INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Project-Team S4

System Synthesis and Supervision, Scenarios

Rennes - Bretagne Atlantique

THEME COM

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 [Research Associate (CR)]

Administrative assistant

Laurence Dinh [TR, part-time in S4]

Research scientists

Éric Badouel [Research Associate (CR), HdR]
Albert Benveniste [Research Director (DR), part-time in S4, HdR]
Philippe Darondeau [Research Director (DR), HdR]
Sophie Pinchinat [Lecturer, University of Rennes 1, on sabbatical leave at the Australian National University until June 2007, then in delegation at INRIA. Funded by a Marie Curie European grant, HdR]
Laurie Ricker [Visiting scientist on sabbatical leave from Mount Allison University, Sackville, Canada, from September 2007]

Ph. D. students

Benoît Delahaye [ENS Cachan, Antenne de Bretagne, until August 2007, then University of Rennes 1]
Rodrigue Djeumen [Funded by SCAC Yaoundé (Service de Coopération et d'Action Culturelle de l'Ambassade de France), part time in France]
Bernard Fotsing [Funded by AUF (Agence universitaire de la Francophonie), part time in France]
Mateus Oliveira [Started October 2007. Funded by the SPEEDS european project]
Jean-Baptiste Raclet [Funded by the CO2 collaboration with France Telecom]
Rodrigue Tchougong [Funded by SARIMA, part time in France]
Maurice Tchoupé [Funded by SCAC Yaoundé (Service de Coopération et d'Action Culturelle de l'Ambassade de France), 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* (UML) [32] (sequence diagrams and statecharts), the *High-Level Message Sequence Charts* (HMSC) [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 environments, 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 is 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.

A set of 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.

The contribution of the S4 Project-Team consists of 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 consistent?) and mapped to lower-level specifications (for instance, communicating automata, or Petri nets).

The scientific approach 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 this 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 four main tracks:

Petri net synthesis: This track follows up the main research theme of the former Team PARAGRAPHE at INRIA Rennes on the synthesis of Petri net models using the theory of regions.

Heterogeneous systems: This track contributes to the extension of the well-established synchronous paradigm to distributed systems. 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 with guarantees on its environment.We are investigating questions related to reactive component refinement and composition. We are also investigating the issues of coherence of views and modularity in complex specifications.

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, extensively defined 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* [32] 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 at providing 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 can be reduced 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 of a distributed hardware architecture, and software to be deployed over that architecture. A particular emphasis is put on *embedded* systems (automotive, avionics, production systems, *etc.*), 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.

Our work on heterogeneous reactive systems facilitates the mapping of pure synchronous designs onto 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 of recent Airbus aircrafts. 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 6.1) 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) has 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. Synthesis of Petri nets

Keywords: *Petri net*, *modular automata*, *synthesis of Petri nets*.

Participant: Philippe Darondeau.

A serious limitation of all known procedures for synthesizing Place/Transition-nets based on regions of graphs or languages is to require computing the full state space of the input transition system. In order to avoid the state space explosion, an attempt towards *modular synthesis* has been made with Laure Petrucci (University of Villetaneuse, Paris). Laure Petrucci's modular automata stay in between indexed families of automata $(A_i)_{i=1}^n$ and their mixed products $\otimes_i A_i$. In a modular automaton $((A'_i)_{i=1}^n, \mathcal{S})$, the modules A'_i are the residues left after all synchronized transitions have been exported to the synchronization graph $\mathcal{S}$. A modular synthesis procedure has been designed for *distributed P/T-nets*. The improvement of performance w.r.t. non-modular synthesis is most significant for the reversible automata. A first version of this work appears in [28].

5.2. Heterogeneous systems

Keywords: *Heterogeneous systems*, *desynchronization*, *endochrony*, *isochrony*.

Participants: Albert Benveniste, Benoît Caillaud.

