INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Project-Team s4

System Synthesis and Supervision, Scenarios

Rennes

THEME COM

Activity Report

2005

Table of contents

1. Team

S4 *is a joint project of* INRIA, CNRS *and the University of Rennes 1, within* IRISA *(UMR 6074).*

Head of project-team
Benoît Caillaud [CR, INRIA]

Administrative assistant
Myriam David [TR, INRIA, part-time in S4]

Research scientists
Éric Badouel [CR, INRIA]
Albert Benveniste [DR, INRIA, part-time in S4]
Philippe Darondeau [DR, INRIA]

Faculty member (University of Rennes 1)
Sophie Pinchinat [Lecturer]

Ph. D. students
Rodrigue Djeumen [Since September 2005, funded by SCAC Yaoundé (Service de Coopération et d'Action Culturelle de l'Ambassade de France), part time in France]
Guillaume Feuillade [INRIA, until August 2005, teaching assistant at University of Rennes 1, from September 2005, PhD defended in December 2005]
Bernard Fotsing [Since March 2005, funded by AUF (Agence Universitaire Francophone), part time in France]
Jean-Baptiste Raclet [INRIA, funded by the collaboration CO2 with France Telecom]
Maurice Tchoupé [Since September 2005, funded by SCAC Yaoundé (Service de Coopération et d'Action Culturelle de l'Ambassade de France), supplemented by SARIMA, part time in France]

2. Overall Objectives

2.1. Overall Objectives

The objective of the project is the realization by algorithmic methods of reactive and distributed systems from partial and heterogeneous specifications. Methods, algorithms and tools are developed to synthesize reactive software from one or several incomplete descriptions of the system's expected behavior, regarding functionality (synchronization, conflicts, communication), control (safety, reachability, liveness), deployment architecture (mapping, partitioning, segregation), or even quantitative performances (response time, communication cost, throughput).

These techniques are better understood on fundamental models, such as automata, Petri nets, event structures and their timed extensions. The results obtained on these basic models are then adapted to those realistic but complex models commonly used to design embedded and telecommunication systems.

The behavioral views of the *Unified Modeling Language* [31] (sequence diagrams and statecharts), the *High-Level Message Sequence Charts* [30] and the synchronous reactive language Signal are the heart of the software prototypes being developed and the core of the technology transfer strategy of the project.

The scientific objectives of the project can be characterized by the following elements:

A focus on a precise type of applications: The design of real-time embedded software to be deployed over dedicated distributed architectures. Engineers in this field face two important challenges. The first one is related to system specification. Behavioral descriptions should be adaptable and composable. Specifications are expressed as requirements on the system to be designed. These requirements fall into four categories: (i) functional (synchronization, conflict, communication), (ii) control (safety, reachability, liveness), (iii) architectural (mapping, segregation) and (iv) quantitative (response time, communication cost, throughput, etc). The second challenge is the deployment of the design on a distributed architecture. Domain specific software platforms, known as *middleware* or *real-time operating systems* or *communication layers*, are now part of the usual software design process in industry. They provide a specialized and platform-independent distributed environment to higher-level software components. Deployment of software components and services should be done in a safe and efficient manner.

A specific methodology: The development of methods and tools which assist engineers since the very first design steps of reactive distributive software. The main difficulty being the adequacy of the proposed methods with standard design methods based on components and model engineering, which most often rely on heterogeneous formalisms and require correct by construction component assembly.

Scientific and technological foundations: Those models and methods which encompass (i) the distributed nature of the systems being considered, (ii) true concurrency, and (iii) real-time.

Team S4 contributes methods, algorithms and tools producing distributed reactive software from partial heterogeneous specifications of the system to be synthesized (functionality, control, architecture, quantitative performances). This means that several heterogeneous specifications (for instance, sequence diagrams and state machines) can be combined, analyzed (are the specifications inconsistent?) and mapped to lower level specifications (for instance, communicating automata, or Petri nets).

The scientific method of Team S4 begins with a rigorous modeling of problems and the development of sound theoretical foundations. This not only allows to prove the correctness (functionality and control) of the proposed transformations or analysis; but can also guarantee the optimality of the quantitative performances of the systems produced with our methods (communication cost, response time).

Synthesis and verification methods are best studied within fundamental models, such as automata, Petri nets, event structures, synchronous transition systems. Then, results can be adapted to more realistic but complex formalisms, such as the UML. The research work of Team S4 is divided in five tracks:

Petri net theory: This track follows up the main research theme of the former Team PARAGRAPHE at INRIA Rennes. In addition to further developments on topics related to the theory of regions, an algebraic theory of nets has been developed.

Scenario languages: Current research work concentrates on the composition of system views expressed in scenario formalisms such as *High-Level Message Sequence Charts* (HMSC) [30]. This research is done in cooperation with France Telecom (section 6.1).

Heterogeneous systems: This track contributes to the extension, to distributed systems, of the well-established synchronous paradigm. The aim is to provide a unified framework in which both synchronous systems, and particular asynchronous systems (so-called weakly-synchronous systems) can be expressed, combined, analyzed and transformed.

