture constraints: This work is carried out in the framework of the COMBEST IST project where we study a translation from the DOL tool developed at ETHZ to BIP. DOL is based on a network calculus for evaluating performance of streaming data flow applications. We plan to investigate how analytical models can be represented in BIP, by translating them into discretized timed systems. A key issue is to achieve compositionality in this translation, by preserving the structure of the initial description. Based on this work we will investigate techniques for translating architectural constraints from DOL to BIP. This will allow to obtain from an application software model a model of the system to be implemented.

4.1.4 Applications and Tools

We will use theoretical results obtained in the other WPs to enhance performance of the BIP execution Engine. The Engine drives the execution of the C++ code generated from a BIP program. A key performance issue is the computation of the set of the possible interactions of the BIP program from a given state. The Engine has access to the set of the connectors and the priority model of the program. From a given global state, each atomic component of the BIP program, waits for an interaction through a set of active ports (ports labelling enabled transitions) communicated to the Engine. The Engine computes from the connectors of the BIP program and the set of all the active ports, the set of the maximal interactions (involving active ports). It chooses one of them, computes associated data transformations and notifies the components involved in the chosen interaction. Currently, the computation of the maximal set of interactions involves a costly exploration of enumerative representations for connectors. We will study methods for reducing overhead in execution times in particular through symbolic representation and handling of connectors and priorities.

We will continue the work on applications by strengthening the activities on using BIP for programming multicore systems.

4.2 Software verification

Our goals for the future are to produce techniques and tools for the verification of **large, industrial-scale C programs**, that use the following aspects:

1. **Dynamically allocated list and array data structures** and combinations of the above, i.e. arrays of lists, lists of lists, etc. In particular, we consider data structures as containers of values ranging over very large (theoretically infinite) domains, such as e.g., integers, floating point rationals, memory addresses, etc.

2. **Parallelism and synchronization** are commonplace in system code and distributed control applications. We target shared memory multithreading applications written in C using the POSIX `pthreads` library. Synchronization between threads uses shared locks that can be kept inside dynamically allocated data structures (lists, arrays).

The practical goal is the implementation of a prototype tool able to verify legacy code, as the one used in EDF power plants. Examples will be of the order of 10K lines of C code, taken directly from existing industrial applications, or from open-source system code, such as the Linux kernel.

The success measure is given by the number of test cases, of size 10K lines of C code or more, *taken directly from existing applications*, that can be verified by our tools. We will consider test cases that use dynamic data structures, multiple concurrent threads, and combinations of both. Below we detail the novel aspects of the project.

Dynamic lists and arrays of unbounded basic data types Dynamically allocated memory cells are usually used within recursive data structures. The simplest, and the most used in practice, is the *list data structure*, typically used as containers of data values (integers, structures, etc.). Despite the apparently simple definition of a singly-linked list (each memory cell in the list has at most one successor), the memory configurations that may result at run-time may involve sharing and circularity. However, the particular shapes of the memory configurations generated by a program using only list data structures (*single-successor heaps*) can be represented by a finite number of *symbolic shape graphs*, in which each linear list segment without incoming pointers can be summarized in a symbolic node. Then, the number of elements in the list segment can be memorized by an integer variable (counter).

During the AVERILES project, two tools have been built based on this idea, namely L2CA (developed at VERIMAG) and TOPICS (developed at LSV). The tools have been tested on a number of programs handling lists, and the experimental results were published in [BBH$^+$06b, WWWd, BIP08]. The idea of summarizing list segments (although by losing track of the list lengths) has been independently used by other research groups. For instance, the TVLA shape analysis engine implements a module dedicated to singly-linked lists [MYRS05], while the Smallfoot tool uses Separation Logic with specialized predicates that describe individual list segments in isolation [BCO04, BCO05].

The experience we acquired working with this method points out its limitations:

1. Even if the number of symbolic heaps is finite (theoretically bounded by a double exponential in the number of program variables), in practice there are cases in which the size of the counter automata generated is too large to be handled by existing verification tools for counter automata.

2. The abstraction has finite range provided that the number of entries in the heap (namely program variables) is finite. This is not always the case for programs with recursive function calls, in which the local variables are allocated onto the caller's stack.

3. It is in general difficult to represent the data within lists. Even when considering list nodes with boolean data fields, list segments cannot be summarized without loss of information. In particular, this is problematic when locks are used inside list nodes, as the state of a lock is in fact a boolean field.

In order to limit the explosion of the symbolic representations, and thus achieve scalability, in this project we intend to conceive a new abstract representation of heaps, whose range is bounded by a polynomial in the number of program variables. This can be done at the cost of losing information, and possibly introducing false alarms. The framework needs to be flexible, and allow to eliminate false alarms, by counter-example guided refinement.