In [14], A. Benveniste *et al.* address the problem of mapping a set of processes which communicate synchronously on a distributed platform. The Time Triggered Architecture (TTA) proposed by H. Kopetz for the communication mechanism of a distributed platform offers a direct mapping that would preserve the semantics of the specification. However, its exact implementation may, at times, be problematic as it requires the distributed platform to have the clocks of its components synchronized with great precision. We propose as implementation architecture a relaxation of TTA called Loosely Time-Triggered Architecture (LTTA), in which writes and reads on distributed shared variables are scheduled by quasi-periodic, but non synchronized local clocks. LTTA offers some of the advantages of TTA with lower hardware cost and greater flexibility. In previous work, LTTA has been studied on uni-directional communications and two-nodes architectures only. In this paper the authors propose a design flow that ensures semantics preservation for LTT communication networks with arbitrary topology. Key elements are two new protocols for clock regeneration and predictive traffic shaping.

Future work will be focused on the axiomatization of the class of deterministic networks of LTT processes. This class corresponds exactly to those networks of synchronous processes which can be safely mapped to LTT architectures. The objective is to provide either design/programming rules or a library of generic components ensuring that any design using this rules/components belong to that class.

5.3. Reactive components

Keywords: *behavioral type*, *coherence of views*, *interface*, *partial views*, *residual specification*.

Participants: Éric Badouel, Albert Benveniste, Benoît Caillaud, Benoît Delahaye, Philippe Darondeau, Rodrigue Djeumen, Bernard Fotsing, Rodrigue Tchougong, Maurice Tchoupé, Jean-Baptiste Raclet.

5.3.1. Partial Views

5.3.1.1. Controlled Shuffle of Message Sequence Charts.

In 2005, we addressed the problem of assembling partial views of the behavior of a distributed system, and introduced for this purpose an operation of controlled shuffle on languages of (compositional) message sequence charts (MSC). The operation of controlled shuffle of two MSCs amounts, for each process, to interleave the send/receive events of these two components, and to amalgamate their synchronization events, without ever introducing circularity in the resulting flow. In this context, a subclass of MSC languages, called the existentially-bounded MSCs, seems to be especially interesting, since no implementation can reasonably be envisaged for MSC languages that do not belong to this class. Answers have been provided for the main decision problems concerning controlled shuffle and existential boundedness. Whether the controlled shuffle of two existentially-bounded languages of MSCs is existentially-bounded is undecidable. However, this problem is decidable when the two component MSCs only synchronize on a single process. The construction of a product of MSC graph has been fully defined in the existentially-bounded case. These results, obtained in cooperation with Blaise Genest and Loïc Hélouët (DISTRIBCOM project), are presented in [27].

5.3.1.2. Coherence of views.

A structured document, described as a tree decorated with attributes, can be used as an interface between various teams of designers involved in different aspects of an heterogeneous specification of a complex system. The set of legal structures are given by an abstract context-free grammar. We forget about the attributes which are related with semantical issues that can be treated independently of the purely structural aspects considered here. This abstract representation may be asynchronously manipulated by a set of independent tools, each of which operates on a distinct partial view of the whole structure. In order to synchronize these multiple partial views, we face the problem of their coherence: can we decide whether there exists some global structure corresponding to a given set of partial views, and in the affirmative, can we compute such a global

structure? In [13] we solve this problem in the case where a view is given by a subset of grammatical symbols, those associated with the so-called visible syntactical categories. The proposed algorithm strongly relies on the mechanism of lazy evaluation. It produces an answer to this problem even when the partial views may correspond to an infinite set of related global structures.

5.3.2. Models of Components

5.3.2.1. Reuse of Components through Residual Specifications.

A component is usually provided with an interface which lists the signature of the services offered by the entity. This light description is sufficient to enable component reuse. However, it provides no guarantee that the reused component will suitably interact with its environment, and critical behavioral mismatch such as deadlock may occur. In [29], [21], [10], we have studied the problem of component reuse at a behavioral level rather than at a signature level. We have addressed the behavioral reuse of a component by describing a quotient operation. Starting from the specifications of the behaviors of the component and of the desired overall system, this operation computes the residual specification characteristic of the systems that, when composed with the given components, satisfy the overall specification. This problem is solved when behaviors are given by modal specifications and when composition allows mixed product and internalization of events. This work has been extended to acceptance specifications to deal with weak form of liveness properties. This quotient operation can be applied to the adaptor synthesis problem, where a supervisor is synthesized to mediate the composition of two components.

