INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Project-Team DaRT

Dataparallelism for Real-Time

Futurs

THEME COM

Activity Report

2006

Table of contents

1. Team

DaRT is a common project with the University of Science and Technologies of Lille (USTL), via the Laboratory of Fundamental Computer Science of Lille (LIFL, associated to the CNRS as UMR 8022).

Team Leader

Jean-Luc Dekeyser [Professor, Université des Sciences et Technologies de Lille, HdR]

Administrative assistant

Karine Lewandowski [INRIA]

Faculty members

Pierre Boulet [Professor, Université des Sciences et Technologies de Lille, HdR]
Philippe Marquet [Associate professor, Université des Sciences et Technologies de Lille]
Samy Meftali [Associate professor, Université des Sciences et Technologies de Lille]
Cédric Dumoulin [Associate professor, Université des Sciences et Technologies de Lille]

Research scientist

Abdoulaye Gamatié [CR CNRS, since oct. 2006]
Eric Rutten [CR INRIA, until sept. 2006, HdR]

External Collaborator

Smaïl Niar [Associate Professor, Université de Valenciennes et du Hainaut-Cambrésis, HdR]

Post-doctoral fellows

Emmanuel Renaux [Post-doctoral fellow, INRIA Futurs, since sept. 2006]
Anne Etien [Post-doctoral fellow, INRIA Futurs, since sept. 2006]
Arnaud Cuccuru [Post-doctoral fellow, Protes project grant in Aoste/DaRT, until august 2006]
Abdoulaye Gamatié [Post-doctoral fellow, INRIA Futurs, until sept. 2006]

Ph.D. students

Yassine Aydi [co-advising USTL/ENIS Tunis]
Rabie Ben Atitallah [Interreg project grant]
Lossan Bonde [Burkina Fasso grant, until dec 2006]
Ali Koudri [CIFRE grant, Thales, ENSIETA Brest]
Ouassila Labbani [CNRS and regional grant, until dec 2006]
Sébastien Le Beux [Interreg project grant]
Ashish Meena [Interreg project grant, until aug. 2006]
Éric Piel [INRIA Futurs and Regional grant]
Safouan Taha [co-advising, CEA grant]
Julien Taillard [VALEO and Regional grant]
Huafeng Yu [INRIA Futurs and Regional grant]
Calin Glitia [INRIA Futurs grant, since oct 2006]
Imran Rafiq Quadri [HEC Pakistan grant, since sept 2006]
Souha Kamoun [French Minister grant, since oct 2006]

Technical staff

Antoine Honoré [Expert Engineer]
Cyril Joly [Expert Engineer, from sept 2006]

Invited Research scientists

Frédéric Guyomarc'h [University of Rennes 1, from oct 2006 to dec 2006]
Aboub El Hassan Benyamina [Algerian Government grant, since may 2006]

2. Overall Objectives

2.1. Overall Objectives

The 2001 International Technology Roadmap for Semiconductors [74] stresses a new problem in the design of electronic systems. Indeed, we face for the first time a design productivity gap, meaning that electronic system design teams are no longer able to take advantage of all the available transistors on a chip for logic. Because of the super-exponential increase of the difficulty of system design, we may well be in a situation in a few years where one could be forced to use more than 90% of a chip area for memory, because of design costs for anything other than memory.

In the same time, the computing power requirements of intensive signal processing applications such as video processing, voice recognition, telecommunications, radar or sonar are steadily increasing (several hundreds of Gops for low power embedded systems in a few years). If the design productivity does not increase dramatically, the limiting factor of the growth of the semiconductor industry will not be the physical limitations due to the thinness of the fabrication process but the economy! Indeed we ask to the system design teams to build more complex systems faster, cheaper, bug free and decreasing the power consumption...

We propose in the DaRT project to contribute to the improvement of the productivity of the electronic embedded system design teams. We structure our approach around a few key ideas:

- Promote the use of *parallelism* to help reduce the power consumption while improving the performance.
- *Separate the concerns* in different models to allow reuse of these models and to keep them human readable.
- Propose an environment starting at the highest level of abstraction, namely the *system modeling* level.
- *Automate code production* by the use of (semi)-automatic *model transformations* to build correct by construction code.
- Develop *simulation techniques* at precise abstraction levels (functional, transactional or register transfer levels) to check the design the soonest.
- Promote *strong semantics* in the application model to allow verification, non ambiguous design and automatic code generation.
- Focus on a *limited application domain*, intensive signal processing applications. This restriction allows us to push our developments further without having to deal with the wide variety of applications.

All these ideas will be implemented into a prototype design environment based on simulation, Gaspard. This open source platform is our test bench and is freely available.

The main technologies we promote are UML 2.0 [48], MDE [91] and EMF [1] for the modeling and the automatic model transformations; Array-OL [65], [66], [60] and synchronous languages [57] as computation models with strong semantics for verification; SystemC [88] for the simulations; openMPfor parallel execution; VHDL for the synthesis; and Java [55] to code our prototypes.

3. Scientific Foundations

3.1. Introduction

ISP Intensive Signal Processing

[1] http://www.eclipse.org/emf

SoC System-on-Chip

These last few years, our research activities are mainly concerned with data parallel models and compilation techniques. Intensive Signal Processing (ISP) with real time constraints is a particular domain that could benefit from this background. Our project covers the following new trend: a data parallel paradigm for ISP applications. These applications are mostly developed on embedded systems with high performance processing units like DSP or SIMD processors. We focus on multi processor architectures on a single chip (System-on-Chip). To reduce the "time to market", the DaRT project proposes a high level modeling environment for software and hardware design. This level of abstraction already allows the use of verification techniques before any prototyping (as in the Esterel Studio environment from Esterel Technologies [70]). This also permits to automatically produce a mapping and a schedule of the application onto the architecture with code generation (as with the AAA method of SynDEx [94]). The DaRT project contributes to this research field by the three following items:

Co-modeling for SoC design: We define our own metamodels to specify application, architecture, and (software hardware) association. These metamodels present new characteristics as high level data parallel constructions, iterative dependency expression, data flow and control flow mixing, hierarchical and repetitive application and architecture models. All these metamodels are implemented with EMF. Some of them are represented as UML profiles.

Optimization techniques: We develop automatic transformations of data parallel constructions. They are used to map and to schedule an application on a particular architecture. This architecture is by nature heterogeneous and appropriate techniques used in the high performance community can be adapted. New heuristics to minimize the power consumption are developed. This new objective implies to specify multi criteria optimization techniques to achieve the mapping and the scheduling.

SoC simulation: The data flow philosophy of our metamodel is particularly well suited to a distributed simulation.

To take care of the architecture model and the mapping of the application on it, we propose to use the SystemC platform to simulate at different levels of abstraction the result of the SoC design. This simulation allows to verify the adequacy of the mapping and the schedule (communication delay, load balancing, memory allocation...). We also support IP (Intellectual Property) integration with different levels of specification (functional, timed functional, transaction and cycle accurate byte accurate levels).

3.2. Co-modeling for SoC design

Keywords: *MDE*, *Metamodel*, *Model*, *Modeling*, *Transformation*, *UML*.

The main research objective is to build a set of metamodels (application, hardware architecture, association, deployment and platform specific metamodels) to support a design flow for SoC design. We use a MDE (Model Driven Engineering) based approach.

3.2.1. Principles

Because of the vast scope of the encountered problems, of the quick evolution of the architectures, we observe a very great diversity regarding programming languages. Ten years ago each new proposed model (for example within the framework of a PhD) led to the implementation of this model in a new language or at least in an extension of a standard language. Thus a variety of dialects were born, without relieving the programmer of the usual constraints of code development. Portability of an application from one language to another (a new one for example) increases the workload of the programmer. This drawback is also true for the development of embedded applications. It is even worse, because the number of abstraction levels has to be added to the diversity of the languages. It is essential to associate a target hardware architecture model to the application specification model, and to introduce also a relationship between them. These two models are practically always different, they are often expressed in two different languages.

From this experience, one can derive some principles for the design of the next generation of environments for embedded application development:

- To refrain from designing programming languages to express the two different models, application and hardware architecture.
- To profit from all the new systems dedicated to simulation or synthesis without having to reformalize these two models.
- To use a single modeling environment possibly supporting a visual specification.
- To benefit from standard formats for exchange and storage.
- To be able to express transformation rules from model to model. Possibly the transformation tools could be generated automatically from this expression.

We believe that the Model Driven Engineering [91] can allow us to propose a new method of system design respecting these principles. Indeed, it is based on the common UML modeling language to model all kinds of artifacts.

The Model Driven Engineering (MDE) approach advocates the use of models at different levels of abstraction. A model represents an abstract view of the reality, it is defined by a metamodel specifying the available concepts. A common MDE development process is to start from a high level of abstraction and to go to a targeted model by flowing through intermediate levels of abstraction. Usually, the high level models contain only domain specific concepts, while technological concepts are introduced smoothly in the intermediate levels. The targeted levels are used for different purposes: code generation, simulation, verification, or as inputs to produce other models, etc. A key point of the MDE is the transformation between models.

The clear separation between the high level models and the technological models makes it easy to switch to a new technology while re-using the old designs. This may even be done automatically provided the right tool.

3.2.2. Models of computation

We briefly present our main computation models that consist of the Array-OL language and the synchronous model. The former allows us to express the parallelism in applications while the latter favors the formal validation of the design.

3.2.2.1. Array-OL

The Array-OL language [65], [66], [60] gives a mixed graphical-textual language in order to express multidimensional intensive signal processing applications. It relies on the following basic principles:

- All the potential parallelism in an application has to be available in the specification, both *task parallelism* and *data parallelism*.
- Array-OL is a *data dependence expression* language. Only the true data dependencies are expressed in order to express the full parallelism of the application, defining the minimal order of the tasks. Thus any schedule respecting these dependencies will lead to the same result. The language is deterministic.
- It is a *single assignment* formalism. No data element is ever written twice. It can be read several times, though. Array-OL can be considered as a first order functional language.
- Data accesses are done through sub arrays, called *patterns*.
- The language is *hierarchical* to allow descriptions at different granularity levels and to handle the complexity of the applications. The data dependencies expressed at a level (between arrays) are approximations of the precise dependencies of the sub-levels (between patterns).
- The spatial and temporal dimensions are treated equally in the arrays. In particular, time is expanded as a dimension (or several) of the arrays. This is a consequence of single assignment.
- The arrays are toroidal. Indeed, some spatial dimensions may represent some physical tori (think about some hydrophones around a submarine) and some frequency domains obtained by FFTs (Fast Fourier Transformations) are toroidal.

The semantics of Array-OL is that of a first order functional language manipulating multidimensional arrays. It is not a data flow language but can be projected on such a language.

3.2.2.2. Synchronous model

The synchronous approach [57] proposes formal concepts that favor the trusted design of embedded real-time systems. Its basic assumption is that computation and communication are instantaneous (referred to as "synchrony hypothesis"). The execution of a system is seen through the chronology and simultaneity of observed events. This is a main difference from visions where the system execution is rather considered under its chronometric aspect (i.e., duration has a significant role).

There are different synchronous languages, which have strong mathematical foundations. These languages mainly differ from their programming styles. For instance, the Esterel language [58] adopts an imperative style. It is well-suited for the design of control dominated applications. Other languages such as Lustre [61] or Signal [78] rather adopt a declarative style. More specifically, Lustre is functional while Signal is relational. These two languages are well-adapted for dataflow-oriented applications. All these languages are associated with formal tool-sets that have been successfully used in several critical domains (e.g. avionics, automotive, nuclear power plants, etc.).

In the context of the DaRT project, we consider the last family of languages (i.e. declarative languages) to model various refinements of Array-OL descriptions in order to deal with the control aspect as well as the temporal aspect present in target applications. The first aspect is typically addressed by using concepts such as mode automata, which are proposed as an extension mechanism in synchronous declarative languages. The second aspect is studied by considering temporal projections of array dimensions in synchronous languages based on clock notion.

The resulting synchronous models can be then analyzed using the formal techniques and tools provided by the synchronous technology.

3.2.3. Transformations and Mappings

A key point of the MDE is the transformation between models. The transformations allow to go from one model at a given abstraction level to another model at another level, and to keep the different models synchronized. Related models are described by their metamodels, on which we can define some mapping rules describing how concepts from one metamodel are to be mapped on the concepts of the other metamodel. From these mapping rules we deduce the transformations between any models conforming to the metamodels.

3.2.4. Use of Standards

The MDE is based on proved standards: UML for modeling and the EMOF (Essential Meta Object Facilities [87]) for metamodel expression. The recent UML 2 [47] standard is specifically designed to be used with the MDE. It removes some ambiguities found in its predecessors (UML 1.x), allows more precise descriptions and opens the road to automatic exploitation of models. The EMOF is oriented to the metamodel specifications.

3.2.5. System-on-Chip Design

SoC (System-on-Chip) can be considered as a particular case of embedded systems. SoC design covers a lot of different viewpoints including the application modeling by the aggregation of functional components, the assembly of existing physical components, the verification and the simulation of the modeled system, and the synthesis of a complete end-product integrated into a single chip. As a rule, a SoC includes programmable processors, memory units (data/instructions), interconnection mechanisms and hardware functional units (Digital Signal Processors, application specific circuits). These components can be generated for a particular application; they can also be obtained from IP (Intellectual Property) providers. The ability to re-use software or hardware components is without any doubt a major asset for a codesign system.