Reactive components: The design of reusable components calls for rich specification formalisms, with which the interactions of a component with its environment combines expectations of the component about its environment, with guarantees offered in return by the component to its environment.We are investigating questions related to reactive component refinement and composition. We are also investigating a component-based language development environment, where attribute grammars are used to define executable specifications of software components.

Discrete event system synthesis and supervisory control: Many synthesis and supervisory control problems can be expressed, with full generality, in the *quantified mu-calculus*, including the existence of optimal solutions to such problems. Algorithms computing winning strategies in parity games (associated with formulas in this logic) provide effective methods for solving such control problems. This framework offers means of classifying control problems, according to their decidability or undecidability, but also according to their algorithmic complexity.

3. Scientific Foundations

3.1. Scientific Foundations

The research work of the team is built on top of solid foundations, mainly, algebraic, combinatorial or logical theories of transition systems. These theories cover several sorts of systems which have been studied during the last thirty years: sequential, concurrent, synchronous or asynchronous. They aim at modeling the behavior of finite or infinite systems (usually by abstracting computations on data), with a particular focus on the control flow which rules state changes in these systems. Systems can be autonomous or reactive, that is, embedded in an environment with which the system interacts, both receiving an input flow, and emitting an output flow of events and data. System specifications can be explicit (for instance, when the system is specified by an automaton, defined extensively by a set of states and a set of transitions), or implicit (symbolic transition rules, usually parameterized by state or control variables; partially-synchronized products of finite transition systems; Petri nets; systems of equations constraining the transitions of synchronous reactive systems, according to their input flows; etc.). Specifications can be non-ambiguous, meaning that they fully define at most one system (this holds in the previous cases), or they can be ambiguous, in which case more than one system is conforming to the specification (for instance, when the system is described by logical formulas in the modal mu-calculus, or when the system is described by a set of scenario diagrams, such as *Sequence Diagrams* [31] or *Message Sequence Charts* [30]).

Systems can be described in two ways: either the state structure is described, or only the behavior is described. Both descriptions are often possible (this is the case for formal languages, automata, products of automata or, Petri nets), and moving from one representation to the other is achieved by folding/unfolding operations.

Another taxonomy criteria is the concurrency these models can encompass. Automata usually describe sequential systems. Concurrency in synchronous systems is usually not considered. In contrast, Petri nets or partially-synchronized products of automata are concurrent. When these models are transformed, concurrency can be either preserved, reflected or even, infused. An interesting case is whenever the target architecture requires distributing events among several processes. There, communication efficient implementations require that concurrency is preserved as far as possible and that, at the same time, causality relations are also preserved. These notions of causality and independence are best studied in models such as concurrent automata, Petri nets or Mazurkiewicz trace languages.

Here are our sources of inspiration regarding formal mathematical tools:

1. Jan van Leeuwen (ed.), *Handbook of Theoretical Computer Science - Volume B: Formal Models and Semantics*, Elsevier, 1990.
2. Jörg desel, Wolfgang Reisig and Grzegorz Rozenberg (eds.), *Lectures on Concurrency and Petri nets*, Lecture Notes in Computer Science, Vol. 3098, Springer, 2004.
3. Volker Diekert and Grzegorz Rozenberg (eds.), *The Book of Traces*, World Scientific, 1995.
4. André Arnold and Damian Niwinski, *Rudiments of Mu-Calculus*, North-Holland, 2001.
5. Gérard Berry, *Synchronous languages for hardware and software reactive systems Hardware Description Languages and their Applications*, Chapman and Hall, 1997.

Our research exploits decidability or undecidability results on these models (for instance, inclusion of regular languages, bisimilarity on automata, reachability on Petri nets, validity of a formula in the mu-calculus, etc.) and also, representation theorems which provide effective translations from one model to another. For instance, Zielonka's theorem yields an algorithm which maps regular trace languages to partially-synchronized products of finite automata. Another example is the theory of regions, which provides methods for mapping finite or infinite automata, languages, or even *High-Level Message Sequence Charts* [30] to Petri nets. A further example concerns the mu-calculus, in which, algorithms computing winning strategies for parity games can be used to synthesize supervisory control of discrete event systems.

Our research aims to contribute effective representation theorems, with a particular emphasis on algorithms and tools which, given an instance of one model, synthesize an instance of another model. In particular we have contributed a theory, several algorithms and a tool for synthesizing Petri nets from finite or infinite automata, regular languages, or languages of *High-Level Message Sequence Charts*. This also applies to our work on supervisory control of discrete event systems. In this framework, the problem is to compute a system (the controller) such that its partially-synchronized product with a given system (the plant) satisfies a given behavioral property (control objective, such as, a regular language or satisfaction of a mu-calculus formula).

Software engineers often face problems like *service adaptation* or *component interfacing*. Problems of this kind are reducible to particular instances of system synthesis or supervisory control problems.

