Model Based Architecting and Construction of Embedded Systems (ACES-MB 2011)

Stefan Van Baelen[1], Sébastien Gérard[2], Ileana Ober[3], Thomas Weigert[4], Huascar Espinoza[5], and Iulian Ober[6]

[1] KU Leuven - DistriNet, Belgium, Stefan.VanBaelen@cs.kuleuven.be
[2] CEA - LIST, France, Sebastien.Gerard@cea.fr
[3] University of Toulouse - IRIT, France, Ileana.Ober@irit.fr
[4] Missouri University of Science and Technology, USA, weigert@mst.edu
[5] Tecnalia, Spain, Huascar.Espinoza@tecnalia.com
[6] University of Toulouse - IRIT, France, Iulian.Ober@irit.fr

Abstract. The fourth ACES-MB workshop brought together researchers and practitioners interested in model-based software engineering for real-time embedded systems, with a particular focus on the use of models for architecture description and domain-specific design, and for capturing non-functional constraints. Six presenters proposed contributions on a systematic transition from systems modeling using SysML to software modeling using UML, verification of initial architecture models against requirements using consistency checking techniques, techniques to check the validity of refinement relation between abstraction levels, constructing rapid prototypes to assess the behavioral design characteristics, new abstraction layers to describe operating system and mixed-signal issues, and a model-driven approach for parallelizing software programs. In addition, a lively group discussion tackled these issues in further detail. This report presents an overview of the presentations and fruitful discussions that took place during the ACES-MB 2011 workshop.

1 Introduction

The development of embedded systems with real-time and other critical constraints raises distinctive problems. In particular, development teams have to make very specific architectural choices and handle key non-functional constraints related to, for example, real-time deadlines and to platform parameters like energy consumption or memory footprint. The last few years have seen an increased interest in using model-based engineering (MBE) techniques to capture dedicated architectural and non-functional information in precise (and even formal) domain-specific models in a layered construction of systems.

MBE techniques are interesting and promising because they allow to capture dedicated architectural and non-functional information in precise (and even formal) domain-specific models, and they support a layered construction of systems, in which the (platform independent) functional aspects are kept separate from architectural and non-functional (platform specific) aspects, where the final system is obtained by combining these aspects later using model transformations.

The fourth workshop on *Model Based Architecting and Construction of Embedded Systems* (ACES-MB 2011) brought together researchers and practitioners interested in all aspects of model-based software engineering for real-time embedded systems. The participants discussed this subject at different levels, from requirements specifications, model specification languages and analysis techniques, embedded systems product lines, model synthesis, to model based verification and validation.

2 Workshop Contributions

6 full papers had been accepted for the workshop, see the ACES-MB 2011 workshop proceedings [1]. A synopsis of each presentation is given below. Extended versions of articles [6] and [4] are included in this MoDELS 2011 workshop reader.

[2] addresses the integration of system modeling and software modeling for embedded systems, which are software intensive systems that consist of both hardware and software components. A systems modeling approach to create structural and behavioral models of the total system using SysML, and the systematic transition to software modeling using UML is been described.

[3] states that designing cyber-physical systems is a challenge originating from the multidisciplinary and mixed-signal requirements. In order to handle this challenge, many design languages have been developed, but none is able to connect different application domains adequately. A new system based view for cyber-physical system design is been proposed, which can be easily adapted by MARTE or SysML as it uses a model based design technique. Instead of defining another UML profile, the paper presents an intuitive idea for the development of cyber-physical systems by refinement, and introduces new abstraction layers that help to describe operating system and mixed-signal issues. Using new abstraction layers, it is now possible to support all views of the platform based design by using one consistent language. The approach explicitly distinguishes between the physical system and the computational system.

[4] describes a model-driven approach that aids the developer in parallelizing a software program. Information from a dynamic call tree, data dependencies and domain knowledge from the developer are combined in a model transformation process. This process leads to a model of the application where it becomes obvious which parts of the application can be executed in parallel.

[5] states that during the development of software intensive systems, typically several models of this system are designed, representing the system structured by different concerns. While these approaches help to cope with complexity, the need of relating the models to one another arises. A major task is to keep model specifications consistent and traceable through special relations. The relation of interest for this paper is the refinement relation between abstraction levels. The paper describes a technique to check the validity of these refinement relations with respect to formal behavior and interface specifications of design items. For evaluation, the proposed refinement technique is being applied to an industrial example modeled with a contract-based methodology.