The multiplicity of the abstraction levels is appropriate to the modeling approach. The information is used with a different viewpoint for each abstraction level. This information is defined only once in a single model. The links or transformation rules between the abstraction levels permit the re-use of the concepts for a different purpose.

3.2.6. Contributions of the team

Our proposal is partially based upon the concepts of the "Y-chart" [71]. The MDE contributes to express the model transformations which correspond to successive refinements between the abstraction levels.

Metamodeling brings a set of tools which will enable us to specify our application and hardware architecture models using UML tools, to reuse functional and physical IPs, to ensure refinements between abstraction levels via mapping rules, to ensure interoperability between the different abstraction levels used in a same codesign, and to ensure the opening to other tools, like verification tools, thought the use of standards.

The application and hardware architecture are described by different metamodels. Some concepts from these two metamodels are similar in order to unify and so simplify their understanding and use. Models for application and hardware architecture may be done separately (maybe by two different people). At this point, it becomes possible to map the application model on the hardware architecture model. A third metamodel, called association metamodel, allows us to express this mapping.

All the previously defined models, application, architecture and association, are platform independent. No component is associated with an execution, simulation or synthesis technology. Such an association targets a given technology (Fortran, SystemC RTL, SystemC CABA, VHDL, etc). Once all the components are associated with some technology, the deployment is realized.

This is done by the refinement of the association model to the deployed model first, and then to further abstraction levels (e.g. polyhedron, loop in figure 1).

The diversity of the technologies requires interoperability between abstraction levels and simulation and execution languages. For this purpose we define an interoperability metamodel allowing to model interfaces between different abstraction levels.

Mapping rules between the deployment metamodel, and interoperability and technology metamodels can be defined to automatically specialize the deployment model to the chosen abstraction levels. From each of the resulting models we could automatically produce the execution/simulation code and the interoperability infrastructure.

The simulation results can lead to a refinement of the application, the hardware architecture, the association or the deployment models. We propose a methodology to work with these models. The stages of design could be:

1. Separation of application and hardware architecture modeling.
2. Association with semi-automatic mapping and scheduling.
3. Selection of IP for each element of application/architecture models.
4. Deployment (choice of simulation or execution level and platform for each component).
5. Automatic generation of the various platform specific simulation or execution models.
6. Automatic simulation or execution code generation.
7. Refinement at the highest level given the simulation results.

3.2.6.1. Gaspard2 foundations

The abstract syntax of application and hardware architecture are described by different EMOF metamodels. Some concepts from these two meta-models are similar, in order to simplify their understanding and use.

They share a common modelling paradigm, the component oriented approach, to ease reusability. Reusability is one of the key point to face the time to market challenge that the conception of embedded systems implies.

The two meta-models also share common construction mechanisms, to express repetitive constructs in a compact way. This kind of compact expression makes them more comprehensible for a compiler or an optimisation tool.

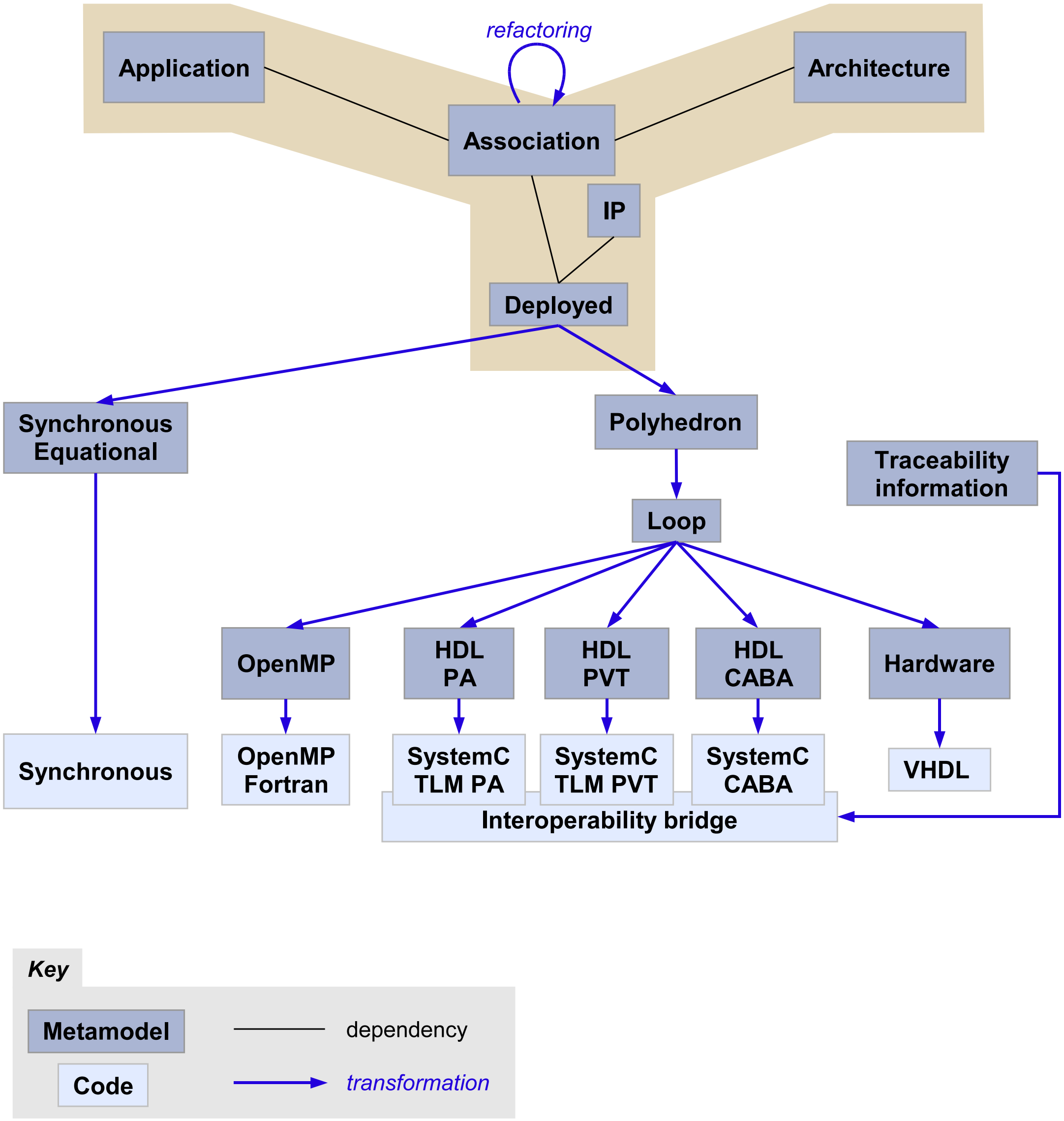

Figure 1. Overview of the metamodels for the "Y" design

To express the mapping of an application model on a hardware architecture model, a third meta-model named association is introduced. This meta-model imports the concepts of the two previously mentioned meta-models.

The definition of the Gaspard profile (a detailed specification can be found in [63]) is oriented by the few guiding ideas mentioned in section 2. It offers a high level modeling environment for high performance systems on chip, and includes UML extensions (formally the UML view point of the profile) as well as an abstract syntax expressed in EMOF (Domain view point). The main evolutions according to past versions of the profile are:

- a better structuring of the various packages, in order to define as much common parts as possible for the various aspects of our Y approach
- a better alignment with related OMG standards (SPT, QoS, SysML)[2].

The architecture of the Gaspard profile is now structured around five packages (fig.2):

- **component package**: Extensively reuse mechanisms defined in UML 2 [49](composition, assembling), with some minor but interesting extensions (capability to parametrize components/IPs, which is a common practice for IP providers). The main goal of this package is to ease and encourage reuse of components. As illustrated by the `import` dependencies, application and hardware architecture packages share a common definition of the Component concept
- **factorization package** [6]: Introduces mechanisms enabling to express in a compact way a repetition of structural elements and the regular topology of the links between them. As illustrated by the `import` dependencies, the component package, and as a consequence the application and hardware architecture and association packages, use these common mechanisms. This proposal is partially inspired by the Array-OL language [65]
- **hardware architecture package** [64]: Enables to dimension a hardware architecture and the resources that compose it, and to describe the topology of their interconnections. The structure of this proposal is inspired by the SPT profile [46].
- **application package**: Defines a simple but powerful design pattern for modelling of application only via data dependencies expression.
- **association package** [59]: Offers mechanisms to take into account two aspects of association, i.e characterization (put into action via the QoS profile [50]) and allocation (via the Allocation concept of SysML [51]). The concepts introduced enable to express mapping directives for allocation of data and tasks of an application onto an hardware architecture, which is itself oriented by the characterizations.
- **control package** [31]: Represents the basic elements used to express an automaton structure and the different running modes of a system. It is one of the recent packages that have been added to the Gaspard profile.

3.2.6.2. Application Metamodel

The application metamodel focuses on the description of data dependencies between components. Components and dependencies completely describe an algorithm without addition of any parasitic information. Actually any compilation optimization or parallelization technique must respect the data dependencies. This gives many benefits:

- simple description of the algorithm,
- no dependency analysis in the compiler,
- all the parallelism and optimization potential of the algorithm is easily available.

[2] In [63], we also show why UML for SoC [52] is not relevant for our goals

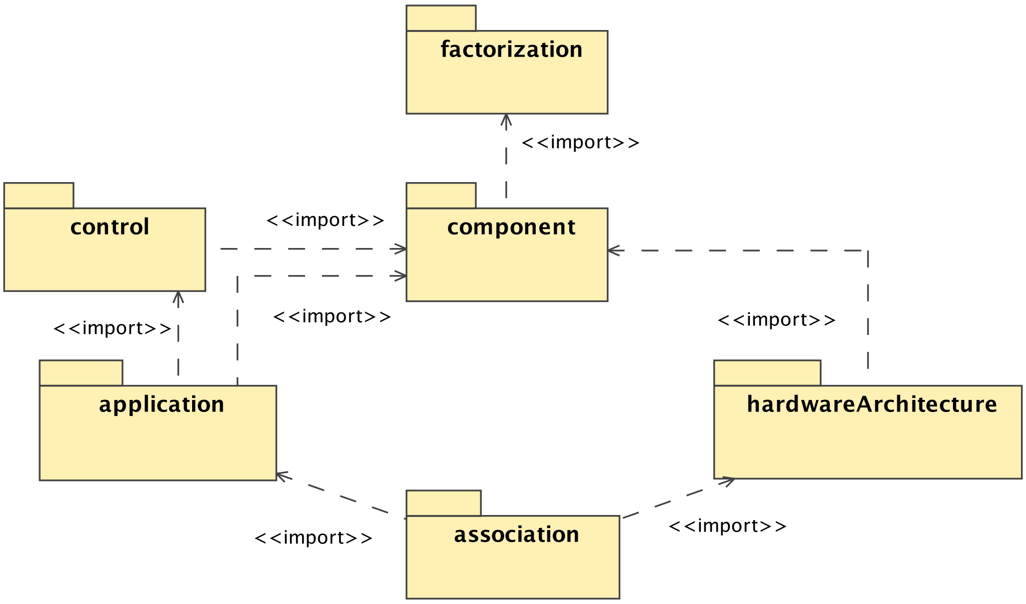

Figure 2. Gaspard Profile Architecture

Application components represent some computation and their ports some data input and output capabilities. Data handled in the applications are mainly multidimensional arrays, with one possible infinite dimension representing time.

The application meta-model introduces three kinds of components : Compound, DataParallel, and ElementaryComponents.

A compound component expresses task parallelism by the way of a component graph. The edges of this graph are directed and represent data dependencies.

A data parallel component expresses data parallelism by the way of the parallel repetition of an inner component part on patterns of the input arrays, producing patterns of the output arrays. Some rules must be respected to describe this repetition. In particular, the output patterns must tile exactly the output arrays.

Elementary components are the basic computation units of the application. They have to be defined for each target technology.

Data parallelism expression is one of the key point of our approach. In domains such as intensive signal processing or telecommunications (typically targeted by embedded systems), applications generally present lot of potential data parallelism.

In order to broaden the application domain of our metamodel, we have also studied a design methodology for synchronous reactive systems, based on a clear separation between control and data flow parts. This methodology allows to facilitate the specification of different kinds of systems and to have a best readability. It also permits to separate the study of the different parts by using the most appropriated existing tools for each of them. Following this idea, we are particularly interested in the notion of running modes and in the Scade tool. Scade is a graphical development environment coupling data processing and state machines (modeled by synchronous languages Lustre and Esterel). It can be used to specify, simulate, verify and generate C code. However, this tool does not follow any design methodology, which often makes difficult the understanding and the re-use of existing applications. We show that it is also difficult to separate control and data parts using

Scade. Thus, regulation systems are better specified using mode-automata which allow adding an automaton structure to data flow specifications written in Lustre. When we observe the mode-structure of the mode-automaton, we clearly see where the modes differ and the conditions for changing modes. This makes it possible to better understand the behavior of the system.