4. Application Domains

4.1. Application Domains

Results obtained in Team S4 apply to the design of real-time systems consisting in a distributed hardware architecture and software to be deployed over that architecture. A particular emphasis is put on *embedded* systems (automotive, avionics, production systems, ...), and also, to a lesser extent, *telecommunication* and *production* systems.

Research on scenario languages, and in particular on compositions of *High-Level Message Sequence Charts* is well suited to the specification and analysis of *services* in *intelligent* telecommunication networks. This work is funded by France Telecom (section 6.1).

Our work on heterogeneous reactive systems facilitates the mapping of pure synchronous designs to a distributed architecture where communication is done by non-instantaneous message passing. These architectures can be usual *asynchronous* distributed systems or, more interestingly, *loosely time-triggered architectures* (LTTA), such as those found on-board recent Airbus planes. In the latter, communication is done by periodically reading or writing (according to local inaccurate real-time clocks) distributed shared variables, without any means of synchronizing these operations. The consequence is that values may be lost or duplicated, and software designed for such specific architectures must resist losses or duplications of messages. In the context of the IST European network of excellence ARTIST (Section 7.1) we have developed a theoretical and methodological framework in which the correct mapping of synchronous designs to such particular distributed architectures can be best understood, at a high level of abstraction.

Our work on Petri net synthesis and distributed control (Section 5.1) have found applications in various domains such as automated production systems (in particular, flexible production cells, in collaboration with Team MACSI of INRIA Lorraine) and work-flow engineering.

5. New Results

5.1. Petri nets

Keywords: *MV-algebra*, *Petri algebra*, *Petri net*, *partial 2-structure*, *persistent Petri net*.

Participants: Éric Badouel, Philippe Darondeau.

We have studied the problem of embedding partial 2-structures into set 2-structures such that the target substructure is full and forward closed and it is minimal w.r.t. these properties. A. Ehrenfeucht and G. Rozenberg introduced the notion of partial 2-structures — an abstract form of transition systems — and studied the problem of representing them as partial set 2-structures — directed graphs made of sets and their ordered symmetric differences. They constructed a representation and gave the conditions under which the representation is an isomorphism. We have proposed in [11] an alternative representation of partial 2-structures by partial set 2-structures which are complete graphs, hence their transitions may be left implicit, yielding a static representations of dynamic systems.

We have shown in [12] that 2-bounded and persistent Petri nets with the same language have indeed isomorphic reachable state graphs. The proof is entirely based on the residual operation which was defined by E. W. Stark in [36].

The firing rule of Petri nets relies on a residuation operation for the commutative monoid of natural numbers. We have identified in [17] a class of residuated commutative monoids, called Petri algebras, for which one can mimic the token game of Petri nets to define the behavior of generalized Petri nets whose flow relation and place contents are valued in such algebraic structures. The sum and its associated residuation capture respectively how resources within places are produced and consumed through the firing of a transition. We have shown that Petri algebras coincide with the positive cones of lattice-ordered commutative groups and constitute the sub-variety of the (duals of) residuated lattices generated by the commutative monoid of natural numbers. We however have exhibited a Petri algebra whose corresponding class of nets is strictly more expressive than the class of Petri nets. More precisely, we have introduced a class of nets, termed lexicographic Petri nets, that are associated with the positive cones of the lexicographic powers of the additive group of real numbers. This class of nets has been proved universal in the sense that any net associated with some Petri algebra can be simulated by a lexicographic Petri net. All the classical decidable properties of Petri nets however (termination, covering, boundedness, structural boundedness, accessibility, deadlock, liveness...) are undecidable on the class of lexicographic Petri nets. Finally we have turned our attention to bounded nets associated with Petri algebras, and have shown that their dynamic can be reformulated in term of MV-algebras.

5.2. Heterogeneous reactive systems

Keywords: *Kahn's networks*, *distributed architectures*, *globally asynchronous*, *locally syncrhonous systems*, *loosely synchronous architectures*, *synchronous systems*, *time-triggered architectures*.

Participants: Albert Benveniste, Benoît Caillaud.

In the framework of the ARTIST network of excellence (see Section 7.1), we have developed a systematic method to formally model heterogeneous reactive systems. This is joint work with Alberto Sangiovanni-Vincentelli, Luca Carloni (U.C. Berkeley and Columbia University) and Paul Caspi (VERIMAG).

The motivation is twofold. On the one hand, heterogeneous models are encountered throughout the design flow for embedded systems: use of UML notations, of Simulink-Stateflow, of synchronous languages. On the other hand, execution architectures for deployment generally follow a *model of computation* that is different from that of the modeling tools. For example, whereas the Time-Triggered Architecture (TTA) of H. Kopetz [35] strictly obeys the synchronous model, this is no longer the case for other commonly used infrastructures (field buses, CAN, ARINC, etc.). In 2002–2003, we analyzed the Loosely Time-Triggered Architecture (LTTA), that is in use at Airbus.