5.3.2.2. Heterogeneous Rich Components Models.

The SPEEDS European project (see Section 6.2), started on May 1st, 2006, is a concerted effort to define a new generation of end-to-end methodologies and supporting tools for safety-critical embedded system design. The technology developped in SPEEDS is based on the concept of heterogeneous rich components, allowing for the description of the expected behavior of a component, on a per-viewpoint basis (functional, timing, reliability, resource usage, etc.), thanks to an assume/guarantee style of reasoning. This formalism enables scalable modular analysis methods capable of checking the realizability of a virtual system model at a early stage of design. Since the beginning of the project, we have contributed to the mathematical semantics of the Heterogeneous Rich Component formalism [26].

5.3.3. Domain-Specific Languages

Language-oriented programming puts emphasis on the use of domain-specific languages (DSL) dedicated with specific aspects of some application domain. We intend to develop a theory of behavioral types, in terms of assume and guarantee conditions, for domain-specific languages by taking inspiration of what has been done for component reuse. A DSL relies on an intentional representation of a program, dissociated from its more or less partial concrete views, that can be manipulated by metaprogramming tools in order to edit, navigate, transform or extract information from this abstract representation. Thus the interface of a DSL with a host language is given by a set of methods for respectively manipulating the intentional representation (through a coalgebra) and for extracting information from it (through an algebra). Combining such languages requires considering a global grammar such that each DSL is associated with some subgrammar. The global grammar need not be explicitly constructed but we should be able to generate (respectively to evaluate) its abstract syntax trees by combining anamorphisms (respectively catamorphisms) of the corresponding subgrammars. We have addressed this problem in [23] where a splitting operation on algebras that relates algebras associated with the a global system to the algebras associated, through Bekìc decomposition, with respectively one of its subsystem and the corresponding residual system. The similar result holds, by duality, for coalgebras and the related generating functions (anamorphisms). We have also preliminary results on the extensions of domain specific languages [25] and a new implementation of attribute evaluation [24].

5.4. Discrete event system synthesis and supervisory control

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

Participants: Eric Badouel, Benoît Caillaud, Philippe Darondeau, Sophie Pinchinat, Laurie Ricker.

5.4.1. Concurrent secrets

We have started a new topic of research by considering the definition and computation of finite and optimal control (of discrete event systems) in the perspective of computer security. Given a discrete event system with regular behavior $L \subseteq \Sigma^*$ and a subset of observable actions $\Sigma_o \subseteq \Sigma$, a *secret set* $S \subseteq L$ is defined as a regular subset of trajectories of the system. The secret is *opaque* if observing actions from Σ_o does not allow to decide whether the trajectory of the system is in S. A *concurrent secret* is specified similarly with a n-tuple $\mathcal{S} = \{(\Sigma_1, S_1), ..., (\Sigma_n, S_n)\}$ where each S_i specifies a secret set to be protected against an observer with the set of observable actions Σ_i. The concurrent secret is *concurrently opaque* if all sets S_i are opaque. Checking concurrent opacity is rather easy. What is more problematic is to compute the largest subset K of L such that $\mathcal{S}$ is concurrently opaque in restriction to K. We give sufficient conditions under which this largest restriction K, which always exists, is regular and can be computed. Remarkably, concurrent opacity can always be enforced by decentralized control when all the actions are controllable. The complete results, obtained in cooperation with Marek Bednarczyk and Andrzej Borzyszkowski (in the realm of joint research action CATALYSIS) appear in [11]. An extension to the case where some actions are uncontrollable is under investigation with J. Dubreil and H. Marchand from the VERTECS team-project.

5.4.2. Synthesis of Strategies and Plans]

5.4.2.1. Logic for strategies in concurrent games.

The emerging technology of interacting systems calls for new formalisms to ensure their reliability. Concurrent games are paradigmatic abstract models for which several logics have been studied. However, the existing formalisms show certain limitations in face of the range of strategy properties required to address intuitive situations. We have proposed in [20] a generic solution to specify expressive constraints on strategies in concurrent games. Our formalism naturally extends alternating-time logics while being highly flexible to combine constraints. Our approach is constructive and can synthesize many types of complex strategies, via automata-theoretic techniques. Morever, it subsumes recent proposals such as Strategic Logic.