3.2.6.3. Hardware Architecture Metamodel

The purpose of this meta-model is to satisfy the growing need of embedded system designers to specify the hardware architecture of the system at a high abstraction level. It enables to dimension the resources of the hardware in a precise enough way to be pertinent, but abstracting irrelevant details so that efficient decision could be taken.

A mechanism similar to the one used in the application meta-model enables to specify repetitive architecture in a compact way. We believe that regular parallel computation units will be more and more present in embedded in systems in the future, especially for Systems on Chips. This belief is driven by two considerations:

1. Time-to-market constraints are becoming so tight that massive reuse of computation units is one of the only ways to get the computation power needed for next generation embedded applications.
2. Parallelism is a good way to reduce power consumption in SoCs. Indeed at equal computing power, a chip able to run several computations simultaneously is clocked at a lower frequency than a chip able to run less computations in a given cycle. As frequency is square in the power consumption equation, this leads to important gains.

The repetitive constructs we propose can be used to model parallel computation units, such as grids, but also complex static or dynamic interconnection networks, or memory banks.

3.2.6.4. Association Metamodel

The association metamodel allows to express how the application is projected and scheduled on the architecture. This metamodel imports the application and architecture metamodels in order to associate their components. The association model associates application components with architecture components to express which hardware component executes which functionality. If the hardware component is programmable, the application components associated with this component are implemented in software, otherwise, they are synthesized as hardware. The dependencies between application components are associated with communication routes. These routes are built as sequences of data paths, components and represent the route of data from one memory to another via processor or DMA initiated data exchanges. The input and output of the functional components are mapped into memories.

As the application and hardware architecture models, the association model takes advantage of a repetitive and hierarchical representation to allow to view the association at different granularities and to factorize its representation.

The association model is the input and the output of the optimization algorithm. Indeed, the optimization can be seen as a refactoring of the association model. Code transformations allow to refactor the application to map it more easily on the target hardware architecture. Once refactored, each application component can be mapped to a corresponding hardware architecture component. The repetitive application components can be distributed to a repetitive hardware architecture component. The association metamodel allows at least the regular mapping strategies of classical data-parallel languages as High Performance Fortran.

3.2.6.5. Transformation Techniques

Model to model transformations are at the heart of the MDE approach. Anyone wishing to use MDE in its projects is sooner or later facing the question: how to perform the model transformations? The standardization process of Query View Transformation [86] was the opportunity for the development of transformation engine as ModTransf. However, since the standard has been published, only few of investigating tools, such as ATL [3] (a transformation dedicated tool) or Kermeta [4] (a generalist tool with facilities to manipulate models) remained compliant with the standard and are powerful enough to fulfil execution needs that we had identified.

[3] http://www.eclipse.org/gmt/atl

[4] http://www.kermeta.org

The existence of efficient transformation tools as well as the difficulties encountered in the design of the Gaspard supply chain lead us to focus on another problem: document transformations. Indeed, transformations are widely encountered in embedded system product chains; each metamodel of the chain involves different and complex concerns; and the exchange on the design of transformation in a co-design context becomes a real need. We, thus, put the next question under the spot-light: how to design and document model transformations?

Thanks to our expertise about implementation of previous research result, ModTransf, and our know-how about model design with UML like notations, we are working on a graphical representation of the transformation to increase documentation, exchange around transformations. We have identified the following requirements to design efficiently a model transformation:

- graphical representation of model transformations;
- portable notation independent from any existing transformation engines;
- mechanisms to divide transformation into rules;
- focus on what should be transformed, rather than how to implement the transformation;
- black boxes mechanism to enable legacy code call;
- input and output metamodel extension to access complex construct in easier way or to define your own methods or attributes;
- precise definition and use of concepts, thanks to a profile and a metamodel;
- free way to implement the transformation, execution on top of ATL, Kermeta, QVT... are possible;
- standard way to specify the rules, facilitating exchange, comprehension, and documentation.

TrML (Transformation Modelling Language) profile is the proposed solution that fulfils all these needs. It is a rule based notation specifying one or more models as inputs and one or more models as outputs. The rules can be expressed using a UML based well-known notation. TrML provides a framework that guides the transformation design by dividing them in more simple units. We are now working on the definition of an algorithm to process all the pieces of the transformation designed with TrML that has to be implemented in the transformation engine chosen by the user.

3.3. Optimization Techniques

Keywords: *Compilation*, *Dataparallelism*, *Heuristics*, *Mapping*, *Optimization*, *Power Consumption*, *Scheduling*.

3.3.1. Optimization for parallelism

We study optimization techniques to produce a schedule and a mapping of a given application onto a hardware SoC architecture. These heuristic techniques aim at fulfilling the requirements of the application, whether they be real time, memory usage or power consumption constraints. These techniques are thus multi-objective and target heterogeneous architectures.

We aim at taking advantage of the parallelism (both data-parallelism and task parallelism) expressed in the application models in order to build efficient heuristics.

Our application model has some good properties that can be exploited by the compiler: it expresses all the potential parallelism of the application, it is an expression of data dependencies –so no dependence analysis is needed–, it is in a single assignment form and unifies the temporal and spatial dimensions of the arrays. This gives to the optimizing compiler all the information it needs and in a readily usable form.

3.3.2. Contributions of the team

We focus on two particular subjects in the optimization field: dataparallelism efficient utilization and multi-objective hierarchical heuristics.

3.3.2.1. Dataparallel Code Transformations

We have studied Array-OL to Array-OL code transformations [60], [95], [68], [67], [69] [44]. Array-OL [65], [66] is a dataparallel language dedicated to systematic signal processing. It allows a powerful expression of the data access patterns in such applications and a complete parallelism expression. It is at the heart of our model of application, hardware architecture and association.

The code transformations that have been proposed are related to loop fusion, loop distribution or tiling but they take into account the particularities of the application domain such as the presence of modulo operators to deal with cyclic frequency domains or cyclic space dimensions (as hydrophones around a submarine for example).

We have studied the relations of the Array-OL model with other computation models [54] such as Kahn Process Networks [76], [77] and multidimensional synchronous dataflow [84], [83].

We pursue the study of such transformations with three objectives:

- Propose utilization strategies of such transformations in order to optimize some criteria such as memory usage, minimization of redundant computations or adaptation to a target hardware architecture.
- Stretch their application domain to our more general application model (instead of just Array-OL).
- Try to link the Array-OL code transformations and the polyhedral model in order to cross fertilize the two domains.

3.3.2.2. Multi-objective Hierarchical Scheduling Heuristics

When dealing with complex heterogeneous hardware architectures, the scheduling heuristics usually take a task dependence graph as input. It is the case in the AAA methodology [94], [93], [72] that is implemented in the SynDEx [92] tool. Both our application and hardware architecture models are hierarchical and allow repetitive expressions. We propose a Globally Irregular, Locally Regular combination of heuristics to allow to take advantage of both task and data parallelism [79].

Furthermore, local optimizations (contained inside a hierarchical level) decrease the communication overhead and allow a more efficient usage of the memory hierarchy. We aim at integrating the dataparallel code transformations presented before in the GILR heuristic in order to deal efficiently with the dataparallelism of the application by using repetitive parts of the hardware architecture.

In embedded systems, minimizing the latency of the application is usually not the good objective function. Indeed, one must reach some real time constraints but it is not useful to run faster than these constraints. It would be more interesting to improve the resource usage to decrease the power consumption or the cost of the hardware architecture. Various techniques exist to reduce power consumption in embedded systems. This research covers:

- The evaluation of the impact of cache management schemas on power consumption [85], [90].
- The study of code compression techniques to reduce the memory requirements of an embedded application [75].
- Clock scaling to choose the slowest speed that satisfies the real-time constraints.

3.4. SoC Simulation

Keywords: *SystemC*, *TLM*.

Many simulations at different levels of abstraction are the key of an efficient design of embedded systems. The different levels include a functional (and possibly distributed) validation of the application, a functional validation of the application and an architecture co-model, and a validation of a heterogeneous specification of an embedded system (a specification integrating modules provided at different abstraction levels).

SoCs are more and more complex and integrate software parts as well as specific hardware parts (IPs, Intellectual Properties). Generally before obtaining a SoC on silicium, a system is specified at several abstraction levels. Any system design flow consists in refining, more or less automatically, each model to obtain another, starting from a functional model to reach a Register Transfer Level model. One of the biggest design challenge is the development of a strong, low cost and fast simulation tool for system verification and simulation.

The DaRT project is concerned by the simulation at different levels of abstraction of the application/architecture co-model and of the mapping/schedule produced by the optimization phase.

3.4.1. Abstraction levels

Design flow systems allow the description of system modules (IPs) mainly at six levels of abstraction (this is the case in the standardization effort of SystemC[5]:

Communicating Processes (CP): a model, at this level, is similar to an executable specification without any information about the hardware architecture. In fact, the system is composed by functions (processes) exchanging parameters. The communication between modules is point-to-point, and usually modelled using abstract channels.

Communicating Processes with Time (CP + T): it is similar to CP but timing delays are added to processes within the design to reflect the timing constraints of the specification.

Programmer's View (PV): at this level, hardware IPs composing the system's architecture appears in the model. Shared communication links should be modelled at PV level, but both behaviour and communication still untimed.

Programmer's View with Time (PV + T): it is similar to PV but timing delays are added to processes within the design to reflect the timing constraints of the specification and also to process delays of the target architecture. Processors, at this level, should be modelled using Instruction Set Processors (ISS).

Cycle Accurate, Bit Accurate (CABA): the internal structure accurately reflects the registers and the combinatorial logic of the target architecture. The communications are described in details in terms of used protocols and timing. Pins appear as in the physical components, and each module's behaviour corresponds exactly to the behaviour of the physical module, locally at each cycle.

Register Transfer Level (RTL): RTL models are very close to CABA ones, in terms of accuracy. The most important differences with CABA level are:

- In RTL the model is accurate both locally (in each cycle) and globally
- In RTL the description is synthesizable.

The five first levels are commonly called TLM levels, and the two first ones functional levels.

3.4.2. Contribution of the team

The results of DaRT simulation package concerns mainly the functional level and the TLM level. We also propose techniques to interact with IPs specified at other level of abstraction (mainly RTL).

At the functional level: we have developed a Distributed Kahn Process Network environment. The result of this simulation guarantees the functionality of the application model. By the observation of the FIFO sizes, we are able to transform the application to improve the load balance of the system. The distributed aspect of this simulator permits to associate IPs from different builders available on different websites.

At TLM level: From the association model of our "Y-model", we are able to simulate the application and the architecture of the SoC in the same time. The results expected from this simulation cover the schedule of elementary tasks, the mapping of the data parallel structure on hierarchical and parallel memories, and the communications involved by this mapping. At this level, our models are still independent from any platform detail.

[5] http://www.systemc.org

At RT level: In order to get physical implementations of our applications, we are developing an RTL metamodel. Models at this level will be obtained by transformations from those represented at TLM.

At SystemC level: we propose some generic wrappers to allow multilevel abstraction interoperability. A special effort was done to support distributed and heterogeneous simulation framework (see figure 3).

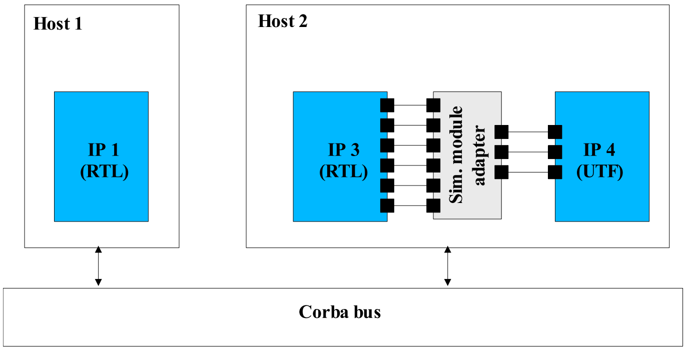

Figure 3. Distributed SystemC Simulation

3.4.2.1. Co-simulation in SystemC

From the association model, the Gaspard environment is able to automatically produce SystemC simulation code. The MDE techniques offer the transformation of the association model to the SystemC Gaspard model. During this transformation the data parallel components are unrolled and the data dependencies between elementary tasks become synchronisation primitive calls.

The SoC architecture is directly produced from the architecture model. A module in SystemC simulates the behaviour of tasks mapped to a particular processor. Other modules contain the data parallel structures and are able to answer to any read/write requests. The communications between tasks and between tasks and memories are simulated via communication modules in SystemC. These last modules produce interesting results concerning the simultaneous network conflicts and the capacity of this network for this application. The Loop metamodel allows automatic SystemC code generation. The association model is first transformed into a model of this Loop metamodel and this model is then automatically transformed into SystemC code. This development is integrated in the Gaspard prototype and uses the MoMoTE tool (see the software section).

