



Project IST-2001-38314, COLUMBUS

Design of Embedded Controllers for Safety Critical Systems

Deliverable DTA1

Report on formal framework of meta-models – resp. INRIA

Reference Period: beginning of project – 31 December 2003

January 23, 2004

Project Co-ordinator

Organisation: *Department of Electrical and Computer Engineering
University of Cambridge*

Responsible person: *Dr John Lygeros*

Address: *University Campus
Rio, Patras, GR 26500, Greece*

Phone: *+30 2610 996458*

Fax: *+30 2610 991812*

E-mail: *lygeros@ee.upatras.gr*

Consortium

<i>Participant name</i>	<i>Acronym</i>	<i>Role</i>
1 <i>University of Patras</i>	<i>UPAT</i>	<i>Coordinator</i>
2 <i>University of Cambridge</i>	<i>UCAM</i>	<i>Contractorr</i>
3 <i>University of l'Aquila</i>	<i>AQUI</i>	<i>Contractor</i>
4 <i>Institut National de Recherche en Informatique et en Automatique</i>	<i>INRIA</i>	<i>Contractor</i>
5 <i>University of California, Berkeley</i>	<i>UCB</i>	<i>Contractor</i>
6 <i>Vanderbilt University</i>	<i>VU</i>	<i>Contractor</i>
7 <i>PARADES, Rome</i>	<i>PARADES</i>	<i>Subcontractor</i>

DOCUMENT HISTORY

Release	Date	Reason of change	Status	Distribution
0.1	13/01/2004	First draft	Draft	Partners
0.2	23/01/2004		Final	Partners

TABLE of CONTENTS

1.	LIST OF DUE DELIVERABLES FOR WPTA	4
2.	SUMMARY OF RESULTS	5
2.1	Recalling the objectives	5
2.2	It is possible to optimize across MoCC boundaries to improve performance and reduce errors in the design at early stages in the process.	5
2.3	Meta-modeling techniques for heterogeneous modeling	7
2.4	References	7
3.	MEETINGS, VISITS, EXCHANGES OF PEOPLE	9
3.1	Meetings	9
3.2	Visits	9
4.	MINUTES OF THE 10TH OCTOBER MEETING	10
5.	ANNEXE	13

1. LIST OF DUE DELIVERABLES FOR WPTA

Id	Deliverable	Respons. Partner	Original due date/ milestone	Revised due date/ milestone	Actual delivery	Status / Comments
DTA1	Report on formal framework of meta-models	INRIA	31/12/03		23/01/04	
DTA2	Report on use of meta-model for hybrid systems.	INRIA	30/6/04			
DTA3	Contribution to consolidated final project report	INRIA	30/6/04			

2. SUMMARY OF RESULTS

2.1 Recalling the objectives

To serve as a reference for assessing our results, we first quote here selected key sentences of the project document regarding this work package:

Execution environments and development environments are two essential components of embedded systems development. Meta models and meta modeling play a crucial role in both areas. In execution environments, meta modeling is used for capturing the semantics of models of computations and communications (MoCC). In development environments meta modeling offers an effective approach to formally model the abstract syntax and static semantics of domain-specific modeling languages. In this work package we will contribute to the development of advanced meta modeling techniques in both directions as described below.

- 1. An essential aspect of the research is the development of formal definition of the semantics of communications so that implementation choices will be correct by construction. Several formal models have been proposed over the years to capture one or more aspects of computations using a unifying theoretical framework introduced recently by Lee and Sangiovanni-Vincentelli. However, this denotational framework has only helped us to identify the sources of difficulties in combining different MoCC that are certainly needed when complex systems are being designed. [...] We believe that it is possible to optimize across MoCC boundaries to improve performance and reduce errors in the design at early stages in the process.*
- 2. Domain-specific modeling languages have significant impact on the design process. In embedded systems, where communication and computation always occur in the context of physical domain, domain-specific languages offer an effective way to structure the information about the system to be designed along the “natural dimensions” of the applications. [...] In this work package, we will extend previous work on the use of meta-models for the specification and validation of modeling languages for hybrid systems. Of particular interest are mathematical formalisms that enable the concise representation and composition of complex, multiple-aspect modeling languages.*

In comparison with the above stated objectives we state below a summary of what has been achieved at this stage of the project.

2.2 It is possible to optimize across MoCC boundaries to improve performance and reduce errors in the design at early stages in the process.