In 2003 and 2004, we addressed the issue of heterogeneity, thanks to the so-called *tag system* model originally due to Edward Lee and Alberto Sangiovanni-Vincentelli. We have simplified and tailored this model to our needs [32]. The new version covers not only synchronous and asynchronous models, timed and untimed models, and their free combination, but also is able to model causality dependence relations, such as those induced by data- and control-flow dependencies. We have formally defined what it means to migrate from one model to another. We have formally defined what heterogeneous parallel composition means, *e.g.*, what $P \| Q$ means, for P synchronous and Q asynchronous. We have formally defined what it means to preserve semantics, *e.g.*, when migrating from a synchronous to a *globally asynchronous, locally synchronous* design (GALS). We have characterized, by algebraic means, those designs that preserve semantics when deployed on an infrastructure which *model of computation* differs.

In 2005, we presented an operational view of tag systems, where we focus on tag machines as mathematical artifacts that act as finitary generators of tag systems. Properties of tag machines have been investigated. Results on homogeneous compositionality are detailed in [18].

In another direction, Benoît Caillaud and Dumitru Potop-Butucaru (VERIMAG, then INRIA, team AOSTE) have developed a theory for the correct deployment of synchronous designs over globally asynchronous, locally synchronous (GALS) architectures. We have introduced the notion of weak endochrony, at a macro-step level, which extends to a synchronous setting the classical theory of Mazurkiewicz traces [15].

In [21], we introduce a micro-step model for the representation of asynchronous implementations of synchronous specifications. The model covers classical implementations, where a notion of global synchronization is preserved by means of signaling, and globally asynchronous, locally synchronous (GALS) implementations where the global clock is removed. Our model offers a more refined framework for reasoning about essential correctness properties of an implementation: the preservation of semantics and the absence of deadlocks.

5.3. Reactive components

Keywords: *aspect weaving*, *attribute grammar*, *behavioral type*, *interface*, *views*.

Participants: Éric Badouel, Benoît Caillaud, Rodrigue Djeumen, Bernard Fotsing, Maurice Tchoupé, Jean-Baptiste Raclet.

Component-based design is acknowledged as an important approach to improving the productivity in the design of complex software systems as it allows pre-designed components to be reused in larger systems. Designing components for reuse however calls for a system of program annotations rich enough to ensure that the components will interact in a coherent manner when connected together. That dynamic information about the interactions of the component with its environment combines expectations of the component about its environment with guarantees offered in return by the component to its environment.

When it comes to reactive systems a natural extension of the usual functional type systems is to consider finite automata with input and output channels as alphabets. These automata can take care of the more involved protocols of communication that take place among components in interaction. L. De Alfaro and Th. Henzinger introduced for that purpose *Interface Automata*, viewed as enriched type systems (the so-called *Behavioral Type Systems*), which capture the temporal aspects of software component interaction. A component refines another component if it imposes less constraint about the environment and offers more guarantee in return. We obtain in this way a compositional semantics due to the fact that a component can be replaced with a more refined version in any environment compatible with the original component: The refined version may offer more services but both are equivalent in restriction to the set of services of the original component; this situation is reminiscent to the sub-class polymorphism in object-oriented programming.

We are investigating some variants of this problem. The first variant is when an open asynchronous reactive system is viewed as a transducer that reacts to its environment by producing some output in response to some input received from its environment and according to its current internal state. Both input and output are buffered leading to communication monoids. A second variant considers synchronous systems whose interface are described by modal transition systems. In each case we intend to provide an operation for component assembly as a product of the components controlled by their respective intended environments, and symmetrically an operation realizing the conjunction of the environment specifications restricted by each component (since each component realizes partly the environment of the other component).

We are also investigating a component-based language development environment based on the formalism of attribute grammars. An abstract syntax tree decorated with attributes (AST+A in short) that conforms to an attribute grammar (AG in short) can be seen as an executable specification of a software component. In this context many problems can be easily stated, if not solved. For instance, aspects can be described almost syntactically at the level of attribute grammars and the aspect weaving is obtained during code generation. A view in an attribute grammar can be given as a subset of "visible" non-terminal symbols, we can then derive a new attribute grammar associated with this partial view of systems. Some problems naturally arise as the problem of coherency of views (decide whether a family of AST+A each conforming with a different view corresponds to the various projections of a same AST+A conforming to the original attribute grammar) and the problem of integration of views (decide whether given a family of attribute grammars, one can find an attribute grammar, and if so a most general one, so that each attribute grammar in the family is the projection of this global attribute grammar associated with some view). We also intend to study in that context invasive compositions of components by composing attribute grammars and to introduce a related notion of behavioral type when attribute grammars are used to describe reactive systems (inputs from the environment are transmitted through inherited attributes and the component responds to its environment by returning synthesized attributes as output).

5.4. Discrete event system synthesis and supervisory control

Keywords: *control*, *discrete event system*, *modal logics*, *model synthesis*, *mu-calculus*, *parity game*, *partial observation*, *regular languages*, *tree automata*, *winning strategy*.

Participants: Sophie Pinchinat, Philippe Darondeau, Guillaume Feuillade, Jean-Baptiste Raclet.