3.4.2.2. Multilevel distributed simulation in SystemC

A multilevel simulation model is an executable specification containing a set of modules described at different abstraction level (e.g an functional IP coupled with an RTL IP). Our contribution is the proposal of a new methodology to validate SoCs by simulation [81]. With this new approach, we can perform a fast and low cost simulation of an assembly of IPs. At the opposite of existing solutions, we do not impose the usage of external libraries. Our solution is based on an internal SystemC library and a rule description language. We generate a simulation module adapter to encapsulate one of the two interconnected modules.

In the same idea of IP integration, we develop a distributed runtime for SystemC using sockets or Corba [82]. With this first implementation of a distributed SystemC, it is now possible to create a SoC with IPs selected from different providers.

Both the multilevel of abstraction runtime and the distributed runtime offer to SystemC the possibility to support a real co-design from world distributed IP providers.

3.4.2.3. TLM: Transactional Level Modelling

Transactional Level modelling (TLM) appeared during the very few last years. It consists in describing systems according to the specifications of the TLM abstraction levels (briefly mentioned in Section 3.4.1). At these levels, communications use function calls (e.g. burst_read(char* buf, int addr, int len);). The major aims of TLM modelling are:

- Enable fast simulations and compact specifications
- Integrate HW and SW models
- Early platform for SW development
- Early system exploration and verification
- IPs reuse

Nowadays, this modelling style is widely used for verification and it is starting to be used for design at many major electronic companies. Recently, many actions and challenges have been started in order to help to proliferate TLM. Thus, several teams are working to furnish to designers standard TLM APIs and guidelines, TLM platform IP and tools supports. SystemC is the first system description language adopting TLM specifications. Thus, several standardization APIs have been proposed to the OSCI by all the major EDA and IP vendors. This standardization effort is being generalized now by the OSCI / OCP-IP TLM standardization alliance, to build on a common TLM API foundation. One of the most important TLM API proposals is the one from Cadence, distributed to OSCI and OCP-IP. It is intended as common foundation for OSCI and OCP-IP allowing protocol-specific APIs (e.g. AMBA, OCP) and describing a wide range of abstraction levels for fast and efficient simulations.

Due to all TLM's benefits, we defined a TLM metamodel as a top level point for automatic transformations to both simulation and synthesis platforms. Our TLM metamodel contains the main concepts needed for verification and design following the Cadence API proposal. But, as we are targeting multi-language simulation platforms, the meta model is completely independent from the SystemC syntax. It is composed mainly by two parts: architecture and application. This clear separation between SW and HW parts permits easy extensions and updates of the meta model.

The architecture part contains all necessary concepts to describe HW elements of systems at TLM levels. The SW part is mainly composed of computation tasks. They should be hierarchical and repetitive. A set of parameters could be attached to each task in order to specify the scheduling dependently of the used computation model. Thus this metamodel keeps hierarchies and repetitions of both the application and the architecture. This permits to still benefit from the data parallelism as far as possible in the design (simulation and synthesis flow). In fact, the designer can choose to eliminate hierarchies when transforming the TLM model into a simulation model, and to keep it when transforming into a synthesis model.

3.4.2.4. Network on Chip (NoC) design and performances estimation

Modern SoCs are very complex and integrate more and more heterogeneous IPs. Due to this complexity, designers need high performance interconnection components. These latter have to be also, as much as possible, flexible to support new applications. This kind of interconnection IPs is unfortunately not available until today. In fact, designers still use buses and simple point-point connections in their designs.

Our contribution in this domain is the proposal of an open Network on Chip library for SoCs design. The NoCs will be mainly an adaptation, for embedded systems, of those proposed for classical multiprocessor architectures. Performances of these networks have been proved, and we believe that such a library will permit the integration of more and more IPs on a chip in a systematic way. This library will be also a support and a completion of existing open SystemC IP libraries as SoCLib. All its components will be OCP compliant [80].

4. Application Domains

4.1. Intensive Signal Processing

Keywords: *multimedia*, *telecommunications*.

The DaRT project aims to improve the design of embedded systems with a strong focus on intensive signal processing applications.

This application domain is the most intensive part of signal processing, composed of:

- systematic signal processing;
- intensive data processing.

Many signal and image processing applications follow this organisation: software radio receiver, sonar beam forming, or JPEG 2000 encoder/decoder.

In the framework of the ModEasy project, we also study computation intensive automotive safety embedded systems.

The systematic signal processing is the very first part of a signal processing application. It mainly consists of a chain of filters and regular processing applied on the input signals independently of the signal values. It results in a characterization of the input signals with values of interest.

The intensive data processing is the second part a of a signal processing application. It applies irregular computations on the values issued by the systematic signal processing. Those computations may depend on the signal values.

Below are three application examples from our industrial partners.

Software Radio Receiver This emerging application is structured in a front end systematic signal processing including signal digitalizing, channel selection, and application of filters to eliminate interferences. These first data are decoded in a second and more irregular phase (synchronization, signal demodulation...).

Sonar Beam Forming A classical sonar chain consists in a first and systematic step followed by a more general data processing. The first step provides frequency and location correlations (so called *beam*) from a continuous flow of data delivered by the hydrophones (microphones disposed around a submarine). It is based on signal elementary transformations: FFT (Fast Fourrier Transformation) and discrete integration. The second step analyses a given set of beams and their history to identify temporal correlation and association to signal sources.

JPEG-2000 Encoder/Decoder JPEG-2000 is a new standard format for image compression. The encoder works in a two-steps approach [53]. The first part (from preprocessing to wavelet decomposition) is systematic. The second part of the encoder includes irregular processing (quantification, two coding stages). The decoder works the other way around: a first irregular phase is followed by a systematic phase.

4.2. Automotive Safety Embedded Systems

The automotive industry has specific problems, particularly due to increased safety requirements and legal framework. The automobile is a hostile environment: especially in the engine compartment. Some failure modes will be benign, whereas others may be dangerous and cause accidents and endanger human life. The Annex to the IEE Guidance Document on EMC and Functional Safety [ref] enumerates 21 electronic systems that may be present in the modern automobile. Some of these electronic systems have the potential to endanger the safety of vehicle occupants or other road users should an error or a mis-operation occur.

In the ModEasy Interreg project we want to model a cruise control connected to the satellite positioning system, GPS. From a UML specification and using classical verification and model checking techniques, we want to assure the correct behaviour of the system. Using model transformation allows the guarantee of these verifications at the lower levels like SystemC/VHDL.

Collision avoidance radars are now integrated into high end models by car manufacturers. The current devices are however based on the frequency modulation and their maximum range is limited if the emitted power is kept under the recommended values The receiver uses digital correlators which have been implemented via DSP microprocessors. The codes are generated using FPGA devices. In order to achieve greater integration and improve security, we are now seeking to design the major parts as embedded systems based on FPGA and SoC devices. In this context, the use of tools developed in the ModEasy project will improve and facilitate the design of such complex systems. Moreover, as ModEasy is based on metamodels and transformations between metamodels, new algorithms or new FPGAs can rapidly be integrated in the system by the re-use of existing functional blocks.

5. Software

5.1. MoMoTE

Keywords: *MDA*, *MDE*, *Model Transformation*, *QVT*, *Query View Transformation*.

Participants: Antoine Honoré [contact person], Cédric Dumoulin, Anne Etien, Emmanuel Renaux.

MoMoTE is a Java Framework that allows us to perform models to model transformations. It is composed of an API and an engine. It takes input models that conform to some metamodels and produces output models that conform to other metamodels. A transformation with MoMote is made of rules. A rule describes a subset of the transformation and can call subrules.

The API is provided in an Eclipse plugin. It works with Ecore-based metamodels. Developers that use MoMoTE extend the rule class to make their own set of rules. Each rule must implement a selection on input models. This is done with the EMFT Query API (an Eclipse plugin to query models based on the OMG QVT standard). For each result of the selection, the rule must define elements to be created. Finally the rules are assembled in a hierarchical tree, passed to the transformation engine.

The MoMoTE transformation engine executes rules recursively, creating output element models first and solving references in a second step. Some of the rules are stamped as root, and are executed first. To perform the transformation, users have to provide input models. The engine will create the output models according to the rules.

MoMoTE is a technical representation of the TrML specification introduced by DaRT. It is used in the Gaspard2 software and distributed with it. It contributes to the transformation chain of Gaspard2. However, it can be used outside Gaspard2 as a standalone transformation engine.

5.2. MoCodE

Keywords: *Code generation*, *MDA*, *MDE*, *language*, *model*.

Participant: Antoine Honoré [contact person].

MoCodE (Models to Code Engine) is an API that enables to perform model to text transformation. It takes a set of input models conform to several metamodels and a set of Java classes. Then, it produces text files.

MoCodE gives an engine that, recursively, gets each element of input models and executes a corresponding Java code. The Java code produces text files that are code written in general in other languages. Developers are in charge of the produced text and the division of the output files structure. At this time, DaRT uses it to produce FORTRAN and SystemC code from models conform to the Gaspard2 metamodel.

The MoCodE API is an Eclipse plugin based on the Eclipse Modeling Framework (EMF) plugin. This is a part of the Gaspard2 software and processes last operations of its transformation chain. However, it can be used like an autonomous plugin in Eclipse.

5.3. Gaspard2

Keywords: *Eclipse*, *IDE*, *SoC Design*, *Visual Design*.

Participants: Pierre Boulet [contact person], DaRT project members.

Gaspard2 is an Integrated Development Environment (IDE) for SoC visual co-modeling. It allows or will allow modeling, simulation, testing and code generation of SoC applications and hardware architectures.

Gaspard2 Its purpose is to provide one single environment for all the SoC development processes:

- High level modeling of applications and hardware architectures
- Application and hardware architecture association
- Application refactoring
- Deployment specification
- Model to model transformation (to automatically produce models for target platforms)
- Code generation
- Simulation
- Reification of any stages of the development

The Gaspard2 tool is based on Eclipse [62]. A set of plugins provides the different functionalities. Application, hardware architecture, association, deployment and technology models are specified and manipulated by the developer through UML diagrams, and saved by the UML tool in an XMI file format. Gaspard2 manipulates these models through repositories (Java interfaces and implementations) automatically generated thanks to the JMI standard.

6. New Results

6.1. Model-driven engineering for Soc design

Participants: Rabie Ben Atitallah, Lossan Bonde, Pierre Boulet, Arnaud Cuccuru, Jean-Luc Dekeyser, Cédric Dumoulin, Abdoulaye Gamatié, Calin Glitia, Frédéric Guyomarc'h, Antoine Honoré, Cyril Joly, Souha Kamoun, Ali Koudri, Ouassila Labbani, Sébastien Le Beux, Philippe Marquet, Éric Piel, Imran Rafiq Quadri, Éric Rutten, Safouan Taha, Julien Taillard, Huafeng Yu.

A lot of efforts have been made by the DaRT project on the MARTE standardization effort and on the software development [16]. These efforts contributed to the improvement of our SoC modeling framework, Gaspard. In the next sections, we highlight the new aspects about the metamodels used in this framework.

6.1.1. General improvements on Gaspard profile and metamodel

The Gaspard profile has evolved in 2006 along several directions. Along with the progression of the definition of the MARTE profile, the Gaspard profile has been streamlined and made more abstract. In addition to its simplification, we have refined the mechanism of association. The new concepts permit to allocate either task, data, or communication on one or several components of the hardware architecture.

Advances have been made on the use of the UML template mechanism in conjunction with the repetitive structure modeling capabilities of Gaspard to model multistage interconnection networks. We have shown how to model the delta family of interconnection networks [45]. This case study shows how to model recursively defined repetitive structures.

A complete documentation of the profile has been written, which documents each stereotype (concept) and the syntax to draw them. Furthermore, the semantics of assembled stereotypes is described. Several examples either making use of all the various concepts of the Gaspard profile, or stressing one particular aspect of the profile have been written.

The notion of *Deployment Specification* has also been introduced. In order to generate compilable code from the SoC model, it is necessary for the designer to specify which IP (or function) will be used to implement an elementary component. The Deployment Specification package proposes concepts which (i) allows to describe the relation between a Gaspard representation of an Elementary component (a box with ports) to a text-based code (a function with arguments) and (ii) allows to inform the Gaspard transformations of specific behaviours of each component (such as average execution time, power consumption...) in order to generate a high abstraction level simulation in adequacy with the real system.

Finally, the Gaspard profile has been extended so as to support control modeling [31], [12] and to allow the representation of low-level concepts that are close to the code generation step (see figure 1). These extensions are presented in the next sections.

6.1.2. A metamodel for the specification of control

The Gaspard framework is particularly adapted for the design of embedded systems that compute large amounts of data in a regular fashion. However, it does not allow to deal with aspects such as temporal constraints imposed by system environment, or control in computations according to different modes or configurations. So, we propose an extension of Gaspard taking into account such aspects by considering mode automata and synchronous equations, such as to bridge the gap between data-intensive computation models and the synchronous models and tools.

6.1.2.1. Control in Gaspard.

The introduction of control in the Gaspard application part is mainly based on the synchronous approach and the control/data flow separation methodology [32]. The introduction of the control into data parallelism applications requires the definition of a *degree of granularity* for these applications [12]. This concept allows to delimit the different execution cycles or *clock signals* in which it becomes possible to take the control values into account. To do that, we introduce new concepts to specify control in Gaspard application models. These concepts are gathered in the `control` package [31].