This paragraph summarizes the results obtained regarding the above point 1 quoted from the project document. Corresponding results have been mainly obtained through a deep, tight, and continuing cooperation between researchers from INRIA (Albert Benveniste, Benoît Caillaud, and Dumitru Potop-Butucaru), PARADES (Alberto Sangiovanni-Vincentelli), and UCB (Luca Carloni and ASV) – in addition, we have invited Paul Caspi, from VERIMAG, to join our group for this work. Main results are listed now.

2.2.1 Theory of heterogeneous modeling

Results. We have made key progresses in the *theory of heterogeneous modeling*, the theory dealing with how to give a precise semantics to the assembly of models based on different MoCCs, and how to map a given design onto an architecture with a different MoCC. To this end, as planned, we have started from the Lee and Sangiovanni-Vincentelli (LSV) tagged systems model.

- We have shown how to adapt this model with the objective of regarding the MoCC as a parameter. This being done, we were able to define *morphisms relating different MoCCs*, i.e., classes of mappings between different MoCCs, satisfying certain properties. Roughly speaking, morphisms formalize the generic concept of relaxing synchronization – it can be by

January 7, 2004

removing synchrony, by losing causality information, or by removing real-time information. Morphisms compose.

- With this at hand, we could define *heterogeneous parallel composition*, a composition that provides formal meaning to the assembly of models with different MoCCs.
- We could formally state what it means to preserve the semantics when deploying a design over an architecture not complying with the original MoCC of the design.
- Finally we got a first set of theorems characterizing when such a deployment does preserve the semantics. This formalizes the notion of “correct-by-construction” deployment.

We believe that this is an important body of work. Corresponding results have been published at the EMSOFT’03 conference [1]. They have been presented at an invited presentation by A. Benveniste at the FMCO’03 workshop [2], with a post-publication paper now in preparation. This whole work will be the subject of a joint long paper, in preparation for the special post-FMCO issue of the journal *Theoretical Computer Science*. Also, Pierre Le Maigat (research technologist at INRIA funded by COLUMBUS) has investigated a model of computation that encompasses synchronous and asynchronous semantics in an algebraic framework – the interest of this work is that corresponding morphisms are more powerful than those of our tagged systems model; for instance, causality can be refined into synchrony and vice-versa [6].

Perspectives. The subsequent steps of this direction of work will consist in making the above results more effective. The theorems obtained are not enough yet to derive effective algorithms for correct-by-construction deployment; the reason is that they deal with models of traces, i.e., infinite behaviours, not effective machines. Our next effort will consist in giving an effective variant of the LSV model, not for trace models, but for effective machines.

2.2.2 A framework to compare desynchronization and latency-insensitive design

Results. L. Carloni and A. Sangiovanni-Vincentelli have used the LSV model as a common framework to offer a comparative exposition of various design approaches: synchronous, asynchronous, GALS, latency-insensitive, and synchronous programming [5]. In particular, they studied the interplay among the concepts of event absence, event sampling, and communication latency in modeling distributed heterogeneous design. Furthermore, they presented a new comparison of synchronous program desynchronization and latency-insensitive design. The main operational difference between latency-insensitive design and synchronous program desynchronization can be expressed as follows: the former knows how to handle *black box* processes but does not know how to analyze/exploit *white box* ones (that are treated uniformly as if they were black box processes); the latter does not know how to handle black box processes and must analyze the inner structure of each white box process in the system (as well as the properties of each communicating pair), but it is clever in exploiting the information resulting from this analysis.

Perspectives. The above results naturally lead to consider two new research avenues for extending the work on latency-insensitive design: (1) for single-clock systems, the analysis of each process that comes as a white box module could identify its input/output causality dependencies to see if they allow us to absorb some stalling events at given states; (2) for multi-clock systems, the two approaches could be combined by applying hierarchically first the endo-isochronous analysis to handle event absence and, then, the latency-insensitive protocol mechanism to handle stalling events.

2.2.3 Improvements on the theory of desynchronization

Results. Dumitru Potop-Butucaru (the post-doc at INRIA working in part for COLUMBUS) and Benoît Caillaud have together discovered a flaw in the 2000 paper by Benveniste, Caillaud and Le Guernic that introduced the new concepts of endochrony and isochrony [3]. The flaw does not concern the central results, but still impairs the satisfactory handling of GALS architectures (GALS with more than 2 components are not covered!). In their quest for correcting this, they have found an important weakening of the two concepts of endochrony and isochrony, respectively called *weak-endochrony* and *weak-isochrony*. These new notions adequately address the problems due to possible internal concurrency inside the components. This is a major result that has been submitted for publication [4]. This paper provides effective criteria and algorithms for the synthesis of correct-by-construction deployment of synchronous designs over GALS architectures.