Synthesis is a challenging problem: in general, *program synthesis* concerns investigations on automated methods to derive a program from a specification. When the synthesis is done according to semantically correct transformation rules, the resulting programs, if any, is a correct implementation of the specification. Therefore, synthesis techniques are generally preferred to *verification-based approaches*, as the latter can be qualified as "post-mortem examinations", where incorrect programs have to be modified. Synthesis problems encompass classic theoretical issues such as satisfy

The general trend is to restrict synthesis problems to simplified situations, where only some aspects of a program are considered rather than the whole program itself. We have investigated synthesis methods both on finite transition systems, unlabeled Petri nets, which naturally capture the concurrency of distributed systems, and modal logic specifications. Non-functional properties such as timing constraints are considered.

It is worthwhile noticing that synthesis problems encompass classic theoretical issues such as satisfiability of logical assertions or control of discrete event systems.

During year 2005, the activity in this track of the S4 team can be summarized as follows:

- In order to show the benefits of unlabeled Petri net synthesis for distributed supervisory control, we have imposed a structural constraint of distributability on nets. Bounded and distributable nets translate to equivalent systems of finite message passing automata. Therefore, distributable net controllers induce distributed control systems for distributed discrete event systems: each DES component receives as its local control component one of the message passing automata, and the locally controlled subsystems interact with one another in fully asynchronous mode. We have studied in [19] the implementation of Ramadge and Wonham's finite state controllers by distributable net controllers.
- In [13] and [27], we present an extension of Badouel & Darondeau's results for unlabeled Petri net synthesis from regular languages: we study synthesis from families of languages defined through modal mu-calculus formulas, which translate into modal specifications. A structural restriction makes this problem is decidable.
- In his PhD thesis [10], Guillaume Feuillade has investigated a particular satisfiability problem: the mu-calculus logic, with its fragments, are considered to specify properties of unlabeled Petri nets. Considering branching-time specifications (mu-calculus sentences) is in sharp contrast with other research works in the literature, which consider only verification or synthesis issues on regular languages. Guillaume Feuillade's results provide a better understanding of the limit between decidable and undecidable cases of the satisfiability problems mentioned above.
- We have developed a logical formalism for the specification of control problems for discrete-events systems under partial observation. We allow quantifications over atomic propositions of the mu-calculus and consider a decidable fragment suitable for control synthesis [14].

- Maximal permissivity is an optimality criterion for controllers that is often the result of dedicated synthesis algorithms: the algorithms are designed for classes of control problems for which maximal permissive controllers exist and are unique. Still, maximally permissive solutions might exist in circumstances which do not fall into such identified classes, but there is no way to ensure that classic algorithms deliver an optimal solution. In [24], we propose a general synthesis procedure which always computes a maximal permissive controller when it exists, for very large classes of control problems.
- In [22], we answer a wide range of control problems for nondeterministic discrete-event systems, relying on recent works based on a second order logic approach for deterministic systems. We investigate a pair of transformations: the first one transforms a nondeterministic system into a deterministic one, thanks to an additional unobservable event; the second transforms logical statements, taking this unobservable event into account. In particular, these transformations are used to reduce control problems for nondeterministic systems to control problems under partial observation for deterministic systems.
- In [23], we clarify the notion of *architecture* in decentralized control, in order to consider the *realizability* problem: given a discrete-event system, a desired behavior and decentralized control architecture, can the desired behavior be achieved by decentralized controllers obeying the given architecture? We investigate this problem for any mu-calculus definable behavior and for a large family of architectures. The method consists in compiling into a single mu-calculus formula both the desired behavior and the desired architecture. We have obtained a general decentralized control algorithm that can encompass a large number of these architectures. Interestingly, this provides an adequate framework to compare different decentralized architectures.

6. Contracts and Grants with Industry

6.1. CO2: Composition of viewpoints based on scenario languages

Participants: Éric Badouel, Benoît Caillaud, Philippe Darondeau.

This collaboration with France Telecom Research and Development, in Lannion and Issy les Moulineaux is a followup of a previous collaboration with them, which has allowed to develop techniques and tools for the analysis and composition of timed *High-Level Message Sequence Charts* (HMSC) [30].

In 2005, we have addressed the problem of assembling partial views of the behavior of a distributed system. For this purpose, we have defined an operation of controlled shuffle on languages of message sequence charts [33]. Controlled shuffle appears to be suitable for a concise description of protocols. The crucial question left open in this work is the decision whether the controlled shuffle of two existentially bounded languages of message sequence charts is still in this class.

7. Other Grants and Activities

7.1. ARTIST I/II – Network of Excellence on Advanced Real-Time Systems

Participants: Albert Benveniste, Benoît Caillaud.

IST-2001-34820 (April 1st, 2002, October 1st, 2005), http://www.artist-embedded.org/ and IST-004527 ARTIST2 (September 2004, September 2008), http://www.artist-embedded.org/FP6/

The ARTIST project was completed this year. Its main contributions have been the ARTIST Roadmap [29] and the report on Embedded Systems Education [37].

The strategic objective of the ARTIST2 Network of Excellence is to strengthen European research in Embedded Systems Design, and promote the emergence of this new multi-disciplinary area.