The `control` package represents the basic elements used to express an automaton structure and the different running modes of the studied system. The question which arises now is: *how to model the control automaton and the different running modes in the* GASPARD2 *UML profile?*

The control part represents an automaton structure based on the mode-automata concept. It allows to clearly specify the various running modes of the system and the switching conditions between modes. To introduce control concepts in the Gaspard application models, we must take the Gaspard model semantics into account. These semantics does not express any flow concept since the time is represented by a no ordered and infinite dimension. However, in the control automaton structure, the flow concept is present and significant for the definition of its behavior. For this reason, it becomes important to propose a model representation of the automaton structure by respecting as much as possible the Gaspard model semantics. A possible solution consists in using a dependency relation between the different repetitions of the automaton transition function. This dependency makes it possible to memorize the previous state of the automaton and then to respect the general automaton structure semantics.

The controlled part represents a special component having a *switch* behavior. To model this part in the Gaspard application profile, we introduce new concepts allowing to model the different running modes of the application. At each calculation time, one and only one running mode is activated according to the information provided by the control part. This information can be the name of the mode to be activated or any other index allowing to distinguish the modes in a clear and single way.

6.1.2.2. Synchronous modeling of Gaspard concepts.

Another way to specify control aspects in Gaspard consists in using clocks as defined in synchronous languages. So, we proposed a synchronous equational model of Gaspard concepts [38] [39]. The target languages are LUCID SYNCHRONE, LUSTRE, and SIGNAL. The resulting equational models are semantically equivalent to considered Gaspard models and enable code generation from these target languages. The implementation of our models is being integrated in the general MDE-based framework of Gaspard. It takes the form of a transformation between Gaspard models and a simple metamodel of synchronous equations. The MDE approach is expected to provide us with facilities in the evolution of our transformations.

Perspectives are being explored towards synchronous modeling of hierarchical applications, and adding control automata involving verification, following the work presented above.

6.1.3. A metamodel based on polyhedra

In order to simplify and factorize the work on model transformations, a metamodel has been defined which is a common denominator of several of the outputs metamodels. Consequently, this metamodel permits representation of a Gaspard model, with no "loss" of information for the model implementations, while being at a lower level of abstraction. The goal of the metamodel is to make the notion of *association* implicit. Instead, the application part of the model is dispatched over the processors (for the tasks) and the memories (for the data). In order to represent the information of which repetition of a task is on which instance of a processor, expressed by the user using the association, the metamodel uses the concepts of polyhedra parametrized by the processor indices. Polyhedron representation has been selected for, at the same time, being able to represent every possible association written by the user and being easily transformed into code.

6.1.4. A metamodel based on loops

This metamodel is used after the one based on polyhedra. It is the last common metamodel before targeting implementation platforms. This metamodel is closed to the polyhedron one. It is used to refine model toward code generation. Polyhedra expressions are transformed into loops expressions using CLooG (Chunky Loop Generator) [56]. CLooG is a tool that generates loops for scanning Z-polyhedra. For each polyhedron, CLooG is called by transformation rules and polyhedra information are replaced by LoopStatements in the loop metamodel. Loops, which are parametrized by the processor indices, indicate which repetition of a task is executed by which processor. So each processor has the same code. Then, we can target SoC simulation or high performance computing.

6.1.5. A metamodel for procedural language with OpenMP statement

We focused on shared memory computers in order to first simplify the transformations. So we do not have to manage communication between processors. This metamodel is used before code generation for high performance computing. It permits the representation of a code written with procedural language (Fortran and C) with OpenMP statement [89]. So, we have a representation close to the code, which can be used to target different languages. Finally, code generation is a just "pretty printer" for this metamodel.

6.1.6. A metamodel for FPGA implementation

In the frame of the ModEasy project, we have implemented a new anti-collision radar algorithm on FPGA that should permit the detection of an obstacle up to 150 meters far away [33], [42]. This implementation done by hand permits to describe a metamodel in which all manipulating concepts will be concretely identified and expressed. The detection is based on higher order statistics and requires much more hardware resources than state-of-the-art algorithms based on correlation. The anti-collision radar algorithm has been modeled by using the Gaspard metamodel and has been successfully coupled with a cruise control with GPS, following the control and data flow separation methodology [34]. The produced VHDL has been synthesized onto a Stratix 2 FPGA.

In order to automate the above transformation from the Gaspard model to VHDL, we have proposed a metamodel. This metamodel enables the description of a hardware accelerator, and is definitively oriented toward expression of parallelism. All the concepts necessary for the realization of a hardware accelerator, such as the anti-collision radar, appear in this metamodel. We have shown that it is possible to model the anti-collision radar within the hardware accelerator metamodel.

Transformation rules allow to transform a Gaspard application into the proposed hardware accelerator metamodel. The first produced rules do not take into account the FPGA characteristics and may provide an electronic design not dimensioned for the FPGA. We are currently working on the definition of rules that enable to generate VHDL code from a model compliant with our hardware accelerator metamodel. Future rules will also propose modification of the electronic design to enhance placement on the target FPGA, using algorithms proposed in [43].

6.2. Transformations and heterogeneous design framework for embedded systems

Participants: Lossan Bondé, Pierre Boulet, Jean-Luc Dekeyser, Cédric Dumoulin, Anne Etien, Antoine Honoré, Emmanuel Renaux.

6.2.1. Model transformation design and implementation

To reduce the "time to market", the DaRT project proposes a high level modeling environment for software and hardware design. The design process is then decomposed in successive transformation towards the different specific meta-models. However the definition of these transformations raises issues as: (i) the expression of the transformations often uses a formalism very different from the metamodels under which they rely, (ii) the documentation of the transformations interweaves implementation details.

Our experience with the transformation engine ModTransf and the internal needs of standardisation for the documentation and communication lead to the proposition of a common formalism TrML (Transformation Modeling Language). This formalism proposes:

- a graphical representation of the transformations similar to the expression of the meta-models;
- a structural decomposition of the transformations into rules smaller and implying the transformation of one or several elements;
- a focus on the "what" (what should be transformed) rather than the "how" (how implement the transformation);
- a black boxes mechanism to call existing code, within a rule;
- a mechanism to enhance the input and output meta-models in order to add classes or properties for the transformation time.

The concepts of this language are precisely defined. This definition takes the form of a profile and is formalised in a metamodel. This formalism and the approach have been experimentally validated on extract of the Gaspard supply chain. Furthermore, the concepts introduced by TrML served as the foundation for the development of the MoMoTE software.

6.2.2. Heterogeneous design framework and traceability

The Dart project promotes the reuse of IPs for the design of SoC. Such IPs come from different sources with heterogeneous models (different abstraction levels). This approach reduces « time to market », but its application requires new design methodologies.

Gaspard proposes a methodology based on Model Driven Engineering which intends the use of many simulation and execution platforms (Java, OpenMP, SystemC, VHDL, etc.) at different levels of abstraction (TLM, RTL, etc.). Models of different platforms and abstraction levels are generated in Gaspard by model transformations. The heterogeneity of the targeted platforms leads to an interoperability problem.

We propose a solution based on MDE model transformations techniques to provide a framework to deal with this interoperability problem [11]. This solution is performed in three steps. First, we introduce traceability in model transformations; therefore a trace model is generated along with model transformations. This trace model is then used as an entry model for a transformation which generates an interoperability bridge model. Finally, the code for interoperability is generated from the bridge model of the previous step. In order to automate the process, metamodels for traceability and interoperability bridge have been designed. We also provide the description of the different transformations involved in the process.

6.3. Verification, compilation, optimization and simulation issues

Participants: Yassine Aydi, Rabie Ben Atitallah, Abou El Hassan Benyamina, Pierre Boulet, Jean-Luc Dekeyser, Abdoulaye Gamatié, Calin Glitia, Sébastien Le Beux, Philippe Marquet, Ashish Meena, Samy Meftali, Smaïl Niar, Éric Piel, Imran Rafiq Quadri, Éric Rutten, Julien Taillard, Huafeng Yu.

6.3.1. Verification based on synchronous models

The objective of the synchronous modeling of data-intensive computations is to make the link between analysis and compilation techniques of parallel systems, on the one hand, and on the other hand, analysis and verification techniques of synchronous data-flow and automata-based languages.

We obtained the first results in this study dedicated to the modeling of data-intensive parallel applications following the synchronous approach. We have defined a synchronous dataflow equational model of Gaspard applications, which enables to address design correctness issues (e.g. verification of frequency/latency constraints) using the formal tools and techniques provided by the synchronous technology. More specifically, we illustrated a synchronizability analysis using affine clock systems. In affine clock systems, two different signals are said to be synchronizable if there is a dataflow preserving way to make them actually synchronous. The synchronizability issue is solved quite easily with synchronous models while it is not possible with Gaspard only.

6.3.2. Compilation and optimization

The Array-OL transformation toolbox proposed in 2005 has been completely implemented and integrated in Gaspard. The change paving transformation has been extended to include a new case [44] and the tiling transformation has been much improved [12].

We have proposed a "pattern-based scheduling algorithm" to handle data-parallel repetitions [13]. This algorithm solves the allocation, assignment and scheduling of a repetition of independent identical tasks on a distributed architecture taking into account the routing of the communications, the processing resource usage optimization (number of processors) and generates a repetitive allocation, assignment and scheduling pattern. This pattern repetition allows to build a repetitive execution code in order to avoid code explosion and to ensure a complexity that is independent on the number of tasks. This heuristic will be used as the regular part of a GILR heuristic. Work has started to study the combination of this pattern-based heuristic with a genetic algorithm one.

6.3.3. Simulation

Simulation has a strong importance in SoC design as the production cost is extremely high and the design is often strongly separated between software and hardware, leading to high number of bugs at the integration of the two. Simulation at a high level of abstraction is useful because it can be done early in the development of the SoC, it permits the developer to debug the system while keeping the concepts closed to the design concepts and it allows very fast simulation time.

6.3.3.1. MPSoC Simulation at TLM-PA Level

Compiling the software for the host processor instead of the target processors [73] permits to speed the simulation up even more and ease the debugging. We propose to enhance this technique by leveraging the fact that in Gaspard the code of the application is generated from the model. This allows to show the user the execution of the program in term of *patterns* usages, the main data element of a Gaspard program, instead of reading and writing of bytes. Moreover, with few adequate data provided by the user, it is possible to give information about the simulation such as the execution time or the power consumption with more accuracy.

We have started the development of the model transformations required to generate the code. The target is currently SystemC code which contains both the simulation of the hardware based on existent IPs (at the TLM-PVT abstraction level) and specific components used to simulate the application. The two parts of the code articulates around a CPU wrapper specially craft according to the hardware description.

6.3.3.2. MPSoC Simulation at TLM-PVT Level

To use the tremendous hardware resources available in next generation multiprocessor Systems-on-Chip (MPSoC) efficiently, rapid and accurate design space exploration (DSE) methods are needed to evaluate the different design alternatives. In our work, we focus on the use of Transaction Level Modeling (TLM) in an MPSoC design exploiting the simplification inter-module communication transactions. Therefore, modelling MPSoC architectures at TLM becomes easier and faster than at the Cycle Accurate Bit Accurate (CABA) level.

We developed a framework that makes fast simulation and performance evaluation possible early in the MPSoC design flow, thus reducing the time-to-market [23] [24]. In this framework and within the TLM approach, we give an efficient implementation to the conventional timed Programmer's View (PVT) level by introducing two modeling sublevels. The first one is PVT Transaction Accurate (PVT-TA) and the second is PVT Event Accurate (PVT-EA). PVT-TA operates at a relatively high abstraction level and does not take a specific communication protocol into account. This permits a rapid exploration of a large solution space by eliminating non-interesting regions from the DSE process. Solutions selected at the PVT-TA level are then forwarded for a new exploration at PVT-EA. This second sublevel specifies a precise communication protocol and takes architectural delays into account. Because estimation methodology that we developed for the PVT-EA is more accurate, it is possible at the price of less simulation speed, to locate the most efficient architecture configurations. PVT-TA and PVT-EA permit the use of PVT models in a coherent methodology, and to have accurate estimations in an efficient way.

To implement our PVT framework, architectural components models (such as processors, caches, interconnection network and a hardware accelerator) for MPSoC design have been developed and a timing model was integrated in our sublevels as a plug-in using SystemC.

Our strategy is based on identifying each component's pertinent activities: the number and types of executed instructions for the MIPS processor; hits and misses for the caches; the number of transmitted/received packets for the interconnect; and the number of read and write operations for the shared memory modules. In addition to counting the activities, execution time estimation also requires attributing an execution time to each activity. In our approach, execution times are measured from the CABA platform and injected into the timing model.