5.4.2.2. Automated planning

Automated planning is a field of Artificial Intelligence which aims at developing concepts and methods to synthesize a plan, that is a way to select and organize a sequence of actions of the system to achieve some objective. When the system is "open" (interacting with an environment), plans are no more sequences but trees: the tree structure represents the alternative responses of the environment when a given planned action is performed. For open systems, the objectives only involve the system; the environment has nothing to achieve in particular, it is just "adversarial", and results have already been published. Generalizing the framework to more than one entity has not been much investigated so far, and yet applications area include web services, or component-based systems. We consider the general setting where several planners coexist, and we follow a natural approach: following the successful adaptation of temporal logics model-checking methods to classic planning, we have explored variants of the model-checking of richer logics designed for multi-player games (e.g. alternating-time temporal logics).

5.4.3. Systems with Imperfect Information: Diagnosis of ω-Regular Properties

Diagnosis problems concern the construction of a device, called a diagnoser, which, during the execution of a given partially observable system, can detect/identify the occurrence of particular (sequences of) events, called patterns. The diagnoser can thereby be used on line to e.g. activate an alarm when an undesirable fault has occurred. Importantly, an occurrence of the pattern might not be detected immediately after this occurrence, but ascertained if the observation is extended for a long enough duration. The diagnosability problem takes this degree of freedom into consideration: given a model of the system and a pattern on its sequence of events, it addresses the question of whether an occurrence of the pattern can be detected in some fixed bounded delay. As witnessed by our results last year, algorithms for checking diagnosability exist for simple co-safety patterns, e.g., permanent faults, multiple faults, fault sequences.

This year, we have extended our approach to capture a broader class of patterns. The adequat setting is a shift to infinitary patterns described by non-deterministic Buchi automata, hence ω-regular. In general, the relevance of an on-line diagnoser is questionable as infinitary patterns may require an infinitely-long observation of the system. And indeed, we have exhibited two necessary and sufficient conditions to guarantee a verdict within a bounded delay. These conditions decompose into the standard diagnosability (as for the case of "simple" co-safety patterns), and the additional condition of openess (a topological notion on infinitary languages, matching co-safety in temporal logics); the latter condition meets consideration in systems' monitoring.

5.4.4. Expanding Communication Options in Decentralized Discrete-Event Control

In [22], L. Ricker and B. Caillaud propose an effective procedure to identify the range of possible communication locations for decentralized supervisory control. This work is the first element of a theoretical framework for optimizing communication for decentralized supervisory control.

6. Other Grants and Activities

6.1. ARTIST II – Network of Excellence on Advanced Real-Time Systems

Participants: Albert Benveniste, Benoît Caillaud.

IST-004527 ARTIST2 (September 2004, September 2008), http://www.artist-embedded.org/artist/

Until 2006, Albert Benveniste was leading the Real-Time Components cluster as well as the INRIA team. Since 2007, the Real-Time Components is headed by Bengt Jonsson and the INRIA team is headed by Alain Girault.

As part of his participation to this NoE, Albert Benveniste has organized, jointly with Paul Caspi, a workshop on *Integrated Modular Avionics* (IMA) http://www.artist-embedded.org/artist/-ARTIST2-meeting-on-Integrated-.html. This meeting was held in Rome, November 12-13, 2007. Participants included speakers from industry (Airbus, Dassault-Aviation, IAI, Honeywell, Windriver, WWW Technology Group, SAE AADL Committee) and academia.

Benoît Caillaud has given a two-hour lecture on the composition and transformation of heterogeneous real-time systems at the MOdelling TestIng and Verification for Embedded Systems (MOTIVES'07) graduate school, sponsored by ARTIST2.

6.2. SPEEDS: Speculative and Exploratory Design in Systems Engineering

Participants: Éric Badouel, Albert Benveniste, Benoît Caillaud, Benoît Delahaye, Mateus Oliveira, Jean-Baptiste Raclet.

Started in May 2006, the SPEEDS project is a FP6 European integrated project. SPEEDS is a concerted effort to define a new generation of end-to-end methodologies, processes and supporting tools for safety-critical embedded system design. They will enable the European systems industry to evolve from model-based design of hardware/software systems, towards integrated, component-based construction of complete virtual system models.

Core partners of the project come from both academia (OFFIS, PARADES, Verimag and INRIA), aeronautics (Airbus, SAAB and IAI), the automotive industry (Daimler-Chrysler, Bosch, Knorr Bremse, Magna Powertrain) and tool vendors (Esterel Technologies, Extessy, Telelogic and TNI).