ARTIST2 has a double core, consisting of leading-edge research in embedded systems design issues (described later in this document) in the Joint Programme of Research Activities (JPRA), and complementary activities around shared platforms and staff mobility in the Joint Programme of Integration Activities (JPIA). The JPRA activities are pure research, and the JPIA are complementary efforts for integration. Both work towards deep integration between the participating research teams. The JPRA and JPIA are structured into clusters - one for each of the selected topics in embedded systems design (in red). Teams may be involved in one or several clusters.

Around this double core is the Joint Programme of Activities for Spreading Excellence (JPASE). These are complementary activities for disseminating excellence across all available channels, targeting industry, students, and other European and international research teams. ARTIST2 is structured into 8 clusters: *Modeling and components*, *adaptive real-time*, *control for embedded systems*, *execution platforms*, *testing and verification*, *compilers and timing analysis*, *and hard real-time*.

The *hard real-time* cluster deals with one of the two main approaches to embedded systems design. It is focused on systems where temporal constraints are critical. This includes applications, which are often safety-critical, such as applications in space, automobile, rail transport, air traffic control, production control, etc. This is the oldest and most mature approach, and has led to significant research results in Europe, such as synchronous languages, fixed-priority scheduling, the time-triggered architectures, and these have been significantly transferred and developed in industry (for example, SCADE and Esterel to Airbus, and TTA to automotive). Albert Benveniste leads the *hard real-time* cluster.

7.2. Catalysis: categorical and algebraic approaches to synthesis

Participants: Éric Badouel, Philippe Darondeau.

CATALYSIS, which stands for *categorical and algebraic approaches to synthesis* is a collaboration initiated in 1999 between Team S4 and the Institute for Computer Science (IPI PAN) in Gdansk, Poland. This collaboration is part of the scientific cooperation framework between CNRS and the Polish Academy of Science. Participant to that collaboration are Éric Badouel and Philippe Darondeau for Team S4, and Marek Bednarczyk, Andrzej Borzyszkowski and Wieslaw Pawlowski for IPI PAN. A two-week visit in Gdansk of two members of the S4 project and a two-week visit in Rennes of one member of IPI PAN (in May) are planned yearly.

8. Dissemination

8.1. Participation to editorial boards and program committees

Éric Badouel has served in the program committee of the CARI 2005 conference and is the secretary of the steering committee of this conference. He is also member of the editorial board of the ARIMA Journal, and has been co-editor of a CARI 2004 special issue of the journal. Albert Benveniste has participated to the program committees of the CONCUR 2005 and EMSOFT 2005 conferences. Benoit Caillaud has been general chair of the ACSD 2005 held in June 2005 in St Malo, France. He his now serving in the steering committee of this conference. He has participated in the program committee of the FMGALS 2005 workshop. Benoit Caillaud has also served in the juries of the PhD thesis of Guillaume Gardey (IRCCYN, Nantes) and the habilitation thesis of Rémi Morin (U. de Provence, Marseille). Philippe Darondeau has been program committee co-chair of the ATPN 2005 conference. He is the secretary of the IFIP WG2.2 working group.

8.2. Demo development

We have developed, with the help of an associate engineer (Agnès Delhaye), a software which shows a demo on control synthesis for a small factory. The demo presents a graphical 3D environment of the working factory and with a simulation of controllers' decisions - for a predefined set of properties -. The controllers are computed beforehand by using existing toolboxes such as UMDES and DESUMA [38]. The software is made as open as possible in the following sense : a demo can be created for an arbitrary number synchronized finite states automata, which are connected to a graphical 3D environment in order to operate on it. The software is programmed in the language oRis [34] for Linux or Windows systems.

8.3. 68NQRT: Theory of computing seminar at Irisa

Sophie Pinchinat organizes the 68NQRT seminar session of Irisa, with the help of another colleague: each week, a scientific talks is given by a local or an invited speaker, in the following research areas: software engineering, theoretical computer science, discrete mathematics, artificial intelligence. The seminar keeps going successfully, as this year, about 30 presentations were given, among which two thirds given by external people.

8.4. Teaching

Teaching related to research undertaken in Team S4 is listed below:

- Master of Computer Science, University of Rennes 1, second year: Benoît Caillaud and Sophie Pinchinat are teaching a course on *formal methods for the verification of reactive systems*.
- Eric Badouel is teaching an advanced course on functional programming in the Second year of the Master of Research in Computer Science, University of Yaoundé 1, Cameroon.

9. Bibliography

Major publications by the team in recent years

[1] E. BADOUEL, M. BEDNARCZYK, P. DARONDEAU. *Generalized Automata and their Net Representations*, H. EHRIG, G. JUHÁS, J. PADBERG, G. ROZENBERG (editors). , Lecture Notes in Computer Science, vol. 2128, Springer, 2001, p. 304–345, http://link.springer.de/link/service/series/0558/bibs/2128/21280304.htm.