January 7, 2004

Perspectives. The follow-up of this work will join the former track. Techniques developed here will be in fact reused to address the case of tag machines, as mentioned in the previous bullet. Thus these two lines of work nicely complement each other, and open the route to a fully general algorithmic treatment of heterogeneity, for both modeling and deployment.

Concluding comments. This line of research conveniently addresses heterogeneous systems made of subsystems with drastically different MoCCs, e.g.: synchronous, asynchronous, with/without causality, with/without scheduling, timed/untimed, etc. *It is our opinion that, in this direction, we have already achieved much more than was expected at the writing of the proposal. In addition, the results we have obtained so far and the new directions for research that emerge are even more promising and will have to be pursued beyond the end of the COLUMBUS contract. This line of research is expected to significantly contribute to the fundamentals of platform-based design.*

2.3 Meta-modeling techniques for heterogeneous modeling

Results. The research track presented in section 2.2 allows us to handle MoCCs that are “simple enough” to formulate so they can be entirely captured by a high-level mathematical language. With the advantage that non-trivial theorems can be derived, this is the interest of the approach developed in section 2.2.

However, not every MoCC is simple enough to be captured and analysed in this way. For example, the “more than 20 different semantics of Statecharts” won’t be distinguished by high-level mathematical features, but rather by little details in the way actions, transitions, micro-steps, steps, and super-steps, are constructed to form the reactions. Such variants are much better described by means of detailed expansions of each MoCC in terms of a common, lower level but more tunable, common semantic basis. This is referred to as the technique of *design patterns*. It is developed in the context of UML at the VU group. It consists in expanding each semantic concept of a Domain Specific Modeling Language (DSML) as a detailed UML model whose constituting elements are elementary enough to be supported by a common sense semantics. This is a convenient approach if the difficulty of the considered semantics lies in the combinatorial complexity of the details of the different concepts, not in their high-level mathematical nature. This approach is implemented in the GME tool developed at VU since 1999 (<http://www.isis.vanderbilt.edu/Projects/gme/default.html>).

This aspect of the problem has been the subject of a joint VU-INRIA investigation performed by Ethan Jackson, from VU. The problem considered was to see how meta-modeling techniques can be used to assist the Signal programmer¹ in developing “safe” programs, i.e., programs that will not be problematic from the point of view of synchronization and clocks. This is an interesting application since it concerns a formalism that is quite different from the more classical imperative, state-based, formalisms such as the Statecharts. To this end, a new DSML with two different aspects has been proposed, see [8]. One aspect concerns data-flow specifications, and the other one addresses synchronization constraints using Signal clocks.

Perspectives. Thanks to compositionality of the Signal language as well as the services included in the Polychrony tool for Signal, specifications involving the two different aspects can be composed and jointly analyzed. The next question is: given specifications derived using *different* tools or notations, e.g., Signal and Statecharts, how can GME be used to facilitate the integration of these specifications?

2.4 References

[1] A. Benveniste, L. Carloni, P. Caspi, A. Sangiovanni-Vincentelli. Heterogeneous Reactive Systems Modeling and Correct-by-Construction Deployment. In Proc. of EMSOFT’2003 conference. LNCS 2855, R. Alur, I. Lee Eds., 35—50, 2003.

¹ Signal is a synchronous language that has been invented and developed at INRIA-Rennes by Paul Le Guernic and Albert Benveniste. A commercial version of it exists and has been developed and is marketed by TNI-Valiosys, a company located in France.

January 7, 2004

- [2] A. Benveniste, B. Caillaud, L. Carloni, P. Caspi, A. Sangiovanni-Vincentelli. Heterogeneous Reactive Systems Modeling and Correct-by-Construction Deployment: extended case. To appear in post-publication of the FMCO'2003 conference, LNCS.
- [3] B. Caillaud, D. Potop-Butucaru, and A. Benveniste. Erratum to : “A. Benveniste, B. Caillaud, P. Le Guernic. Compositionality in data-flow dychronous languages, specification and distributed code generation. Information and Computation 163, 125—171 (2000)”. September 2003. <http://www.irisa.fr/prive/Benoit.Caillaud/erratum-ic2003.pdf> .
- [4] D. Potop-Butucaru, B. Caillaud, and A. Benveniste. Concurrency in Synchronous systems. INRIA Research Report, Jan. 2004. Also submitted for publication. Dec. 2003.
- [5] L. Carloni and A. Sangiovanni-Vincentelli. A Formal Modeling Framework for Deploying Synchronous Designs on Distributed Architectures. First International Workshop on Formal Methods for Globally Asynchronous Locally Synchronous Architectures (FMGALS 2003).
- [6] P. Le Maigat. Mesochronous pomsets. COLUMBUS Draft.
<http://www.irisa.fr/s4/docs/lemaigat2003.pdf>
- [7] Neema, S., Sztipanovits, J., Karsai, G, Butts, K.: “Constraint-Based Design Space Exploration and Model Synthesis,” Proc. of EMSOFT'2003, Philadelphia, PA, October 2003
- [8] Jackson, E. An evolving Signal programming environment. Draft, annex to DMM1.