Simulation results show that with PVT-TA it possible to accelerate the simulation by a factor of up to 18. Adding processors increased this speedup factor due to the amplification of the communication between the processors and the shared memory modules. Despite its performance in terms of speedup, the PVT-TA sublevel suffers from a significant performance estimation error. This error can be as much as 28% and increases when communications become more significant. The performance error with PVT-EA was reduced to nearly zero. However, the speedup decreases by less than 30% compared to PVT-TA. Our approach is also interesting in terms of modeling effort. It allows designers to develop simulation platforms in less time. Using such a tool early in the design process reduces the exploration space, shortens the time-to-market, and increases the design team's productivity. In terms of lines of code, required to design an MPSoC platform for the three approaches: CABA, PVT-TA and PVT-EA. The modeling effort using PVT-TA and PVT-EA is reduced, respectively, by a factor of 34.8% and 26.4%.

6.3.4. Massively Parallel Processing SoC

We began works on the definition and design of mppSoC, a massively parallel system on a chip. The objective of the mppSoC project is to reconsider the interest of massively parallel machines with nowadays design methodologies based on IP reuse and nowadays integration technologies.

MppSoC is a SIMD architecture composed of a grid of processing elements (the PEs) and memories connected by a regular neighbourhood network and a general purpose global network.

Some improvements of the system architecture are possible because of the high degree of integration: The mppSoC PEs share most of their design with the control processor, the integrated network allows to exchange data between PEs, but also between the control processor and the PE memories, and even to connect the external devices to the system.

We have design a cycle-accurate bit-accurate SystemC simulator of this architecture, and an implementation prototype on FPGA. A complete tool chain based on an data-parallel language allows to generate binary programs that execute both on the simulator and the hardware implementation.

We have been able to integrate 16 processing elements running in a parallel way in a FPGA. In the future, we will increase the number of implemented PEs and will add some communication resources [27].

7. Contracts and Grants with Industry

7.1. The PROTES Project: A Carroll Project

Partners: CEA, Thales, INRIA (AOSTE, DaRT, ESPRESSO).

This project concerns the effort of standardisation of a UML profile for embedded and real time systems. The first step of this project was to initiate a request for proposal (RFP) to the OMG. The second step was to propose an answer to the RFP. The answer has taken the form of the MARTE profile proposal made by the ProMarte group[6]. This later is composed of the Protes partners and new partners like Alcatel, Artisan, IBM, Loockeed Martin, Telelogic...

In this project, three INRIA teams are involved. All of them are concerned with synchronous data-flow/control-flow models. This opportunity to develop together a UML profile for embedded and real-time systems and to support this proposal to the OMG strengthens internal collaborations between DaRT, AOSTE and ESPRESSO.

This project is now closed, and a new project called Cortess ensures the support for the end of the standardization process.

7.2. The PEAMS Project: A Carroll Project

Partners: CEA, Thales, INRIA (ALCHEMY, AOSTE, DaRT).

The objective of this project is to evaluate different architecture options in order to set up the Ter@ops project. The associated programming models will be also defined. The needs of the development framework will be defined and different tool options will be proposed. PEAMS will concentrate on data streaming dominated applications.

In this context, the DaRT project proposes his knowledge in the modeling of parallel architecture performance. It will participate to the extension of the SPEAR metamodel in order to enable the modeling of the Ter@ops architecture.

7.3. The OpenEmbedd Project: A RNTL Project

Partners: Airbus, Anyware Technologies, CEA, CS SI, France Telecom, INRIA (AOSTE, DaRT, ESPRESSO), LAAS (CNRS), Thales Aerospace, Thales R&D, Verimag.

The OpenEmbedd project[7] aims is to develop an engineering model driven open-source platform for real time and embedded systems. It deals with (1) UML standard for Real Time and Embedded systems, (2) innovating technology for interoperability, (3) mastering methodology chain, (4) real time models for simulation. The tools are evaluating in practical domains, e.g. the aeronautic sector, automobile sector, and telecom sector. The project will succeed in providing a technological core for Model Driven Engineering, by producing a set of tools dealing with different concerns about real time and embedded systems, and by validating an approach in the representative domains, in both applicative and methodological concerns. Software developed will be open-source. Future platform aims to federate academic partners effort and will guarantee a wide diffusion of the software.

[6] http://www.promarte.org
[7] http://www.openembedd.org

Our OpenEmbedd partners are Airbus, Anyware Technologies, CEA, CS SI, France Telecom, INRIA, LAAS (CNRS), Thales Aerospace, Thales R&D, Verimag. The project has link with three competitiveness poles: Minalogic, System@tic, Aerospace Valley.

The activity of the DaRT Project in the OpenEmbedd RNTL project is to normalize models about real time and embedded systems domains, and more precisely the MARTE profile. The objectives are to participate to elaboration of a graphical editor generated from OpenEmbedd tools, and to work on plugins dedicated to simulation and checking. During the OpenEmbedd first technical workshop that has been held in November the 22th and 23th, DaRT presented its transformation graphical notation TrML.

7.4. The Ter@ops Project: A System@tic Project

The Ter@ops project of the System@tic competitiveness pole aims at developing an hardware platform and the associated development framework for computation intensive applications. This project has just started in December 2006.

7.5. Collaboration within the competitiveness pole I-Trans

I-Trans is the official industrial cluster, which aims at bringing together major French actors in rail technology and innovative transport systems. The DaRT project strongly participates to this initiative through the collaborations with both concerned academic and industrial actors.

In this direction, we already have many discussions with partners (Inrets-Estas, Lagis, Lamih, Utc/Heudiasyc, Alstom and Certifier). These discussions lead to a few project proposals (Geneve, Klifr) that are under submission for ANR (Agence Nationale de la Recherche) funding. The goal of these proposals concerns on the one hand, the decrease of validation and certification costs in the implementation of the new European railway system ERTMS/ETCS and on the other hand, the ease of interoperability through the mutual recognition of ERTMS components between European member countries. This qualification requires costly and long tests. More specifically, in the context of innovate equipments for railways, the objective consists in developing methods, models and tools dedicated to the generation of scenario tests for the validation of ERTMS components.

The contribution of the DaRT project to the definition of solutions to the above issues is twofold: first, its capabilities in the design of architectures and the association between application and architecture, so as to explore impacts on the test of the deployment on embedded architectures; second, its design experience in the automotive domain (see section 8.1).

These aspects are currently studied by a starting PhD student in the DaRT project, conjointly with test problematics. This thesis is co-supervised with the Lagis laboratory.

7.6. Collaboration with CEA List

Partners: CEA List, DaRT

Since last year we have started a point to point collaboration with CEA around an a UML profile for co-design. This work is done together with a PhD student (CEA funding). The main objective consists in defining a metamodel for hardware architecture for the future MARTE standard. The results of this research could be also integrated in the Gaspard tools at INRIA and in the AccordUML environment at CEA.

This collaboration is complementary with the above partnership between CEA and DaRT in the Protes project.

7.7. Collaboration with Valeo - CNRT Futurelec

Partners: Valeo, INRIA (DaRT), L2EP

The objective of this project consists of fitting Gaspard for high performance computing and meta-computing. This work is done with a PhD student (Valeo/region funding). The results of this research are to be considered for integration in the Gaspard tool-set at INRIA and will be tested on finite element code for electric alternator simulator developed by Valeo and L2EP.

7.8. SoCLib RNRT Platform Project

Partners: CEA, CNRS, Thales Communications, ST Microelectronics, Prosilog, TurboConcept, and 13 academic laboratories.

The goal of SoCLib project[8] is to build an open platform for modelling and simulation of multi-processors system on chip, that can be used by both universities and industrial companies. The core of the platform is a library of simulation models for virtual components (IP cores), with a guaranteed path to silicon.

The DaRT team participated to this effort via the CNRS SoCLib "equipe-projet". Our contribution concerned the optimisation of the SystemC runtime. We proposed adapters for interoperability. Recently, we have also contributed to the project by adding performances estimation models into the available IPs at both CABA and TLM-T levels.

8. Other Grants and Activities

8.1. ModEsay Interreg III A Franco-English Cooperation

The ModEasy project[9] develops software tools and techniques in order to facilitate the development of reliable microprocessor-based electronic (embedded) systems using advanced development and verification systems. The defined tools will be evaluated in practical domains such as automotive.

We have two partners in this project. First, the University of Kent, which works on formal system verification, embedded system development support and hardware integration from research councils and industry. Second, the IEMN (Institut Electronique Microelectronique Nanotechnologies), which has substantial expertise in the safety of land-based transportation systems especially for collision avoidance.

In this context, we have proposed a FPGA prototype that merges a reactive cruise control and an anti-collision radar in a single chip. The merge of both system parts has been realized by hand, but current efforts are trying to make the design flow fully automatic, from an application UML modeling toward a FPGA implementation.

8.2. International initiatives

8.2.1. ECSI

The European Electronic Chips & Systems design Initiative Missions are to identify, develop and promote efficient methods for electronic system design, with particular regards to the needs of the System-on-Chip and to provide ECSI members with a competitive advantage in this domain for the benefit of the European industry. The list of participants is on:
http://www.ecsi.org.

Our team became an ECSI member in 2004. In this context we organized the ECSI conference in Lille: FDL'04. Pierre Boulet is a member of the executive committee of ECSI and secretary of ECSI.

8.2.2. University of California - Irvine

We started collaboration with the Center of Embedded Computer Systems (CECS) of the University of California at Irvine around networks on-chip design. Our objective consists mainly in modelling high performance networks as multistage Delta networks as cycle accurate, SystemC, reusable IPs [80]. We plan to continue this in this topic our collaboration and move in the future to FPGA implementations of these networks. In fact, aspects of dynamic reconfiguration of FPGAs make them very adapted to 'intelligent' NoC implementation.

[8] http://soclib.lip6.fr/soclib.html
[9] http://www.lifl.fr/modeasy

8.2.3. University of Montreal

The collaboration with University of Montreal laboratory continued this year. The interrested laboratory is the LASSO (Laboratoire d'Analyse et de Synthèse des Systèmes Ordinés[10]), from DIRO department (Département d'Informatique et de Recherche Opérationnelle[11]). Student Frédéric Bastien came in the team during two months, and worked on the VHDL description of a massively parallel machine on FPGA.

8.2.4. ENS Tunis

In the relation with the co-advising of a PhD student on the topic of "Modeling and high-level design of communication architectures for systems on chip : network on chip with dynamical reconfiguration", a collaboration with the team of Pr. Mohamed Abid at CES-ENIS is being built up. This collaboration is supported by the STIC Inria-Tunisia program, which aims at promoting the design of metamodels, transformation tools and techniques for the implementation of reconfigurable systems-on-chip. The resulting codesign environment will be validated on embedded systems dedicated to security in automobile, and more specifically in the design of cruise control systems integrating anti-collision radars.

Several successful student exchanges have been realized in 2006 between DaRT and CES-ENIS. Many other exchanges are already programmed for 2007.

8.2.5. University of Oran

A collaboration has started with the university of Oran, Algeria. Abou El Hassan Benyamina has been invited for one year to initiate the collaboration. He is working with Pierre Boulet on scheduling and mapping algorithms for SoC.

8.3. National initiatives

We are members of the ASR[12] and SoC-SiP[13] GDRs (research groups from the CNRS).

9. Dissemination

9.1. Scientific Community

Pierre Boulet was in the steering committee and the program committee of FDL'06[14]. He is the UML topic program chair of FDL'07. He is secretary and member of the executive committee of ECSI. He has been invited to give a lecture at the ECSI UML profiles for embedded systems workshop in March 2006 and at the third MDD4DRES school[15] in September 2006, and to participate to the UML panel of FDL06. He runs the UML for SoC design mailing list (umlsoc@univ-lille1.fr) with 136 subscribers from all over the world. He is a member of the INRIA evaluation committee since September 2005. He was in 5 PhD thesis committees in 2006 including reporter for the thesis of Christophe Jouvray, defended 11 December 2006 at Paris-Sud university.

Jean-Luc Dekeyser has been member of different program committees: ECMDA-FA 2006 recosoc06; invited speaker in Soc'06 tempere, referee for some journals and conferences in MDE and SoC design. He is project leader of the ModEasy interreg. He was in about ten PhD and HdR thesis committees. As director of the Ph.D. program at LIFL, Inria and Doctoral School SPI, he was involved in the belgium/france relationship concerning PhD program. He was visiting ENIS Sfax tunisa in june and november 2006.

[10] http://www.iro.umontreal.ca/~lablasso/lablasso/
[11] http://www.iro.umontreal.ca
[12] http://asr.cnrs.fr/
[13] http://www.lirmm.fr/soc_sip/
[14] http://www.ecsi.org/fdl
[15] http://www.mdd4dres.info/

Cedric Dumoulin was in the program committee of IDM 06 and IDM 07 (French workshop on Model Driven Engineering) held in Paris[16]. He was the organization chair of IDM'06 which was held in Lille. He was in one Phd thesis committee.

Abdoulaye Gamatié, Éric Rutten and Huafeng Yu participated to the 13th Synchronous Workshop[17], Alpe d'Huez, november 27 – december 01, 2006.

Sébastien Lebeux, Jean-Luc Dekeyser and Anne Etien participated to the ModEasy meeting in the Kent University (November 7th).

Philippe Marquet was in the organization committee of the 2e Journées sur l'Ingénérie Dirigée par les Modèles (IDM 06, Lille, France, June 26-28 2006). He is in the program committee of the 15th International Conference on Real-Time and Network systems (RTNS 07, Nancy, France, March 29-30, 2007).