A novel, alternative, method of dealing with containers of data, is to consider *array structures*, instead of lists. In general the size of the array is fixed once the structure is allocated, and does not change until it is entirely disposed, unlike the lists which may grow or shrink, as needed. On the other hand, arrays are accessed in constant time, compared to list access which is linear in the size of the structure. We have recently investigated the problem of verifying properties of programs working with integer arrays. To this respect, we have defined specification logics for reasoning about integer arrays. The decidability of the satisfiability problems for these logics rely on theoretical results concerning counter automata [HIV08b, HIV08a]. These preliminary results open the possibility of extending the method developed in the AVERILES project for programs with lists without data, to deal with lists as containers of data.

Finally, we intend to tackle programs that use *composite data structures*, such as arrays of pointers to lists, lists of lists, or lists in which certain elements belong to other lists. These structures are used in industrial applications, in order to speed up the access to data, as it is the case of hash tables. To handle such structures efficiently, we intend to develop adequate symbolic representations. Possibilities include use of tree automata (suggested for instance, by previous work on Regular Model Checking at LIAFA), and extending suitably the models used for representing singly linked lists.

Modularity as key to scalability Large industrial-scale applications are usually structured in small functions, each of which performs a rather simple task. The key to having a scalable program verification method is to exploit the decomposition of the program into small pieces of code, analyze each such piece under given pre-conditions, and reuse the analysis results, each time the function is invoked. Since the functions are usually rather small, the summary information can be kept in succinct structures, and plugged in on demand. Preliminary results in this direction were obtained during the AVERILES project [BFQ07], however these ideas have not yet been tried on large scale examples.

On a different line of work, we can view a piece of code as composed of (possibly nested) loops, interceded with sequences of instructions. Usually, in real-life applications, loops are fairly small with respect to their enclosing context. Previous experience with symbolic representations for heap structures, partially arising from projects such as AVERILES and VERDYN, allow us to assert that certain formalisms (namely logics) are more suitable to deal with sequential code (one can develop lighter post-condition calculi), whereas others are suitable for inference of loop invariants (namely automata). A novelty of this project is to explore the possibility of combining different symbolic representations into dealing with different parts of the code. This involves developing theoretical connections between different formalisms (such as, the classical connections between logic and automata [Büc62, Rab72]).

Parallelism and synchronization The inefficient way to tackle the verification of a concurrent code with a fixed number of synchronization points and a fixed number of thread is to simply build the product of the control flow graph of each thread. Not only inefficient, this approach also relies on some hypothesis that do not hold in many interesting cases. They are several approaches to tackle the verification of concurrent codes more efficiently, ranging from assume-guarantee (Owicki and Gries) for concurrent racy programs, to Hoare monitors for well-compartmented concurrency with clearly identified shared resources and critical sections. All of these approaches do not address concurrent programs with dynamic memory allocation, for several reasons, an obvious one being that the possible aliasing between data complicates the analyzes of the interferences or the definition of the shared resources.

Recently, several approaches based on separation logic have been proposed for extending the approaches we mentioned to the context of concurrent programs with dynamic memory allocation. Hoare resource invariants is naturally handled in concurrent separation logic (CSL), whereas assume guarantee, orthogonal to the disjoint concurrency rule of separation logic, has been integrated into an hybrid proof system (RGSep) that combines both separation logic and rely guarantee, the benefit of such a system being that one may focus the interference analysis on the part of the heap where programs do effectively perform some concurrent accesses. The automation of the proof inference is at a very early stage for RGSep, and completely unknown for CSL. A quite hard problem for automation is to automatically construct the resource invariants used in CSL.

4.3 Security

4.3.1 Computational security proofs

Cryptography plays a central role in the design of secure and reliable systems. Nevertheless, designing secure cryptographic schemes is notoriously hard. *Provable security* [GM84] aims to provide a mathematical foundation for reasoning about the correctness of cryptographic schemes. In the provable security setting, security statements are formulated in complexity-theoretical terms: typically, they express that feasible adversaries have a negligible advantage of achieving some goal, for example distinguishing between two ciphertexts, or forging a signature. Moreover, statements are proved formally by reduction to supposedly hard problems. While provable security allows rigorous security definitions and proofs, there is a lack of rigorously justified proof systems that capture standard reasoning patterns for reasoning about cryptographic schemes. Existing attempts to formalise provable security, such as the game-based approach of Bellare and Rogaway [BR06a] and Shoup [Sho04], often limit themselves to provide a (more or less) rigorous modeling language for describing the interaction between adversaries and schemes. However, the general principles used to carry proofs remain largely informal. Indeed, formal methods for security, including symbolic methods, typing systems, theorem proving, etc..., have been by large restricted to the symbolic verification of security protocols and systems making the hypothesis that cryptographic primitives are ideal. While this abstraction can be justified in many cases, symbolic methods seem to have some limitations. It does not seem to be obvious how cover one-way functions, for instance, in the symbolic approach.