The main objective of the SPEEDS project is to develop a modeling formalism, the Heterogeneous Rich Component formalism (HRC), capable of supporting not only scalable modular analysis methods for component based design, but also speculative design processes where several teams work in parallel on a design, one team making assumptions about the subsystem designed by another team.

A component in the HRC is described as a set of assume/guarantee contracts which explicates assumptions about its environment and state corresponding guarantees on the service offered by the component to its environment. Contracts fall in several categories, in which both functional and non-functional (timing, reliability, resource consumption, etc.) properties of the component's behavior can be expressed [26].

The HRC formalism is built on top of existing standard (UML and SysML of the OMG) and will be implemented as an Eclipse plugin, using state-of-the-art meta-modeling technology [31]. The HRC Eclipse plugin will have gateways with existing IDE tools, including SCADE (Esterel Tech.), Rapsody (Telelogic), RT-Builder (TNI) and MatLab/Simulink.

6.3. Odas: categorical and algebraic approaches to synthesis

Participants: Éric Badouel, Philippe Darondeau, Jean-Baptiste Raclet.

ODAS stands for *Open Dynamic Agent Systems*. It is a follow up of CATALYSIS (*categorical and algebraic approaches to synthesis*), 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. Participants to this collaboration are Éric Badouel, Philippe Darondeau, and Jean-Baptiste Raclet for Team S4, and Marek Bednarczyk, Andrzej Borzyszkowski and Wieslaw Pawlowski for IPI PAN. Visits in Gdansk of members of the S4 project and visits in Rennes of members of IPI PAN take place every year.

7. Dissemination

7.1. Participation to editorial boards and program committees

Éric Badouel is the secretary of the steering committee of CARI, the African Conference on Research on Computer Science and Applied Mathematics. He is a member of the editorial board of the ARIMA Journal.

Albert Benveniste is associated editor at large (AEAL) for the journal *IEEE Trans. on Automatic Control*. He is PC member for the conference Emsoft 2007. He is member of the Strategic Advisory Council of the Institute for Systems Research, Univ. of Maryland, College Park, USA. He belongs to the Scientific Advisory Board of INRIA, where he is in charge of the area of Embedded Systems.[1]

Benoît Caillaud has been a member of the steering and program committees of the 7th International Conference on Application of Concurrency to System Design (ACSD'07). He has also been in the program committee of the Workshop on Model-driven High-level Programming of Embedded Systems (SLA++P'07).

Philippe Darondeau is the secretary of the IFIP WG2.2 working group.

Sophie Pinchinat has entered the Advisory Board of the Marie Curie Fellows Association as Public Relations Associate and Science Policy Advisor.

7.2. 68NQRT: Theory of computing seminar of Irisa

Sophie Pinchinat takes part to the organization of the 68NQRT seminar series of IRISA, dedicated to software engineering, theoretical computer science, discrete mathematics, and artificial intelligence.

[1] Only facts related to the activities of Team S4 are mentioned. Other roles or duties concern the DistribCom or Sisthem teams, to which A. Benveniste also belongs.

7.3. Teaching

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

- 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.
- Benoît Caillaud gave a two-hour lecture on the composition and transformation of heterogeneous real-time systems at the MOdelling TestIng and Verification for Embedded Systems (MOTIVES'07) graduate school, sponsored by the ARTIST2 NoE.
- Philippe Darondeau gave six hours of lectures on the Synthesis of Petri Nets to PhD students in Milano Bicocca (February 2007)

8. 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, 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.

Year Publications

Doctoral dissertations and Habilitation theses

[10] J.-B. RACLET. *Quotient de spécifications pour la réutilisation de composants*, Ph. D. Thesis, École doctorale Matisse, université de Rennes 1, 2007.

Articles in refereed journals and book chapters

[11] E. BADOUEL, M. BEDNARCZYK, A. BORZYSZKOWSKI, B. CAILLAUD, P. DARONDEAU. *Concurrent Secrets*, in "Discrete Event Dynamic Systems", 2007.

[12] E. BADOUEL, J. CHENOU, G. GUILLOU. *An axiomatization of the token game based on Petri algebras*, in "Fundamenta Informaticae", vol. 77, 2007, p. 187-292.

