Hindawi Publishing Corporation
EURASIP Journal on Embedded Systems
Volume 2008, Article ID 376920, 2 pages
doi:10.1155/2008/376920

Editorial

Selected Papers from SLA++P 07 and 08 Model-Driven High-Level Programming of Embedded Systems

Florence Maraninchi,[1] Michael Mendler,[2] Marc Pouzet,[3] Alain Girault,[4] and Eric Rutten[4]

[1] *VERIMAG Laboratory, 38610 Gieres, France*
[2] *University of Bamberg, 96045 Bamberg, Germany*
[3] *Laboratoire de Recherche en Informatique (LRI), Université Paris-Sud 11, 91405 Orsay Cedex, France*
[4] *INRIA Grenoble - Rhône-Alpes, 38334 Saint Ismier Cedex, France*

Correspondence should be addressed to Florence Maraninchi, florence.maraninchi@imag.fr

Received 31 December 2008; Accepted 31 December 2008

Model-based high-level programming of embedded systems has become a reality in the automotive and avionics industries. These industries place high demands on the efficiency and maintainability of the design process as well as on the performance and functional correctness of embedded components. These goals are hard to reconcile in the face of the increasing complexity of embedded applications and target architectures. Research efforts towards meeting these goals have brought about a variety of high-level engineering design languages, tools, and methodologies. Their strength resides in clean behavioral models with strong semantical foundations providing a rigorous way to go from a high-level description to mathematically certifiable executable code.

The most successful representatives of this trend of putting logic and mathematics behind design automation in embedded systems are synchronous languages; they have been receiving increasing attention in industry ever since they emerged in the 80s. Lustre, Esterel, and Signal are now widely and successfully used to program real-time and safety critical applications, from nuclear power plant management layer to Airbus air flight control systems. Their recent successes in the automatic control industry highlight the benefits of formal verification and automatic code generation from high-level models.

Model-based programming is making its way in other fields of software engineering too. Strong interest is emerging in component programming for large-scale embedded systems, in the link between simulation tools and compiler tools, in languages for describing the system and its environment, integrated tools for both compilation and simulation of more general models of communication and coordination, and so forth. The impact of such unifying methodologies will depend on the extent to which it will be possible to maintain the high degree of predictability and verifiability of system behavior that is the strength of the classic synchronous world.

List of Published Papers

This special issue features five very interesting papers. The first paper, "Lutin: a language for specifying and executing reactive scenarios," is by P. Raymond et al. It introduces the Lutin language, which targets the description and the execution of constrained random scenarios for reactive systems. It does so by allowing the user to express, in a Lustre-like dataflow style, constraints on input/output relations. The language constructs are inspired by regular expressions and process algebra.

The second paper, "Compilation and worst-case reaction time analysis for multithreaded Esterel processing," is by R. Von Hanxleden et al. It presents the compiling method used for Esterel programs onto the Kiel Esterel Processor (KEP), a multithreaded reactive architecture equipped with a dedicated instruction set to handle the Esterel features. On top of providing very efficient code, it is predictable, which allows the computation of the Worst Case Reaction Time (WCRT) of Esterel programs, an essential feature for real-time systems.

The third paper, "Formal analysis tools for the synchronous aspect language Larissa," is by D. Stauch. It presents two tools for the formal analysis of the aspect language Larissa, which extends the Argos synchronous language. The first tool allows the combination of design-by-contract with Larissa aspects. The second tool allows to weave aspects in a less conflict-prone manner, therefore allowing the static detection of remaining conflicts statically.

The fourth paper, "Embedded systems programming: accessing databases from Esterel," is by G. Luettgen and D. White. It presents two Application Programming Interfaces (APIs) which enable the use of relational databases inside Esterel programs. The first API is dedicated to database requests that can be answered very fast, and hence that complies to the synchrony hypothesis, while the second API is dedicated to database requests that must be handled asynchronously thanks to the external task mechanism of Esterel.

The fifth and final paper, "SoC Design Approach using Convertibility Verification," is by R. Sinha et al. It addresses the compositional design of systems on chip from verified components, and particularly the issue of protocol converters enabling the matching of different components. Convertibility is verified using Kripke structures, model checking of ACTL temporal logic, and a tableau-base converter generation algorithm (Bamberg, Grenoble, and Paris, April 10th, 2009.).

Florence Maraninchi
Michael Mendler
Marc Pouzet
Alain Girault
Eric Rutten