What we aim at is a formal proof system that captures cryptographic principles at a level of abstraction on par with informal proofs.

4.3.1.1 Approach Partially with Gilles Barth (IMDEA, Madrid), Bruce Kapron (University of Victoria, Victoria, Canada) and Christine Paulin-Mohring (LRI, Orsay)

Computational Indistinguishability Logic (CIL) is a general logic for proving the security of cryptographic schemes, without committing to a particular model or setting. It allows reasoning about security of many cryptographic schemes, including encryption and signatures, against adaptive adversaries *directly* in computational models, including the standard model and idealized models such as the random oracle model. Cryptographic schemes are modeled in using computational frames, which extend frames of applied π-calculus [AF01] with random sampling, adversary calls, and oracles. While frames in [AF01] only account for one interaction between the adversary and the primitive, computational frames allow multiple interactions, and can model indistinguishability games against adaptive adversaries, and many other properties which escape the scope of π-calculus frames.

The exact security of cryptographic schemes is expressed in CIL using indistinguishability statements of the form $s \sim_\epsilon t$, where s and t are frames and ϵ is an upper bound for the probability of an adversary being able to distinguish between them, or negligibility statements of the form $s :_\epsilon E$, where s is a frame, E is an event over the output of s, and ϵ is an upper bound for the probability of E to hold on the output of s. Indistinguishability statements can be thought as a generalization of observational equivalence to probabilistic equivalence, whereas negligibility statements can be thought as a generalization of postconditions to probabilistic postconditions.

CIL uses a small set of deduction rules to capture common reasoning patterns, and interface rules to connect with external reasoning. CIL deduction rules handle simulations and reductions steps, whereas external reasoning is used for observational equivalence of frames, logical equivalence of events, or postconditions. The combination of deduction rules and external reasoning supports concise and informative proofs, and greatly simplifies soundness proofs. More speculatively, CIL should provide a solid basis for a systematic investigation of the theory of cryptographic proofs. In turn, these investigations would help building formally justified and succinct cryptographic proofs.

The benefits of CIL are many. First, CIL is widely applicable. Indeed, computational frames provide a language to describe the interactions between an adversary and a player, but do not commit to a language for their computations. As a result, one can instantiate CIL (and use its logic) to the different settings used to carry formal cryptographic proofs: mathematics [Now07a], processes [Bla06], λ-calculus [BBU08], imperative programs [BGZ09]. Second, CIL has a clean foundation. Indeed, the functional semantics of frames elicits many of the idiosyncrasies found in more syntactic frameworks for games. As a result, the soundness of the logic is intuitive. Third, CIL is extensible.

We started a Coq-based formalization of the first version of CIL. Our goal is to build a proof checker on top of Coq. The tool should support the development of cryptographic proofs in CIL. Each inference step is Coq-checked in a transparent way to the user. An other aspect, we started investigating is the question of proof search.

4.3.2 Components for end-to-end secure distributed systems

Partially with Takoua Abdellatif Université de Sousse, Tunisia)