Samy Meftali was in the program committee for DATE'06 [18].

Eric Rutten is co-editor of a special issue of the journal of Discrete Events Dynamical Systems on the modelling of reactive systems. He was in the PhD committees of A. Kerbaa (Computer Science, University of Grenoble 1, Verimag) as a reporter, and of O. Labbani (Computer Science, University of Lille1, LIFL). He was a member of the recruiting commission of University of Western Brittany(UBO), in Brest. He made an invited seminar at LRI at the University of Orsay.

Emmanuel Renaux was at the Kermeta Days (October 10th) in Rennes The goal of the Kermeta days is to promote Model Driven Engineering tools. It was the opportunity to bring together research communities that share an interest about Kermeta language. We are working on potential collaboration around model transformation.

Emmanuel Renaux, Antoine Honoré, Cédric Dumoulin and Anne Etien participated to the OpenEmbedd meeting in Rennes (November 22th and 23th)

9.2. Teaching

As the DaRT team is mostly composed of professors and associate professors, we have a very large teaching activity. The more directly related to the research themes of the team are the master-level courses "System-on-Chip design" (Pierre Boulet, Jean-Luc Dekeyser, Samy Meftali, Abdoulaye Gamatié) and "Introduction to real-time operating systems" (Philippe Marquet), "Simulation of Systems and Architectures" (Philippe Marquet and Samy Meftali). Rabie Ben Atitallah, Éric Piel and Julien Taillard have respectively given lectures about Assembler language, Linux installation and administration, and Java programming to Licence and Master students of Université de Valenciennes and USTL.

The following internships were advised in the team:

- Imran Rafiq Quadri, M2R Computer Science, Université des Sciences et Technologies de Lille,
- Calin Glitia, M2R Computer Science, ENS Lyon,
- Cyril Joly, M2P Computer Science, Université de Valenciennes
- Axel Lienard, M2P Computer Science, Université de Valenciennes
- Mohammed-Jalal Kebbaj, M2P Computer Science, Université des Sciences et Technologies de Lille
- Frédéric Bastien, M2R, Université de Montreal
- Joan Kubera, M1 Computer Science, Université des Sciences et Technologies de Lille,
- Simon Duquennoy, M1 Computer Science,Université des Sciences et Technologies de Lille.

[16] http://www.planetmde.org/idm07

[17] http://pop-art.inrialpes.fr/~girault/Synchron06

[18] http://www.date-conference.com/conference/?page=date06_presentations

10. Bibliography

Major publications by the team in recent years

[1] A. C. Aljundi, J.-L. Dekeyser, M. T. Kechadi, I. D. Scherson. *A universal performance factor for multi-criteria evaluation of multistage interconnection networks.*, in "Future Generation Comp. Syst.", vol. 22, n^{o} 7, 2006, p. 794-804.

[2] A. Amar, P. Boulet, J.-L. Dekeyser. *Algorithms and Tools for Parallel Computing On Heterogeneous Clusters*, ISBN: 1-60021-049-X, chap. Towards Distributed Process Networks with CORBA, Nova Science Publishers, Inc, 2006.

[3] P. Boulet, J.-L. Dekeyser, C. Dumoulin, P. Marquet. *MDA for System-on-Chip Design, Intensive Signal Processing Experiment*, in "FDL'03, Fankfurt, Germany", September 2003.

[4] P. Boulet, C. Dumoulin, A. Honoré. *From MDD concepts to experiments and illustrations*, S. Gérard, J.-P. Babeau, J. Champeau (editors). , ISBN:1-905209-59-0, chap. Model Driven Engineering for System-on-Chip Design, ISTE, September 2006, http://iste.co.uk/index.php?f=a&ACTION=View&id=147.

[5] L. Charest, P. Marquet, J.-L. Dekeyser, El Mostapha. Aboulhamid, G. Bois. *Using design pattern for type unification, structural unification, semantic clarification and introspection in SystemC*, in "Annals for Micro and Nano Systems", 2006.

[6] A. Cuccuru, J.-L. Dekeyser, P. Marquet, P. Boulet. *Towards UML 2 extensions for compact modeling of regular complex topologies*, in "MODELS/UML 2005, ACM/IEEE 8th international conference on model driven engineering languages and systems, Montego Bay, Jamaica", October 2005.

[7] A. Gamatié, T. Gautier. *Synchronous Modeling of Avionics Applications using the SIGNAL Language.*, in "IEEE Real Time Technology and Applications Symposium, Washington D.C.", 2003, p. 144-151.

[8] O. Labbani, J.-L. Dekeyser, É. Rutten. *Safe Design Methodology for an Intelligent Cruise Control System with GPS*, in "VTC'2006 Fall: 64th IEEE Vehicular Technology Conference, Montréal, Québec, Canada", (Also to appear as an IEEE ITSS Newsletter in December 2006), September 2006.

[9] S. Meftali, J. Vennin, J.-L. Dekeyser. *Automatic Generation of Geographically Distributed System Simulation Models for IP/SoC Design*, in "The 46th IEEE International Symposium on Circuits and Systems, Cairo, Egypt", December 2003.

[10] H. Sbeyti, S. Niar, L. Eeckhout. *Pattern-Driven Prefetching for Multimedia Applications on Embedded Processors*, in "Journal of System Architecture", vol. 52, n^{o} 4, April 2006, p. 199-212.

Year Publications

Doctoral dissertations and Habilitation theses

[11] L. Bondé. *Transformations de Modèles et Interopérabilité dans la Conception de Systèmes Hétérogènes sur Puce à base d'IP*, Doctorat en Informatique, Université des Sciences et Technologies de Lille, December 2006.

[12] O. LABBANI. *Modélisation de haut niveau du contrôle dans des applications de traitement systématique à parallélisme massif*, Doctorat en Informatique, Université des Sciences et Technologies de Lille, November 2006.

[13] A. MEENA. *Allocation, Assignation et ordonnancement pour les systèmes sur puce multi-processeurs*, Doctorat en Informatique, Université des Sciences et Technologies de Lille, December 2006.

Articles in refereed journals and book chapters

[14] A. C. ALJUNDI, J.-L. DEKEYSER, M. T. KECHADI, I. D. SCHERSON. *A universal performance factor for multi-criteria evaluation of multistage interconnection networks.*, in "Future Generation Comp. Syst.", vol. 22, n^{o} 7, 2006, p. 794-804.

[15] A. AMAR, P. BOULET, J.-L. DEKEYSER. *Algorithms and Tools for Parallel Computing On Heterogeneous Clusters*, ISBN: 1-60021-049-X, chap. Towards Distributed Process Networks with CORBA, Nova Science Publishers, Inc, 2006.

[16] P. BOULET, C. DUMOULIN, A. HONORÉ. *From MDD concepts to experiments and illustrations*, S. GÉRARD, J.-P. BABEAU, J. CHAMPEAU (editors). , chap. Model Driven Engineering for System-on-Chip Design, ISTE, September 2006, http://iste.co.uk/index.php?f=a&ACTION=View&id=147.

[17] L. CHAREST, P. MARQUET, J.-L. DEKEYSER, EL MOSTAPHA. ABOULHAMID, G. BOIS. *Using design pattern for type unification, structural unification, semantic clarification and introspection in SystemC*, in "Annals for Micro and Nano Systems", 2006.

[18] A. GAMATIÉ, T. GAUTIER, P. LE GUERNIC. *Synchronous Design of Avionic Applications based on Model Refinements*, in "Journal of Embedded Computing (JEC), IOS Press", (To appear), 2006.

[19] A. GAMATIÉ, T. GAUTIER, P. LE GUERNIC, J.-P. TALPIN. *Polychronous Design of Embedded Real-Time Applications*, in "ACM Transactions on Software Engineering and Methodology (TOSEM)", (To appear), 2006.

[20] S. MEFTALI, J. VENNIN, J.-L. DEKEYSER. *An optimized distributed simulation environment for SoC design*, in "Annals for Micro and Nano Systems", 2006.

[21] H. SBEYTI, S. NIAR, L. EECKHOUT. *Pattern-Driven Prefetching for Multimedia Applications on Embedded Processors*, in "Journal of System Architecture", vol. 52, n^{o} 4, April 2006, p. 199-212.

Publications in Conferences and Workshops

[22] R. B. ATITALLAH, L. BONDÉ, S. NIAR, S. MEFTALI, J.-L. DEKEYSER. *Multilevel MPSoC performance evaluation using MDE approach.*, in "International Symposium on System-on-Chip 2006, Tampere, Finland", november 2006.

[23] R. B. ATITALLAH, S. NIAR, A. GREINER, S. MEFTALI, J. L. DEKEYSER. *Estimating Energy Consumption for an MPSoC Architectural Exploration*, in "ARCS'06: Architecture of Computing Systems, Frankfurt, Germany", March 2006, http://arcs06.cs.uni-frankfurt.de/.

[24] R. B. ATITALLAH, S. NIAR, S. MEFTALI, J.-L. DEKEYSER. *Accelerating MPSoC Performance Evaluation with TLM.*, in ""Advanced Computer Architecture and compilation for Embedded Systems ACACES", HIPEAC network of Excellence summer school, Laquilia, Italy", ISBN 90 382 0981 9, july 2006, p. 229–232.

[25] G. DELAVAL, E. RUTTEN. *A domain-specific language for task handlers generation, applying discrete controller synthesis*, in "Proceedings of the 21st ACM Symposium on Applied Computing, SAC'06, Special track on Embedded Systems: Applications, Solutions, and Techniques, Dijon, France", April 2006.

[26] C. DUMOULIN, A. CUCCURU, A. HONORÉ. *Motifs pour la métamodélisation : Relation, relation Dirigée, association*, in "2eme journées sur l'Ingénierie Dirigée par les Modèles, Lille France", In french, September 2006, p. 217-222, http://planetmde.org/idm06/actes.pdf.

[27] S. DUQUENNOY, S. LE BEUX, P. MARQUET, S. MEFTALI, J. DEKEYSER. *MpNoC Design: Modeling and Simulation*, in "15th edition of IP/SOC", 2006.

[28] A. GAMATIÉ, C. BRUNETTE, R. DELAMARE, T. GAUTIER, J.-P. TALPIN. *A Modeling Paradigm for Integrated Modular Avionic Design*, in "Proceedings of Euromicro Conference Software Engineering and Advanced Application (SEAA'06), Cavtat/Dubrovnik - Croatia", August 2006.

[29] A. GAMATIÉ, T. GAUTIER, P. LE GUERNIC. *Towards Static Analysis of SIGNAL Programs using Interval Techniques*, in "Proceedings of Synchronous Languages, Applications, and Programming (SLAP'06), Vienna - Austria", March 2006.

[30] A. GIRAULT, H. YU. *A flexible method to tolerate value sensor failures*, in "11th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA 2006), Prague, Czech republic", September 2006, p. 86-93.

[31] O. LABBANI, J.-L. DEKEYSER, P. BOULET, É. RUTTEN. *UML2 Profile for Modeling Controlled Data Parallel Applications*, in "FDL'06: Forum on specification and Design Languages, Darmstadt, Allemagne", September 2006.

[32] O. LABBANI, É. RUTTEN, J.-L. DEKEYSER. *Safe Design Methodology for an Intelligent Cruise Control System with GPS*, in "64th IEEE Vehicular Technology conference, Montréal, Québec, Canada", September 2006.

[33] S. LE BEUX, V. GAGNÉ, E. M. ABOULHAMID, P. MARQUET, J.-L. DEKEYSER. *Hardware/Software Exploration for an Anti-collision Radar System*, in "The 49th IEEE International Midwest Symposium on Circuits and Systems, San Juan, Puerto Rico", August 2006, http://www.lifl.fr/west/publi/MWCAS06.pdf.

[34] S. LE BEUX, P. MARQUET, O. LABBANI, J.-L. DEKEYSER. *FPGA Implementation of Embedded Cruise Control and Anti-Collision Radar*, in "DSD'2006, 9th Euromicro conference on digital system design, Dubrovnik, Croatia", August 2006, http://www.lifl.fr/west/publi/dsd06.pdf.

[35] E. PIEL, P. MARQUET, J. SOULA, J.-L. DEKEYSER. *Real-Time Systems for Multi-Processor Architectures*, in "14th International Workshop on Parallel and Distributed Real-Time Systems, In conjunction with IPDPS, 20th IEEE International Parallel and Distributed Processing Symposium, Island of Rhodes, Greece", Invited paper, IEEE Computer Society Press, April 2006, http://dce.felk.cvut.cz/hanzalek/wpdrts06/.

[36] J. Taillard, J.-L. Dekeyser, F. Piriou. *A Design of a UML Profile for Meta-computing*, in "PDP 2006, 14th Euromicro Conference on Parallel, Distributed and Network-based Processing, Montbéliard-Sochaux, France", February 2006, http://www.pdp2006.org.

[37] J.-P. Talpin, C. Brunette, T. Gautier, A. Gamatié. *Polychronous Mode Automata*, in "Proceedings of 6th Annual ACM Conference on Embedded Software (EMSOFT'06), Seoul - South Korea", October 2006.