[13] E. BADOUEL, M. TCHOUPÉ. *Projections et cohérence de vues dans les grammaires algébriques*, in "Revue ARIMA", 2007, http://www-direction.inria.fr/international/arima/.

[14] A. BENVENISTE, B. CAILLAUD, L. P. CARLONI, P. CASPI, A. L. SANGIOVANNI-VINCENTELLI. *Composing Heterogeneous Reactive Systems*, in "ACM Transactions on Embedded Computing Systems (TECS)", to appear, 2007, http://www.irisa.fr/s4/download/papers/TECS-2005-0054-final.pdf.

[15] E. BEST, P. DARONDEAU, H. WIMMEL. *Making Petri Nets Safe and Free of Internal Transitions*, in "Fundamenta Informaticae", 2007.

[16] G. FEUILLADE, S. PINCHINAT. *Modal Specifications for the Control Theory of Discrete-Event Systems.*, in "Discrete Event Dynamic Systems", vol. 17, n^{o} 2, 2007, p. 211–232.

[17] P. POTOP-BUTUCARU, B. CAILLAUD. *Correct-by-Construction Asynchronous Implementation of Modular Synchronous Specifications*, in "Fundamenta Informaticae", vol. 78, n^{o} 1, 2007, p. 131–159.

Publications in Conferences and Workshops

[18] A. BENVENISTE, P. CASPI, M. DI NATALE, C. PINELLO, A. L. SANGIOVANNI-VINCENTELLI, S. TRIPAKIS. *Loosely Time-Triggered Architectures based on Communication-by-Sampling*, in "EMSOFT", 2007.

[19] P. DARONDEAU. *Synthesis and Control of Asynchronous and Distributed Systems*, in "78th International Conference on Application of Concurrency to System Design, Bratislava, Slovak Republic", T. BASTEN, G. JUHAS, S. SHUKLA (editors), 2007.

[20] S. PINCHINAT. *A generic constructive solution for concurrent games with expressive constraints on strategies.*, in "5th International Symposium on Automated Technology for Verification and Analysis, Tokyo, Japan", 2007.

[21] J.-B. RACLET. *Residual for Component Specifications*, in "Proceedings of the 4th International Workshop on Formal Aspects of Component Software, Sophia-Antipolis, France", 2007.

[22] L. RICKER, B. CAILLAUD. *Mind the Gap: Expanding Communication Options in Decentralized Discrete-Event Control*, in "46th IEEE Conference on Decision and Control, New Orleans, LA, USA", 2007.

Internal Reports

[23] E. BADOUEL, R. DJEUMEN. *Modular Grammars and Splitting of Catamorphisms*, Research Report, n^{o} 6313, INRIA, 2007, https://hal.inria.fr/inria-00175793.

[24] E. BADOUEL, B. FOTSING, R. TCHOUGONG. *Yet Another Implementation of Attribute Evaluation*, Research Report, n^{o} 6315, INRIA, 2007, https://hal.inria.fr/inria-00175810.

[25] E. BADOUEL, M. TONGA. *Growing a Domain Specific Language with Split Extensions*, Research Report, n^{o} 6314, INRIA, 2007, https://hal.inria.fr/inria-00175805.

[26] A. BENVENISTE, B. CAILLAUD, R. PASSERONE. *A Generic Model of Contracts for Embedded Systems*, Research report, n^{o} 6214, INRIA Rennes, 2007, https://hal.inria.fr/inria-00153477.

[27] P. DARONDEAU, B. GENEST, L. HÉLOUËT. *Products of Message Sequence Charts*, Research Report, n^{o} 6258, INRIA, 2007, https://hal.inria.fr/inria-00156035.

[28] P. DARONDEAU, L. PETRUCCI. *Modular Automata 2 Distributed Petri Nets 4 Synthesis*, Research Report, n^{o} 6192, INRIA, 2007, https://hal.inria.fr/inria-00148133.

[29] J.-B. RACLET. *Residual for Component Specifications*, Research Report, n^{o} 6196, INRIA, 2007, https://hal.inria.fr/inria-00141898.

References in notes

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

[31] *D.2.1.b SPEEDS Meta-model Syntax and Static Semantics*, 2007, SPEEDS project deliverable.

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