Building distributed systems that satisfy end-to-end security requirements is a difficult task. Most existing techniques concentrate on access control policies and security protocols, that are essential for ensuring data confidentiality and integrity but do not provide end-to-end security guarantees. Access control policies do not track information flow through the entire system and do not cope with implicit information flow. Similarly, cryptographic components are used for secure communication and authentication but do not guarantee any global security properties. Information flow policies [DD77, GM82, BL75, Bib77] are natural for specifying end-to-end confidentiality and integrity requirements because they put global constraints on the flow of information. For example, an access control policy can require that only users with the appropriate read rights can read file f; while an information-flow policy would require that only users with the required security level can get any information about the content of file f even indirectly. Security-typed languages [VIS96, HR98, Aba07, ZZNM02] provide a promising framework for describing and implementing such policies. In this framework, types express restrictions on the flow of information. Typing annotations can be used at compile time to check that the program respects the information-flow constraints or at run-time to enforce such constraints. To date, most security typed languages have addressed systems implemented on a single trusted host. In distributed systems, multiple hosts cooperate in order to implement a function. This entails that data and computation are distributed among several, often distant hosts. Moreover, the dominant work on building distributed secure systems is devoted to access control policies. A notable exception is the approach proposed by the JIF group in [ZZNM02]. In this work, a program partitioning algorithm is presented. It uses security types in order to split data and computations on heterogeneously trusted hosts while respecting the original security requirements. The distributed system generated by JIF/Split is secure under the assumption that the communication infrastructure is secure, that is, sent messages are confidential and arrive at destination unaltered and in-order. In this paper, we take a complementary view to JIF/split and develop the *Component Information Flow (CIF for short)* framework. Our starting point is a specification of the architecture of the system that describes the components of the system and their interaction. Each component includes a set of input and output ports, respectively. An output port of a component C can be linked to an input port of an other component; this defines a link between these components. Each component comes with a code describing its behavior and an interface describing how it can be connected to other components. CIF components are also equipped with a *security interface.* I.e., inter-component links are tagged with security labels that express information-flow policies. More specifically, each port p is tagged with a security label $(C(p), I(p))$, where $C(p)$ is the confidentiality label and $I(p)$ its integrity. In case p is an output port of a component C, the security label $(C(p), I(p))$ expresses the requirement that data sent via port p must be protected such that only components with confidentiality level higher than $C(p)$ are allowed to read this data. Moreover, the data sent can only be used by components who require less integrity than $I(p)$. A security-typed architecture is well-typed, if the components interconnection is consistent with the security-types: consider a component C_1 with output port p connected to an input port q of component C_2 . Then, the conditions $C(p) \sqsubseteq C(q)$ and $I(p) \sqsupseteq I(q)$ must hold. The condition $C(p) \sqsubseteq C(q)$ means that C_1 treats data received on port q as at least as confidential as $C(p)$, thus in particular, C_1 does not send this data on a port with less confidentiality. The condition $I(p) \sqsupseteq I(q)$ means that C_1 may assume that the received data has at least the integrity level $I(q)$. The consistency of the inter-components security types does not guarantee that there is no undesirable information-flow, since it entails restrictions on the behavior of the components that must be verified. This is achieved by the intra-component type system that verifies that each component satisfies its security interface, i.e., the security labels attached to its ports. The problem that remains to be solved now is to generate code that implements the functional behavior of the components but also implements the security interfaces. To this end, we develop a system architecture transformer (SAT) that takes as input an architecture with security tags and generates a target system architecture where the security tags are implemented using cryptographic primitives. In our current implementation, we use strongly secure (IND-CCA) asymmetric encryption and (EF-CMA) digital signature to implement confidentiality and integrity requirements, respectively. We demonstrate the feasibility of our approach o two case studies: the battleship game and a standard Web service application.

5 Detailed Perspectives: TEMPO team

5.1 Hybrid Systems

The effort in the hybrid systems domain will consist of three major inter-related axes: tool development and integration, new algorithms and deeper excursions into application domains.

5.1.1 Toolset Framework Development

Continuing the development of the framework for analysis of continuous and hybrid systems will be a central activity of our group in the next 4 years, financed mostly by the MULTIFORM project and its potential successors, as well as other projects (the ongoing ATHOLE project and the new ANR project VEDECY). The goals of this framework, whose chief architect is Goran Frehse, are both inside and outside directed. Internally, the components of the framework will allow us to perform rapid explorations of new algorithms, data structure or analysis techniques, without having to reinvent everything from scratch. It will facilitate the development of new prototype tools by new students, and will increase the probability that their work will be preserved after they leave the lab. Externally, the framework will allow us to put in the public domain reliable versions of our tools, with a common user interface and visualization tools, and thus disseminate our result among potential user communities and obtain valuable feedback. The availability of the framework will also help the scientific community at large by reducing the investment in making a thesis in the domain. We hope to recruit a permanent research engineer to make the development process much more efficient and smooth.

The first packages that are planned to go public are the *linear hybrid automata scenario* (packaged as a new version of Phaver) and the *support function scenario* which will make available our recent advances in reachability computation for linear and piecewise-linear differential equation of high-dimension. Another high priority activity of is to integrate simulations and systematic exploration of *parameter spaces* into the framework. We also intend to exchange ideas and, perhaps, software, with the group of INRIA which develops a library of abstract domains.

The major efforts planned for this period are:

5.1.2 New Problems and Algorithms

On the other side of the R&D pipeline we will continue to widen the scope of our analysis techniques, and investigate new theoretical and computational problems. These activities will be partially supported by the new VEDECY project with LJK and INRIA, coordinated locally by Thao Dang.