3. MEETINGS, VISITS, EXCHANGES OF PEOPLE

3.1 Meetings

The following meetings relevant to the WPTA were held during the considered period.

- COLUMBUS WPTA technical meeting, October 10th, Philadelphia. Since this was a very important meeting for WPTA, we enclose the minutes to this report. The INRIA-PARADES-UCB-VU-VERIMAG presentations have resulted in several important insights in the relationships among the individual efforts. The VU-INRIA presentations and the discussions have identified the following issues:
 - What are the best ways to utilize graphical representations?
 - Can aspect oriented techniques provide insight?
 - How well can errors be caught with static semantic constraints?

The Signal community also has important concerns:

- Can the Signal debugging cycle be improved?
- How must the Signal compiler be modified to accommodate new features?
- How should Signal compiler structures be presented?

Finally, if VU DSML technology is to be used for system design then:

- Can other MOCs (models of computation) share the graphical notations that have been applied to Signal?
- If so, can this aid in the heterogeneous composition of Signal to DSMLs with differing MOCs?
- Can composition occur at the meta-model level?

These questions have generated such a strong interest in the participants that it was decided that a direct visit and interaction between the VU and INRIA groups will be necessary.

- WPTA technical meetings at PARADES with UCB (Alessandro Pinto and Luca Carloni)

3.2 Visits

- Ethan Jackson, from VU, has visited the group of Benoît Caillaud at INRIA-Rennes – this was mentioned as an action item in the minutes of the 10th October meeting, see below. In Rennes, Ethan also interacted closely with Thierry Gautier and Loic Besnard, who are key members of the Signal research & development team.
- Alberto Sangiovanni Vincentelli has been travelling back and forth from UCB to PARADES on a regular basis. Alessandro Pinto spent 15 days between December and January at PARADES.

4. MINUTES OF THE 10TH OCTOBER MEETING

Participants

- Vanderbilt Univ: Janos Sztipanovits, Ethan Jackson, Sandeep Neema
- PARADES: Alberto Sangiovanni-Vincentelli
- UCB: Luca Carloni
- INRIA : Albert Benveniste, Benoît Caillaud
- Verimag : Paul Caspi.

Janos

[SLIDES AVAILABLE]

Metamodeling for domain specific modeling languages (DSML)

[provides generic SW engineering tools suited to DSML]

Concrete syntax \leftrightarrow abstract syntax \leftrightarrow semantic domain (structural semantics only).

Alternative is modeling by mapping to a pre-defined formalism (abstract syntax, structural semantics and/or behavioral semantics). Ex: semantics of HSF (Hierarchical Signal Flow) via SDF model (Synchronous Data Flow).

DSML specification: abstract syntax can be parameterized by using MOF as meta-model. GME defines map concrete \rightarrow abstract syntax. Allows to generate easily concrete/abstract syntax and their mappings, for new DSML. What about semantics of MOF? MOF modeling is a meta-circular specification, i.e., it gets its semantics from a small number of primitive constructs, which have an “understood” semantics (alternatively a formal TLA semantics can be considered). Other constructs are formally decomposed into these primitives, and this is specified by MOF itself.

Behaviors: have a “universal semantics” for flat Mealy automata using TLA. All behavioral semantics are then obtained by translating into this “universal” model for automata. This mapping is performed again using MOF as graph manipulations. Allows to automatically generate the behavioral translation of a DSML into the universal basic model of automata.

[This approach is very similar to the expansion of different models of computation into Signal; the latter approach is used in architecture modeling tools, e.g., at TNI-Valiosys and in our group; same with Lustre or Scade is CRISYS.]

