Quantitative Aspects of Embedded Systems

—Executive Summary Dagstuhl Seminar 07101—

March 5–9, 2007

Organizers:
Boudewijn Haverkort, University of Twente, the Netherlands
Joost-Pieter Katoen, RWTH Aachen University, Germany
Lothar Thiele, ETH Zurich, Switzerland

August 23, 2007

Embedded software systems

Embedded software controls the core functionality of many systems: it controls telephone switches and satellites, drives the climate control in our offices and cars, runs pacemakers, is at the heart of our power plants, and makes our cars and TVs work properly. As such systems are massively encroaching on daily life, our reliance on embedded software is growing rapidly. But, how justifiable is this reliance?

Whereas traditional software has a rather transformational nature mapping input data onto output data, *embedded software is different* in many respects. Most importantly, embedded software is subject to complex and permanent interactions with their—mostly physical—environment via sensors and actuators. Typically, software in embedded systems does not terminate and interaction usually takes place with multiple concurrent processes at the same time. Reactions to the stimuli provided by the environment should be prompt (timeliness or responsiveness), i.e., the software has to "keep up" with the speed of the processes with which it interacts.

Furthermore, characteristic for embedded systems is that they have to meet a multitude of quantitative constraints. These constraints involve the resources that a system may use (computation resources, power consumption, memory usage, communication bandwidth, costs, etc.), assumptions about the environment in which it operates (task arrival rates, task sizes), requirements on the services that the system has to provide (timing constraints, performance, response time) and requirements of the continuity with which these services are delivered (availability, dependability, fault tolerancy, etc.).

The observed difference between traditional software and embedded software has recently led to initiatives for a variety of dedicated international conferences and journals. Also, in various countries research institutes on embedded systems have been set up with a strong industrial cooperation, for instance, in Denmark (CISS), the Netherlands (ESI), and the US (CHESS).

Dagstuhl Seminar Proceedings 07101
Quantitative Aspects of Embedded Systems
http://drops.dagstuhl.de/opus/volltexte/2007/1137

A lack of quantitative assessment

Despite the importance of the quantitative constraints for the well-operation of embedded software systems, the proper assessment of cost, resources, performance, dependability, robustness, etc., often comes as an afterthought. It is rather common for embedded software to be fully designed and functionally tested before any attempt is undertaken to determine its performance, dependability or resource-usage characteristics. One of the main reasons for this situation is that well-developed and rigorous evaluation techniques for non-functional, i.e., quantitative system aspects have not become an integral part of standard software engineering practice. This undesirable situation has led to the increased interest by embedded software researchers to extend the usual functional specification and properties with a set of "performance indices", e.g., stated in terms of costs, timeliness, speed and the like, and constraints on these indices. Also in industry, a growing interest in assessing non-functional aspects of embedded systems as early as possible in the system design life cycle can be witnessed.

Where are we going?

Model-Driven Development (MDD) is a new software development technique in which the primary software artifact is a model. Ideally, the MDD technique allows engineers to (graphically) model the requirements, behaviour and functionality of computer-based systems. The design is iteratively analysed, validated, and tested throughout the development process, and automatically generated production-quality code can be output in a variety of languages.

Existing MDD tools for embedded systems are rather sophisticated in handling functional requirements but their treatment of quantitative constraints is still in development. Although methods for verification of real-time system designs, using for instance timed automata, are being developed, these methods are not yet mature enough for dealing with larger industrial embedded systems. Hence, MDD will not realise its full potential in the embedded systems area unless the ability to handle quantitative properties is drastically improved.

In contrast to the situation in the design of embedded software systems, in the design of computer-communication systems, **quantitative methods** to determine the quality of the system, expressed in terms of throughput or response time, have been used for a long time. Next to methods from classical queueing theory and discrete-event simulation, recently the use of analytical/numerical methods for evaluating complex systems has become more widespread. This has lead to methods and techniques to specify complex system behaviour using formal methods (like Petri nets, process algebra) enhanced with time and probabilities. Subsequently, appropriate models are generated from these high-level models, which, after numerical analysis, provide detailed insight in performance and dependability measures of interest.

Over the last, say, 5 years, very good progress has been made in pairing the above quantitative techniques to techniques known from the verification area, esp. model checking of properties specified in logics like CSL (an extension of CTL with stochastic time). This has led to model checking algorithms and tools for Markovian models of system. The state-of-the-art, however, is still such that expert knowledge is required to use these techniques, hence, large-scale application in embedded software system design and implementation is still a dream rather than a reality.

Of course, also known quantitative techniques from the area of real-time systems (classical

ones, such as EDF, or more advanced compositional ones), or methods known from network calculus (originally developed for dimensioning communication networks at a high level of abstraction), data flow graphs, and so on, can, and probably should be used as part of the embedded system design.

What is clear, though, is that all of the above techniques can only be used during the design of embedded systems after appropriate adaptation and embedding in a design trajectory, e.g., based on MMD.

Furthermore, were each of the above mentioned approaches has its strengths and weaknesses, an important first task is to map these strength and weaknesses (applicability, scope, modelling power, costs of evaluation, etc.). A second and more challenging question is then how to combine or integrate these methods. Such questions can only be answered when key researchers for these various approaches come together and exchange and discuss their ideas.

Seminar goal

Given the above considerations, the goal of this Dagstuhl seminar has been to bring together experts in the areas of embedded software design and implementation, model-based analysis of quantitative system aspects, and researchers working on extending all kinds of formal (design and analysis) methods with quantitative system aspects. These three areas are clearly well-related in the context of embedded systems, but have not been addressed as such in the past, as they have been worked upon in different communities. Thus, the seminar will lay bridges between these three areas, so that knowledge and experience can be shared, transferred and, ultimately, be generated.

Seminar results

The results of the seminar can be classified in four areas:

- Three tutorials have been presented, that helped in bringing together the variety of disciplines involved in model-based embedded (software) system design:
  - "Worst-case estimation methods" by Kim Larsen and Reinhard Wilhelm;
  - "Stochastic model checking" by Holger Hermanns;
  - "Model-based design for embedded systems" by Pieter Mosterman.
- 24 regular talks across the whole spectrum of topics were presented.
- A discussion session was held about which ingedients are necessary for a (master) curriculum on embedded systems. Moderated by Holger Hermanns, Reinhard Wilhelm reported about the ARTIST embedded system curriculum design, and Gerard Smit reported about the initiative at the 3 technical universities in the Netherlands to start a joint embedded system curriculum between the three electrical engineering and the three computer science departments.
- At the first day of the seminar working groups were formed, that partly later split and joined, but that led to a number of peer working group meetings and working group reports. At the end of the seminars, three working groups reported on their

findings about "worst-case versus stochastic methods", "how good are formal methods for model-based design", and "performance measures other than time".

In all of the four mentioned areas, the sharing and transfer of ideas between researchers with different background and viewpoints played a foreground role. This role, as such, can be regarded an important result in itself (the process, rather than the product). Cooperations have been started, and mutual interest in each others areas, conferences and scientific communities has been aroused.