5.1.2.1 Nonlinear Systems We intend to improve the reachability techniques for nonlinear systems in general and polynomial systems in particular, as they play a central role in models of biochemical networks. One one hand we will continue the focus on special classes of nonlinear systems, in particular polynomial systems and exploit their properties to develop efficient representation and computation schemes. On the other hand, we intend to work on the *dynamic hybridization* scheme to produce a mature technology that can be later transferred into a tool. The first step will be to combine it with the novel linear reachability algorithms that employ implicit representations using support functions and zonotopes. This will allow us to go to higher dimensions than what is possible with the current vertex-based representation. Then we will optimize the various ingredients of the scheme such as the techniques for choosing linearization domains, determination of their sizes and orientation. We also need to extend the technique to treat external inputs. Hopefully, at the end of the process, for which we intend to recruit a PhD student, we will be able to treat nonlinear systems with 15-30 state variables.

The laws of *mass action* govern models dealing with chemical reactions, population dynamics and potentially additional socio-economical situations. The control of such systems by changing interaction parameters and thus leading the dynamics to alternative steady states is an intellectually-challenging problem with potential applications to drug design. Consequently there is recently a growing interest in the domain, combining idea from chemical reaction theory, dynamical systems and systems biology. We intend to investigate this problem using a simplified abstract model of *population-preserving mass action* systems and to study techniques for isolating equilibria for such system starting from the special case of *multilinear* systems.

5.1.2.2 Abstraction Refinement The application of formal methods to hybrid systems clearly hinges on the complexity of computing with continuous sets with the desired precision. Whenever the desired properties can be shown with a model (or algorithm) of less accuracy, this may qualitatively reduce the required computation effort. The ordering on models and their complexity is known: nonlinear dynamics, affine and linear dynamics, followed by piecewise constant and timed dynamics. The cost of basic image operators ranges from doubly exponential and worse down to cubic. The goal of abstraction refinement is to focus the computation effort on behaviors where precision is indispensable, and to apply coarser abstractions wherever it is not. We plan develop an approach based on hybridization and mixing analyzes on abstractions of different precision and type, generalizing CEGAR-type methods to the hybrid domain. After hybridization of the continuous state space, i.e., dividing it up into smaller partitions, the dynamics can be over-approximated in each partition. In a sequence of refinements steps, the precision of the over-approximation may be reduced based on the result (forward/backward refinement, cut sets, etc.). The initial approximation may be very coarse, such as piecewise constant dynamics; subsequent steps may increase a precision parameter (size of integer coefficients, time step in integration routines, number of faces in polyhedral approximations) up to a certain point. They may also decide to switch to a different abstraction, e.g., from timed to piecewise constant, from piecewise constant to affine, etc. Finally, the refinement may decide to use a different approach altogether, such as switching from a set-based computation to a trajectory-based method (verification by simulation of trajectories).

5.1.2.3 Connection with SMT Having acquired access and expertise in the powerful technology of constraint solving using SMT, we will investigate its application in complementary methods for analyzing continuous and hybrid dynamical systems. In this approach questions such as the existence of bounded length trajectory of a certain property or the existence of a system invariant of a specific type can be formulated as a satisfiability problem in some theory. For some systems the solver that we have developed for linear constraints can be used while for the others we will need to extend the solver to handle more complex theories such as the theory of polynomial constraints. We intend to interact more intensively with other members of Verimag who investigate similar problems in the context of program verification.

5.1.3 Applications

5.1.3.1 Systems Biology Together with increasing the capabilities of our tools, we intend to look closer at real-life biochemical models. We will continue the collaboration with TIMC on using our techniques for the analysis of a model of *ongiogenesis*, an important process in development, healing and cancer. This process which underlies the sprouting of new blood vessels (which serve, among others, the nutritional needs of tumors) starts with a process of *collagen proteolysis* which is the response of the cell to external signalling by initiating the secretion of enzymes that loosen the extracellular matrix, and provides for cell migration. We will study a biochemical model of this phenomenon and try to gain insight on the influence of various parameters on its dynamics starting with our simulation-based approach for exploring the parameter space. This model, which admits as many as 18 state variables and 23 parameters, brings an excellent opportunity to stretch the muscles of our tools and also to develop the methodological aspects related to the use of such tools in the biological research cycle. We have initiated a collaboration with TIMC and LJK that will hopefully intensify systems biology activities in Grenoble.