[38] H. Yu, A. Gamatié, E. Rutten, P. Boulet, Jean-Luc. Dekeyser. *Vers des transformations d'applications à parallélisme de données en équations synchrones*, in "Actes du Symposium en Architecture de Machines, Sympa'06, Perpignan, France", October 2006.

Internal Reports

[39] A. Gamatié, H. Yu, É. Rutten, P. Boulet, J.-L. Dekeyser. *Synchronous Modeling of Data Intensive Applications*, Rapport de Recherche, n^{o} 5876, INRIA, April 2006, http://hal.inria.fr/inria-00001216.

[40] O. Labbani, J.-L. Dekeyser, P. Boulet, É. Rutten. *Introducing Control in the Gaspard2 Data-Parallel Metamodel: Synchronous Approach*, Rapport de Recherche, n^{o} 5794, INRIA, January 2006, http://hal.inria.fr/inria-00070228.

[41] O. Labbani, J.-L. Dekeyser, É. Rutten. *Separating Control and Data Flow: Methodology and Automotive System Case Study*, Research Report, n^{o} RR-5832, INRIA, February 2006, http://hal.inria.fr/inria-00070193.

[42] S. Le Beux, V. Gagné, El Mostapha. Aboulhamid, P. Marquet, J.-L. Dekeyser. *Hardware/Software Exploration for an Anti-collision Radar System*, Research Report, n^{o} 5820, INRIA, January 2006, http://hal.inria.fr/inria-00070205.

[43] S. Le Beux, P. Marquet, J.-L. Dekeyser. *FPGA Configuration of Intensive Multimedia Processing Tasks Modeled in UML*, Research Report, n^{o} RR-5810, INRIA, France, January 2006, http://hal.inria.fr/inria-00070214.

Miscellaneous

[44] C. Glitia. *Code transformations for systematic signal processing and memory size optimizations*, Master recherche, Université des sciences et technologies de Lille, 2006.

[45] I. R. Quadri. *Modélisation de topologies de réseaux d'interconnexions à base de multiplicités multidimensionnelles*, Master recherche, Université des sciences et technologies de Lille, 2006.

References in notes

[46] Object Management Group, Inc. (editor). *(UML) Profile for Schedulability, Performance, and Time Version 1.1*, http://www.omg.org/technology/documents/formal/schedulability.htm.

[47] Object Management Group, Inc. (editor). *U2 Partners' (UML 2.0): Superstructure, 2nd revised submission*, January 2003, http://www.omg.org/cgi-bin/doc?ptc/03-01-02.

[48] OBJECT MANAGEMENT GROUP, INC. (editor). *(UML 2.0): Superstructure Draft Adopted Specification*, July 2003, http://www.omg.org/cgi-bin/doc?ptc/03-07-06.

[49] OBJECT MANAGEMENT GROUP, INC. (editor). *UML 2 Superstructure (Available Specification)*, October 2004, http://www.omg.org/cgi-bin/doc?ptc/2004-10-02.

[50] OBJECT MANAGEMENT GROUP, INC. (editor). *UML Profile for Modeling Quality of Service and Fault Tolerance Characteristics and Mechanisms*, September 2004, http://www.omg.org/cgi-bin/doc?ptc/2004-09-01.

[51] OBJECT MANAGEMENT GROUP, INC. (editor). *SysML v0.9*, January 2005, http://www.omg.org/cgi-bin/doc?ad/05-01-03.

[52] OBJECT MANAGEMENT GROUP, INC. (editor). *UML Extension Profile for SoC RFC*, March 2005, http://www.omg.org/cgi-bin/doc?realtime/2005-03-01.

[53] M. D. ADAMS. *The JPEG-2000 Still Image Compression Standard*, Technical report, n^{o} N2412, ISO/IEC JTC 1/SC 29/WG 1, September 2001.

[54] A. AMAR, P. BOULET, P. DUMONT. *Projection of the Array-OL Specification Language onto the Kahn Process Network Computation Model*, in "International Symposium on Parallel Architectures, Algorithms, and Networks, Las Vegas, Nevada, USA", December 2005, http://sigact.acm.org/ispan05/.

[55] K. ARNOLD, J. GOSLING, D. HOLMES. *The Java Programming Language*, 3rd, Addison-Wesley, 2000.

[56] C. BASTOUL. *Code Generation in the Polyhedral Model Is Easier Than You Think*, in "PACT'13 IEEE International Conference on Parallel Architecture and Compilation Techniques, Juan-les-Pins", September 2004, p. 7–16.

[57] 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", vol. 91, n^{o} 1, January 2003, p. 64-83.

[58] G. BERRY. *Proof, Language and Interaction: Essays in Honour of Robin Milner*, chap. The Foundations of Esterel, MIT Press, 1998, http://www-sop.inria.fr/meije/esterel/doc/main-papers.html.

[59] P. BOULET, A. CUCCURU, J.-L. DEKEYSER, A. MEENA. *Model Driven Engineering for Regular MPSoC Co-design*, in "ReCoSoC-05, Montpellier, France", June 2005, http://www.lirmm.fr/RECOSOC05/.

[60] P. BOULET, J.-L. DEKEYSER, J.-L. LEVAIRE, P. MARQUET, J. SOULA, A. DEMEURE. *Visual Data-parallel Programming for Signal Processing Applications*, in "9th Euromicro Workshop on Parallel and Distributed Processing, PDP 2001, Mantova, Italy", February 2001, p. 105–112.

[61] P. CASPI, D. PILAUD, N. HALBWACHS, J.A. PLAICE. *Lustre: a declarative language for real-time programming*, in "Proceedings of the 14th ACM SIGACT-SIGPLAN symposium on Principles of programming languages (POPL'87)", ACM Press, 1987, p. 178-188.

[62] E. CONSORTIUM. *The Eclipse Project*, 2003, http://www.eclipse.org.

[63] A. CUCCURU. *Modélisation unifiée des aspects répétitifs dans la conception conjointe logicielle / matérielle des systèmes sur puce hautes performances*, Doctorat en Informatique, Université des Sciences et Technologies de Lille, December 2005.

[64] A. CUCCURU, P. MARQUET, J.-L. DEKEYSER. *IFIP International Federation for Information Processing*, vol. 176, chap. UML 2 as an ADL: Hierarchichal Hardware Modeling, Springer, 2005, p. 133-147.

[65] A. DEMEURE, A. LAFAGE, E. BOUTILLON, D. ROZZONELLI, J.-C. DUFOURD, J.-L. MARRO. *Array-OL : Proposition d'un Formalisme Tableau pour le Traitement de Signal Multi-Dimensionnel*, in "Gretsi, Juan-Les-Pins, France", September 1995.

[66] A. DEMEURE, Y. DEL GALLO. *An Array Approach for Signal Processing Design*, in "Sophia-Antipolis conference on Micro-Electronics (SAME 98), France", October 1998.

[67] P. DUMONT, P. BOULET. *Transformations de code Array-OL : implémentation de la fusion de deux tâches*, Technical report, Laboratoire d'Informatique fondamentale de Lille et Thales Communications, October 2003.

[68] P. DUMONT. *Étude des Transformations d'un Code Array-OL dans Gaspard*, Research Report, n^{o} 02-11, Laboratoire d'informatique fondamentale de Lille, Université des sciences et technologies de Lille, France, September 2002, http://www.lifl.fr/LIFL1/publications/2002-11.ps.

[69] P. DUMONT. *Spécification multidimensionnelle pour le traitement du signal systématique*, (In French), Thèse de doctorat (PhD Thesis), Laboratoire d'informatique fondamentale de Lille, Université des sciences et technologies de Lille, 2005.

[70] ESTEREL TECHNOLOGIES. *SoC Design, Validation and Verification*, 2002, http://www.esterel-technologies.com/v3/?id=29453.

[71] D. D. GAJSKI, R. KUHN. *Guest Editor Introduction: New VLSI-Tools*, in "IEEE Computer", vol. 16, n^{o} 12, December 1983, p. 11-14.

[72] T. GRANDPIERRE, C. LAVARENNE, Y. SOREL. *Optimized Rapid Prototyping for Real-Time Embedded Heterogeneous Multiprocessors*, in "Proceedings of the 7th International Workshop on Hardware/Software Codesign (CODES99), New York", ACM Press, May 3–5 1999, p. 74–78.

[73] S. HONDA, T. WAKABAYASHI, H. TOMIYAMA, H. TAKADA. *RTOS-Centric Hardware/Software Cosimulator for Embedded System Design*, in "Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS'04), Stockholm, Sweden", September 2004, http://portal.acm.org/ft_gateway.cfm?id=1016760&type=pdf.

[74] ITRS. *Design, 2001 edition*, 2001, http://public.itrs.net/.

[75] N. KADRI, S. NIAR, A. BABA-ALI. *Impact of Code Compression on the Power Consumption in Embedded Systems*, in "international conference on Embedded Systems and Applications ESA'03", June 2003.

[76] G. KAHN. *The Semantics of a Simple Language for Parallel Programming*, in "Information Processing 74: Proceedings of the IFIP Congress 74", J. L. ROSENFELD (editor). , North-Holland, IFIP, August 1974, p. 471–475.

[77] G. Kahn, D. B. MacQueen. *Coroutines and networks of parallel processes*, in "Information Processing 77: Proceedings of the IFIP Congress 77", B. Gilchrist (editor). , North-Holland, 1977, p. 993–998.

[78] P. Le Guernic, J.-P. Talpin, J.-C. Le Lann. *Polychrony for System Design*, in "Journal for Circuits, Systems and Computers", vol. 12, n^{o} 3, April 2003, p. 261–304.

[79] A. Meena, P. Boulet. *Model Driven Scheduling Framework for Multiprocessor SoC Design*, in "Workshop on Scheduling for Parallel Computing, SPC 2005, Poznan, Poland", September 2005, http://www.cs.put.poznan.pl/mdrozdowski/spc-ppam05/.

[80] S. Meftali, J.-L. Dekeyser, Isaac D. Scherson. *Scalable Multistage Networks for Multiprocessor System-on-Chip Design*, in "International Symposium on Parallel Architectures, Algorithms, and Networks, Las Vegas, Nevada, USA", December 2005, http://sigact.acm.org/ispan05/.

[81] S. Meftali, J. Vennin, J.-L. Dekeyser. *A fast SystemC simulation Methodology fo Multi-Level IP/SoC Design*, in "IFIP International Workshop On IP Based System-on-Chip Design, Grenoble, France", November 2003.

[82] S. Meftali, J. Vennin, J.-L. Dekeyser. *Automatic Generation of Geographically Distributed System Simulation Models for IP/SoC Design*, in "The 46th IEEE International Symposium on Circuits and Systems, Cairo, Egypt", December 2003.

[83] P. K. Murthy, E. A. Lee. *Multidimensional Synchronous Dataflow*, in "IEEE Transactions on Signal Processing", July 2002.

[84] P. K. Murthy. *Scheduling Techniques for Synchronous and Multidimensional Synchronous Dataflow*, Ph. D. Thesis, University of California, Berkeley, CA, 1996.

[85] S. Niar, L. Eeckhout, K. DeBosschere. *Comparing multiported cache schemes*, in "PDPTA-2003", June 2003.

[86] Object Management Group, Inc.. *MOF 2.0 Query / Views / Transformations RFP*, OMG paper, 2003.

[87] Object Management Group, Inc.. *Meta Object Facility (MOF) Core Specification, Version 2.0*, January 2006, http://www.omg.org/docs/formal/06-01-01.pdf.

[88] Open SystemC Initiative. *SystemC*, 2002, http://www.systemc.org/.

[89] OpenMP Architecture Review Board. *OpenMP Application Programme Interface*, May 2005, http://www.openmp.org/drupal/mp-documents/spec25.pdf.

[90] H. Sbeyti, S. Niar, L. Eeckhout. *Adaptive Prefetching for Multimedia Applications in Embedded Systems*, in "DATE'04, Paris, France", EDA IEEE, February 2004.

[91] D. Schmidt. *Model-Driven Engineering*, in "IEEE Computer", vol. 39, n^{o} 2, February 2006, p. 41-47.

[92] Y. Sorel, C. Lavarenne. *SynDEx Documentation Index*, INRIA, 2000, http://www-rocq.inria.fr/syndex/doc/.

[93] Y. SOREL, C. LAVARENNE. *Modèle unifié pour la conception conjointe logiciel-matériel*, in "Traitement du Signal (numéro spécial Adéquation Algorithme Architecture)", vol. 14, n^{o} 6, 1997, p. 569-578, http://www-rocq.inria.fr/syndex/pub.htm.

[94] Y. SOREL. *Massively Parallel Computing Systems with Real Time Constraints - The "Algorithm Architecture Adequation" Methodology*, in "Proceedings of the 1st International Conference on Massively Parallel Computing Systems, Los Alamitos, CA, USA", IEEE Computer Society Press, May 1994, p. 44–54.

[95] J. SOULA. *Principe de Compilation d'un Langage de Traitement de Signal*, (In French), Thèse de doctorat (PhD Thesis), Laboratoire d'informatique fondamentale de Lille, Université des sciences et technologies de Lille, December 2001.