[2] E. BADOUEL, B. CAILLAUD, P. DARONDEAU. *Distributing Finite Automata through Petri Net Synthesis*, in "Journal on Formal Aspects of Computing", vol. 13, 2002, p. 447–470, http://dx.doi.org/10.1007/s001650200022.

[3] E. BADOUEL, P. DARONDEAU. *Theory of regions*, Lecture Notes in Computer Science, vol. 1491, Springer, 1999, p. 529–586.

[4] A. BENVENISTE, B. CAILLAUD, P. LE GUERNIC. *Compositionality in dataflow synchronous languages: specification and distributed code generation*, in "Information and Computation", vol. 163, 2000, p. 125-171.

[5] A. BENVENISTE, L. CARLONI, P. CASPI, A. SANGIOVANNI-VINCENTELLI. *Heterogeneous reactive systems modeling and correct-by-construction deployment*, in "Embedded software, third international conference, EMSOFT 2003", R. ALUR, I. LEE (editors). , Lecture notes in computer science, vol. 2855, Springer, oct 2003, p. 35–50, http://www.springerlink.com/openurl.asp?genre=article&issn=0302-9743&volume=2855&spage=35.

[6] A. BENVENISTE, P. CASPI, S. EDWARDS, N. HALBWACHS, P. LE GUERNIC, R. DE SIMONE. *The Synchronous Languages Twelve Years Later*, in "Proceedings of the IEEE", Special issue on modeling and design of embedded software, vol. 91, n^o 1, 2003, p. 64–83, http://www.irisa.fr/s4/download/papers/Benveniste-proc-ieee-2003.pdf.

[7] B. CAILLAUD, P. DARONDEAU, L. HÉLOUËT, G. LESVENTES. *HMSCs as specifications... with PN as completions*, F. CASSEZ, C. JARD, B. ROZOY, M. DERMOT (editors). , Lecture Notes in Computer Science, vol. 2067, Springer, 2001, p. 125–152, http://www.irisa.fr/s4/download/papers/hmsc2pn_movep2k_lncs.ps.gz.

[8] H. MARCHAND, S. PINCHINAT. *Supervisory Control Problem using Symbolic Bisimulation Techniques*, in "2000 American Control Conference, Chicago, Illinois, USA", jun 2000, p. 4067–4071.

[9] S. RIEDWEG, S. PINCHINAT. *Quantified Mu-Calculus for Control Synthesis*, in "MFCS 2003, 28th International Symposium on Mathematical Foundations of Computer Science", Lecture notes in computer science, vol. 2747, Springer, aug 2003, p. 642–651, http://www.springerlink.com/openurl.asp?genre=article&issn=0302-9743&volume=2747&spage=642.

Doctoral dissertations and Habilitation theses

[10] G. FEUILLADE. *Spécification logique de réseaux de Petri*, Thèse de doctorat, Université de Rennes 1, école doctorale MATISSE, dec 2005.

Articles in refereed journals and book chapters

[11] A. BORZYSZKOWSKI, P. DARONDEAU. *Transition systems without transitions*, in "Theoretical Computer Science", vol. 338, 2005, p. 1-16.

[12] P. DARONDEAU. *Equality of languages coincides with isomorphism of reachable state graphs for bounded and persistent Petri nets*, in "Information Processing Letters", vol. 94, 2005, p. 241-245.

[13] G. FEUILLADE, S. PINCHINAT. *Spécifications modales de réseaux de Petri.*, Modélisation des Systèmes Réactifs, Autrans-Grenoble, France, 5-7 Octobre 2005, vol. 39, n^o 1–3, oct 2005.

[14] S. PINCHINAT, S. RIEDWEG. *A Decidable Class of Problems for Control under Partial Observation*, in "Information Processing Letters", vol. 95, n^o 4, August 2005, p. 454–465, http://www.lsv.ens-cachan.fr/Publis/PAPERS/PDF/PR-IPL05.pdf.

[15] D. POTOP-BUTUCARU, B. CAILLAUD, A. BENVENISTE. *Concurrency in Synchronous Systems*, in "Formal Methods in System Design", To appear, 2005, http://www.irisa.fr/s4/download/papers/potopFMSD2005.pdf.

[16] D. POTOP-BUTUCARU, R. DE SIMONE, J.-P. TALPIN. *The Synchronous Hypothesis and Synchronous Languages*, R. ZURAWSKI (editor). , CRC Press, 2005.

Publications in Conferences and Workshops

[17] E. BADOUEL, J. CHENOU, G. GUILLOU. *Petri Algebras*, in "Automata, Languages and Programming, ICALP 2005", L. CAIRES, G. ITALIANO, L. MONTEIRO, C. PALAMIDESSI, M. YUNG (editors). , Lecture Notes in Computer Science, vol. 3580, Springer, 2005, p. 742–754, http://dx.doi.org/10.1007/11523468_60.

[18] A. BENVENISTE, B. CAILLAUD, L. P. CARLONI, A. L. SANGIOVANNI-VINCENTELLI. *Tag Machines*, in "Proceedings of the fifth ACM International Conference on Embedded Software (Emsoft), Jersey City, NJ, USA", ACM Press, September 2005, p. 255–263, http://doi.acm.org/10.1145/1086228.1086276.