5.1.3.2 Analog Circuits Our second favorite application domain, electronic circuits, is characterized by an astronomic number of state variables. Not only does a circuit admit millions of transistors, but unlike the past where models of a single transistor could range in 5-80 state variables, depending on the physical fidelity, today new technologies may require around 20000 (!) state variables to faithfully model the physics of *one* transistor. For these and other reasons we cannot expect our verification techniques to be applied routinely in the analysis of analog circuits (or of digital circuits at the electrical level). The most we can expect from these exhaustive techniques is to analyze high-level models of isolated components. Consequently we will focus on less ambitious simulation-based techniques. We plan to continue the investigation of novel coverage notions not only for safety and reachability properties but also for a more general class of properties, especially those of interest in the domain of circuit design. Test generation algorithms to achieve such good coverage will then be investigated. This investigation will benefit from exploration of real-life circuit applications using the newly-developed interface of our test generation tools that can handle the industrial standard SPICE circuit description. Through collaboration with Mentor Graphics and ST Microelectronics we plan to adapt our monitoring techniques to the industrial context and study, in particular, the role of mixed-signal assertions and test-generation specifications in the integration of the digital and analog design flows. The flow of information between low-level physical models and higher-level architectural models seems to be a major bottleneck in the design of high-performance low-power systems and we hope to contribute some insights in this area.

5.1.3.3 Embedded Control Systems The recent advances in nonlinear reachability will allow us to verify the properties of closed-loop behaviors of nonlinear control systems subject to disturbances.

5.2 Timed Systems and the Multicore Anxiety

The decision of the semiconductor industry to achieve performance via parallelism is a source of very difficult research problems that will occupy computer science for the coming years. This *multicore* challenge, to which we have been already exposed within the ATHOLE project, will be the major driving force for our activities in the domain. Our group will try to contribute new ideas, theoretical results and tools to this worldwide effort. We intend to work along the following lines, profiting from the proximity of some major players in the domain such as ST Microelectronics and CEA-LETI.

5.2.1 High-Level Analysis Tools

It is of primary importance to be able to evaluate the performance of an application on an architecture at early stages of the design process of both the application and the architecture, using any working combination of simulation, analytical and formal methods. We intend to continue our work on building a high-level performance evaluation and design-space exploration tool based on models in the granularity of tasks and processors, modeled at the level of abstraction of *timed automata*. This modeling framework will allow us to model applications, architectures, scheduling policies and external environment at a high-level of abstraction appropriate for rapid performance estimation. Being aware of the scalability issues in timed automata analysis, and observing that their underlying worst-case reasoning is inappropriate for performance analysis of soft real-time systems, we will develop complementary probabilistic approaches for analyzing timed automata by discrete-event simulation where delays and durations are drawn randomly from the temporal uncertainty intervals.

5.2.2 Mapping and Scheduling

The efficient deployment of applications on multiprocessors is a key for high performance and low power consumption. We intend to continue our work in the domain with emphasis on communication and data sharing costs, studying both static (compile time) and dynamic (run time) aspects. The main challenges is to find a level of modeling which should satisfy several non-trivial requirements. First, it should be sufficiently abstract to provide for rapid analysis and optimization that can be integrated in the software development and deployment cycle. In the same spirit, it should be sufficiently general in order not to tailor by hand a new solution for every application instance. Finally it should be sufficiently faithful to real applications and architectures and rely on information that can be easily extracted from the code of the application annotated with lightweight performance data. We will continue our work based on task-data graphs and will have to extend our scheduling algorithms, developed for a distributed memory architecture to deal with shared memory. We will use SMT solvers as the main computational tool for solving these hard optimization problems.

5.2.3 Satisfaction and Multi-Criteria Optimization

The use of SMT solvers opens new possibilities for solving optimization problems while providing approximation guarantees based on unsatisfiability results. We will develop a systematic search methodology using the SMT solver in the inner loop of the optimization process. Of particular interest is the problem of *multi-criteria optimization* which is crucial in choosing different configurations of an embedded solution that may differ in price, performance, power consumption and other criteria. In this setting there is no unique optimal solution and one would like to provide designers with a good sampling of the possible trade-offs between these criteria, formalized by the notion of Pareto solutions. We are developing novel methods for approximating the Pareto surface using an SMT solver. The problems here are both conceptual (what is the appropriate definition of an ϵ-approximation of the Pareto surface) and algorithmic (how to converge to such an approximation in a high-dimensional space using a search algorithm which submits queries to the solver). Having access to the internals of the solver, we can optimize it for the purpose of optimization by sharing learned clauses among subsequent queries.

5.2.4 Foundations

The focus on multicores calls for a rethinking of commonly-used computability and complexity concepts, putting computation and communication on equal footing as special instances of a more general notion of taking data residing in one location, transforming it and moving the result to another location. Characterizing different classes of applications in terms of their inherent computation and communication requirements, various ratios between them and the distribution of data volumes over the computation tree can be helpful for choosing the appropriate architecture for executing these applications. The utility of such investigations concerning the computation/communication structure is not restricted to multicore but can be applied in to parallel computing in other scales such as clusters and web computing.