[6] argues that the electric and electronic architecture (EEA), which is built up during the concept phase of automotive electronics development, has fundamental impact on the success of a vehicle under development. The complexity of upcoming architectures requires novel approaches to support system architects during the design phase. The paper describes a model-based generic approach which allows verifying an EEA with regard to its requirements by using techniques of consistency checks during an early design phase, including the handling of incomplete models. In this case, it offers the possibility to automate consistency checks and facilitates an automatism for optimization and design space exploration to check different realization alternatives of an initial EEA. Automatic report generation of results serves for documentation.

[7] describes an approach for constructing rapid prototypes to assess the behavioral characteristics of real-time embedded software architecture designs. Starting with a software architecture design nominally developed using the concurrent object-oriented design method COMET, an executable Colored Petri Net (CPN) prototype of the software architecture is developed. This prototype allows an analyst or engineer to explore behavioral and performance properties of a software architecture design prior to implementation. This approach is suitable both for the engineering team developing the software architecture as well as independent assessors responsible for oversight of the architectural design.

3 Summary of the Workshop Discussions

The workshop was divided into 3 sessions: system and software development and modeling, system and software architecture, and a plenary discussion session. During the 2 technical sessions, specific discussions on each paper have been held. In the plenary discussion session, a group discussion was held on broader questions and recurring issues that were raised during the session presentations. The following integrates and summarizes the conclusions of the discussions.

Traceability

A discussion was being held about the usefulness of traceability between models at different abstraction levels. Traces can be defined between a variety of model elements, such as system components, events, functional elements, and non-functional descriptions. It is often easy to generate traceability, at least in a model-driven approach where traceability information can be generated during the model transformation execution. But it is hard to decide how to use traces in a meaningful manner and where to generate useful traces, since one can easily end up with a huge number of inter-model traces that are hard to be used efficiently afterwards. Therefore methodological guidance is needed on what and where to trace. Semantic information could be added to the traces (e.g., by labelling traces) in order to generate semantically rich traces to increase the usability of the tracebility information.

Ontologies, metamodels and UML profiles

For expressing domain concepts and dependencies, domain experts can ei-

ther build ontologies using dedicated ontology languages, develop domain-specific modeling languages (DSMLs), defining the metamodels and mappings (transformations) between the metamodels, or define UML profiles, defining the set of annotations for labelling and enriching UML elements). Each of these approaches can have their specific benefits in a particular situation. Although the definition of a DSML is not always straightforward, the actual definition, extension and usage of a metamodel is usually meant only for domain experts, and not for application engineers.

Domain-specific languages for the automotive industry
The acceptance of EAST-ADL, a domain-specific language defined specifically for describing automotive applications, is very low in the automotive sector while AUTOSAR instead is heavily used. One of the reasons could be the inappropriateness of the concrete syntax used by EAST-ADL. One could think on building a new concrete syntax on top of the existing abstract syntax, but it is not clear whether it is only the issue of concrete vs. abstract syntax, or whether the abstract syntax of EAST-ADL is not sufficient.

Acknowledgements

This workshop was supported by the IST-004527 ARTIST2 Network of Excellence on Embedded Systems Design (http://www.artist-embedded.org), the research project EUREKA-ITEA EVOLVE (Evolutionary Validation, Verification and Certification, http://www.evolve-itea.org), the research project EUREKA-ITEA VERDE (Validation-driven design for component-based architectures, http://www.itea-verde.org), and the research project EUREKA-ITEA OPEES (Open Platform for the Engineering of Embedded Systems, http://www.opees.org).

References

1. Van Baelen, S., Gérard, S., Ober, I., Weigert, T., Espinoza, H., Ober, I., eds.: Fourth International Workshop on Model Based Architecting and Construction of Embedded Systems. CEUR Workshop Proceedings Vol.795, CEUR, Aachen, Germany (2011)
2. Gomaa, H.: Towards integrated system and software modeling for embedded systems. In: [1]. pp. 9–22
3. Slomka, F., Kollmann, S., Moser, S., Kempf, K.: A multidisciplinary design methodology for cyber-physical systems. In: [1]. pp. 23–37
4. Sackmann, M., Ebraert, P., Janssens, D.: A model-driven approach for software parallelization. In: [1]. pp. 39–53
5. Weber, R., Gezgin, T., Girod, M.: A refinement checking technique for contract-based architecture designs. In: [1]. pp. 55–69
6. Adler, N., Graff, P., Müller-Glaser, K.D.: Model-based consistency checks of electric and electronic architectures against requirements. In: [1]. pp. 71–83
7. Pettit IV, R.G., Gomaa, H., Fant, J.S.: Modeling and prototyping of real-time embedded software architectural designs with colored petri nets. In: [1]. pp. 85–98