[19] P. DARONDEAU. *Distributed implementations of Ramadge-Wonham supervisory control with Petri nets*, in "44th IEEE Conference on Decision and Control and European Control Conference, Sevilla, Spain", to appear, december 2005.

[20] S. DASGUPTA, D. POTOP-BUTUCARU, B. CAILLAUD, A. YAKOVLEV. *From weakly endochronous systems to delay-insensitive circuits*, in "Proceedings of the second international workshop on formal methods for globally asynchronous locally synchronous design (FMGALS 2005)", 2005.

[21] D. POTOP-BUTUCARU, B. CAILLAUD. *Correct-by-construction asynchronous implementation of modular synchronous specifications*, in "Proceedings of the Fifth International Conference on Application of Concurrency to System Design, ACSD 2005", 2005, http://www.irisa.fr/s4/download/papers/Potop-acsd2005.pdf.

[22] J. RACLET, S. PINCHINAT. *The control of non-deterministic Systems: a logical approach*, in "Proc. 16th IFAC Word Congress, Prague, Czech Republic", jul 2005.

[23] S. RIEDWEG, S. PINCHINAT. *On the Architectures in Decentralized Supervisory Control.*, in "44th IEEE Conference on Decision and Control and European Control Conference ECC 2005, Seville, Spain", dec 2005.

[24] S. RIEDWEG, S. PINCHINAT. *You Can Always Compute Maximally Permissive Controllers Under Partial Observation When They Exist*, in "Proc. 2005 American Control Conference., Portland, Oregon", jun 2005.

[25] J.-P. TALPIN, D. POTOP-BUTUCARU, J. OUY, B. CAILLAUD. *From multi-clocked synchronous processes to latency-insensitive modules (short paper)*, in "Proceedings of the fifth ACM International Conference on Embedded Software (Emsoft), Jersey City, NJ, USA", ACM Press, September 2005, p. 282–285, http://doi.acm.org/10.1145/1086228.1086279.

Internal Reports

[26] E. BADOUEL, M. BEDNARCZYK, A. BORZYSZKOWSKI, B. CAILLAUD, P. DARONDEAU. *Concurrent Secrets*, Rapport de recherche, n° 5771, INRIA, nov 2005, http://www.inria.fr/rrrt/rr-5771.html.

[27] G. FEUILLADE. *Modal specifications are a syntactic fragment of the Mu-calculus*, Research Report, n° RR-5612, INRIA, jun 2005, http://www.inria.fr/rrrt/rr-5612.html.

[28] J.-P. TALPIN, D. POTOP-BUTUCARU, J. OUY, B. CAILLAUD. *Compositional synthesis of latency-insensitive systems from multi-clocked synchronous specifications*, Research report, n° 1730, IRISA, jun 2005, http://www.irisa.fr/bibli/publi/pi/2005/1730/1730.html.

Bibliography in notes

[29] B. BOUYSSOUNOUSE, J. SIFAKIS (editors). *Embedded Systems Design: The ARTIST Roadmap for Research and Development*, Lecture Notes in Computer Science, vol. 3436, Springer, 2005, http://www.springerlink.com/openurl.asp?genre=volume&id=doi:10.1007/b106761.

[30] *ITU-TS Recommendation Z.120: Message Sequence Chart (MSC)*, International Telecommunication Union, Geneva, September 1993, http://www.itu.int/home/index.html.

[31] *OMG Unified Modeling Language, version 2.0*, October 2003, http://www.omg.org/uml/, Draft specification.

[32] A. BENVENISTE, B. CAILLAUD, L. CARLONI, P. CASPI, A. SANGIOVANNI-VINCENTELLI. *Heterogeneous Reactive Systems Modeling: Capturing Causality and the Correctness of Loosely Time-Triggered Architectures (LTTA)*, in "Proceedings of the Fourth ACM International Conference on Embedded Software, EMSOFT'04", vol. September, ACM Press, September 2004, http://doi.acm.org/10.1145/1017753.1017790.

[33] P. DARONDEAU. *Controlled Shuffle of Message Sequence Charts*, June 2005, Deliverable to the CRE CO2 collaboration.

[34] F. HARROUET. *oRis : s'immerger par le langage pour le prototypage d'univers virtuels à base d'entités autonomes*, http://www.enib.fr/~harrouet/.

[35] H. KOPETZ. *Real-Time Systems, Design Principles for Distributed Embedded Applications*, Kluwer Academic Publishers, 1997.

[36] E. STARK. *Concurrent Transition Systems*, in "Theoretical Computer Science", vol. 64, 1989, p. 221–269.

[37] THE ARTIST CONSORTIUM. *Guidelines for a Graduate Curriculum on Embedded Software and Systems*, Roadmap Report, Artist, 2003, http://www.artist-embedded.org/Education/index.html.

[38] UMDES. *The DESUMA Software and the UMDES Software Library*, http://www.eecs.umich.edu/umdes/toolboxes.html.