5.2.5 Experimental Validation

Our participation in the ATHOLE project brings an opportunity to be engaged in activities related to the new *platform 2012* developed by ST with participation of other local actors such as CEA and INRIA. In particular we hope to influence the process of defining the programming model of this new architecture while taking software deployment and performance aspects into account at early stage of the design. This platform consists of a network whose nodes are “islands” of 8 memory-sharing processors and is intended to support high-performance video applications. Participating in the P2012 efforts can allow us in the future to apply our performance analysis, mapping and scheduling solutions to concrete applications and architectures and have real impact on the domain. Other multicore platforms that already exist worldwide can serve for experimentation purpose until this platform is fully developed.

5.3 Summary for the TEMPO team

We have come to a critical point where our modeling and analysis know-how in timed and hybrid systems has the potential of being applied to two important and high-impact domains, computational systems biology and deployment of software on multicores. It is clear that a small team of 3 permanent researchers cannot meet this challenge without a significant augmentation of the work force. We have identified two urgent needs:

- A *research engineer* to ensure *continuity* of our tool development effort, to participate in *experimentation* with multicore platforms and to maintain a leading edge in the promising SMT technology;
- A researcher/lecturer with a general computer science/applied mathematics culture and interest in biology to serve as an interface between us and scientists coming from experimental disciplines such as Physics and Biology in TIMC and elsewhere to help in the dissemination of our analysis techniques to this domain and in their adaption to its needs.

6 References

[Aba07] Martín Abadi. Access control in a core calculus of dependency. *Electr. Notes Theor. Comput. Sci.*, 172:5–31, 2007.

[AF01] Martín Abadi and Cédric Fournet. Mobile values, new names, and secure communication. In *28th Annual Symposium on Principles of Programming Languages (POPL) (London, UK)*, pages 104–115. ACM, January 2001.

[BBH$^+$06b] Ahmed Bouajjani, Marius Bozga, Peter Habermehl, Radu Iosif, Pierre Moro, and Tomás Vojnar. Programs with lists are counter automata. In *CAV*, volume 4144 of *Lecture Notes in Computer Science*, pages 517–531, 2006.

[BBU08] Michael Backes, Mathias Berg, and Dominique Unruh. A formal language for cryptographic pseudocode. In *Proceedings of LPAR'08*, pages 353–376, 2008.

[BCO04] Josh Berdine, Cristiano Calcagno, and Peter W. O'Hearn. A decidable fragment of separation logic. In *FSTTCS*, pages 97–109, 2004.

[BCO05] Josh Berdine, Cristiano Calcagno, and Peter W. O'Hearn. Smallfoot: Modular automatic assertion checking with separation logic. In *FMCO*, pages 115–137, 2005.

[BFQ07] Ahmed Bouajjani, Severine Fratani, and Shaz Qadeer. Context-bounded analysis of multithreaded programs with dynamic linked structures. In *Proceedings of the International Conference on Computer Aided Verification (CAV'06)*, volume 4590 of *Lecture Notes in Computer Science*, Berlin, Germany, July 2007. Springer.

[BGM+09] Loïc Besnard, Thierry Gautier, Matthieu Moy, Jean-Pierre Talpin, Kenneth Johnson, and Florence Maraninchi. Automatic translation of c/c++ parallel code into synchronous formalism using an ssa intermediate form. In *Ninth International Workshop on Automated Verification of Critical Systems (AVOCS'09)*, Swansea, Wales, GB, September 2009.

[BGZ09] Gilles Barthe, Benjamin Grégoire, and Santiago Zanella Béguelin. Formal certification of code-based cryptographic proofs. In *Proceedings of POPL'09*, pages 90–101, 2009.

[Bib77] K. J. Biba. Integrity considerations for secure computer systems. Technical report, USAF Electronic Systems Division, Bedford, MA, 1977.

[BIP08] Marius Bozga, Radu Iosif, and Swann Perarnau. Quantitative separation logic and programs with lists. In *Proc. 4th International Joint Conference on Automated Reasoning (IJCAR 2008)*, 2008.

[BL75] D. E. Bell and L. J. LaPadula. Secure computer system: Unified exposition and multics interpretation. Technical Report ESD-TR-75-306, MITRE Corp. MTR-2997, Bedford, MA, 1975. Available at NTIS AD-A023 588.