[Suggestion for COLUMBUS action: for glueing different formalisms, instead of making the connection by do-it-yourself description in terms of the basic universal model, try to use the higher-level tag approach (relate different MoC by tag sets manipulations); by doing so we inherit the available mathematical results; prepare a library of basic tag types plus a library of maps between tags.]

[Alberto: this approach is intended to give tools and to formally define the notions of abstract syntax and transformations btw different languages using the abstract syntax as a basic intellectual method. Useful for designing novel design languages that could be anchored over solid semantic foundations.]

Running an example – Sandeep Neema

Automobile sector as an illustration. Mapping a {Stateflow + hardware} model to a distributed architecture (Stateflow used for SW spec). Code generation includes C code generation for components plus system code integration. Showing Stateflow2SF-C generation (model of C code). Pretty complete translation from Stateflow to C by closely following the documentation rules.

[DEMO]

component models are executable but topological models are not, they just provide structural info. Multiple references are not allowed in this version (unable to replicate components for distributed deployment – e.g. for redundancy).

January 7, 2004

Detailed description of the Stateflow2SFC translation using the metamodeling approach – SFC is a restricted C to represent the running of simple embedded automaton C code. Both formalisms are specified in UML. (MOF). Translation scripts are specified by means of *design patterns* (graph transformations between the different diagrams). These pattern mimic the Matlab documentation for Stateflow. Yields a flat automaton-like model that can be animated and given for model checking – low level modeling since the target model does not accommodate abstractions easily.

[The code generated via design patterns is not *optimized* code generation, but is *structure preserving* translation. To obtain optimized code, the idea would be to encode optimization using the same technique – not obvious that deep transformations can be addressed this way (think of transformations performed e.g., in the compilation of synchronous languages). Opinions differ...]

Application to the model integration of Signal – Ethan Jackson

Experiment to apply the GME tool to Signal. Aot of Signal meaning is captured through dataflow equations that do not fit so much the graphical specification using design patterns. So this was an interesting challenge.

Albert & Benoît

RT-Builder – Albert

[TNI SLIDES]

[DEMO] on a mixed CAN-FLEXRAY application example; mixed functional-nonfunctional modeling and simulation. Built on the top of Signal. Key technical advantages in building on the top of synchronous language is 1/ to have parallelism captured, and 2/ to cleanly separate and still handle jointly logical and physical time. Greatly simplified the design of the tool.

New progresses in desynchronization – Benoît

Weak-endochrony, weak-isochrony. Puts into the center issues of concurrency and confluence between successive reactions. An important improvement over previous results.

Alberto & Luca

Network platforms – ASV

Client-server point of view for the relation btw functions and network as a computing resource. How does the network respond to queries? Time-synchronization is one important feature (combined with dynamic scope for interaction); but its need is dynamically varying and is not static – a new feature wrt that we have been considering. But we (may) have a nice background to address this. Additional item is robustness to disruption of the network.

Status of Metropolis – ASV

Importance of considering different levels of abstraction and the binding btw these; keep both levels of refinement simultaneously visible; refinement checked with SPIN. Functions, architectures, refinements, and platforms. One of the refinement steps is to replace unbounded FIFOs by bounded ones; making the refinement discovery of bugs in the design of the application in video transmission. In the spec heterogeneous MoC is accepted: asynch is the default choice, more synchro must be specified by coordination constraints explicitly. Both functional and nonfunctional aspects are considered. Regarding nonfunctional aspects, *quantities* can be attached to components. Quantities can be time, power, etc., but also logic. Under progress.

Summary and priority items for the last year of Columbus

Important points

- Showing new and deeper cooperations btw EU and US.
- Prepare arguments for Columbus2 by showing news deliveries (promises for the future will easily follow – for Columbus2).
- Improve ties wit Janos in any possible way.

Some ideas

January 7, 2004

- Use Signal as a benchmark for Janos'GME tool: how to show aspects {dataflow, causalities, clocks} to the designer to guide for safe and easily debuggable program. Have Ethan visiting Rennes. Consider getting Janos'tools in Rennes.
- How can some theoretical results already obtained get used into prototype tools? Which ones?
- Considering the introduction of concurrency in Janos'meta-modeling framework, quite different from dealing with automata (or TLA).

5. ANNEXE

The following document is considered as the central one, and is therefore integral part of this report:

- Extended version of Ref. [1] above, published as both UCB research report UCB/ERL M0323 and IRISA Res. Rep. 1549. This last document is in particular downloadable from url http://www.irisa.fr/sigma2/benveniste/pub/PI_2003_emsoft_techReport.pdf