[Bla06] Bruno Blanchet. A computationally sound mechanized prover for security protocols. In *2006 IEEE Symposium on Security and Privacy (S&P 2006), 21-24 May 2006, Berkeley, California, USA*, pages 140–154. IEEE Computer Society, 2006.

[BMF09] T. Bouhadiba, F. Maraninchi, and G. Funchal. Formal and executable contracts for transaction-level modeling in systemc. In *ACM International Conference on Embedded Sofware (EMSOFT 09)*, Grenoble, France, October 2009.

[BR06a] Mihir Bellare and Phillip Rogaway. The security of triple encryption and a framework for code-based game-playing proofs. In *Proceedings of EUROCRYPT'06*, pages 409–426, 2006.

[Büc62] J.R. Büchi. On a decision method in restricted second order arithmetic. In *Logic, Methodology and Philosophy of Science*, pages 1–11, 1962.

[DD77] Dorothy E. Denning and Peter J. Denning. Certification of programs for secure information flow. *Commun. ACM*, 20(7):504–513, 1977.

[GGMC+07b] Gregor Gößler, Susanne Graf, Mila E. Majster-Cederbaum, Moritz Martens, and Joseph Sifakis. Ensuring properties of interaction systems. In *Program Analysis and Compilation, Theory and Practice*, volume 4444 of *Lecture Notes in Computer Science*, pages 201–224, 2007.

[GM82] Joseph A. Goguen and José Meseguer. Security policies and security models. In *IEEE Symposium on Security and Privacy*, pages 11–20, 1982.

[GM84] S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, April 1984.

[GS05] G. Gössler and J. Sifakis. Composition for component-based modeling. *Sci. Comput. Program.*, 55(1-3), 2005.

[HIV08a] P. Habermehl, R. Iosif, and T. Vojnar. A Logic of Singly Indexed Arrays. In *Proc. of LPAR'08*, volume 5330 of *LNAI*. Springer, 2008.

[HIV08b] Peter Habermehl, Radu Iosif, and Tomas Vojnar. What else is decidable about integer arrays? In *Proc. of Eleventh International Conference on Foundations of Software Science and Computation Structures*, volume 4962 of *Lecture Notes in Computer Science*, pages 474–489, 2008.

[HR98] Nevin Heintze and Jon G. Riecke. The slam calculus: Programming with secrecy and integrity. In *POPL*, pages 365–377, 1998.

[LAM09] Yanhong Liu, Karine Altisen, and Matthieu Moy. Granularity-based interfacing between rtc and timed automata performance models. Technical report, Verimag, Centre Équation, 38610 Gières, August 2009.

[LPT09] Kai Lampka, Simon Perathoner, and Lothar Thiele. Analytic real-time analysis and timed automata: A hybrid method for analyzing embedded real-time systems. In *8th ACM & IEEE International conference on Embedded software, EMSOFT 2009*. ACM, October 2009.

[MYRS05] Roman Manevich, Eran Yahav, G. Ramalingam, and Mooly Sagiv. Predicate abstraction and canonical abstraction for singly-linked lists. In Radhia Cousot, editor, *Proceedings of the 6th International Conference on Verification, Model Checking and Abstract Interpretation, VMCAI 2005*, volume 3148 of *Lecture Notes in Computer Science*, pages 181–198. Springer, January 2005.

[Now07a] David Nowak. A framework for game-based security proofs. In *Proceedings of ICS'07*, volume 4861, pages 319–333, 2007.

[Rab72] M.O Rabin. Decidability of second-order theories and automata on infinite trees. *Journal of Symbolic Logic*, 37:618–619, 1972.

[RLS06] J. I. Rasmussen, K. G. Larsen, and K. Subramani. On using priced timed automata to achieve optimal scheduling. *Form. Methods Syst. Des.*, 29(1):97–114, 2006.

[RW87] P. J. G. Ramadge and W. M. Wonham. Supervisory control of a class of discrete event processes. *SIAM Journal of Control Optimization*, 25(1):206–230, 1987.

[Sho04] Victor Shoup. Sequences of games: a tool for taming complexity in security proofs, 2004. http://eprint.iacr.org/2004/332.

[VIS96] Dennis M. Volpano, Cynthia E. Irvine, and Geoffrey Smith. A sound type system for secure flow analysis. *Journal of Computer Security*, 4(2/3):167–188, 1996.

[WWWd] http://www.lsv.ens-cachan.fr/ sangnier/topics/index.php.

[ZZNM02] Steve Zdancewic, Lantian Zheng, Nathaniel Nystrom, and Andrew C. Myers. Secure program partitioning. *ACM Trans. Comput. Syst.*, 20(3):283–328, 2002.