07241 Abstracts Collection
Tools for the Model-based Development of Certifiable, Dependable Systems
— Dagstuhl Seminar —

Michaela Huhn[1], Hardi Hungar[2] and Doron A. Peled[3]

[1] TU Braunschweig, DE
huhn@ips.cs.tu-bs.de
[2] OFFIS Oldenburg, DE
hungar@offis.de
[3] Bar-Ilan Univ. - Ramat-Gan, IL
doron.peled@gmail.com

Abstract. From June 10th to June 15th 2007, the Dagstuhl Seminar 07241 "Tools for the Model-based Development of Certifiable, Dependable Systems" was held in the International Conference and Research Center (IBFI), Schloss Dagstuhl. During the seminar, several participants presented their current research, and ongoing work and open problems were discussed. Abstracts of the presentations given during the seminar as well as abstracts of seminar results and ideas are put together in this paper. The first section describes the seminar topics and goals in general. Links to extended abstracts or full papers are provided, if available.

Keywords. Dependable systems, safety, security, certification, formal methods, modelling, verification, tools.

07241 Summary – Tools for the Model-based Development of Certifiable, Dependable Systems

This paper summarizes the objectives and structure of a seminar with the same title, held from June 10th to June 15th, 2007 at Schloss Dagstuhl, Germany.

Keywords: Dependable systems, safety, security, certification, formal methods, modelling, verification, tools

Joint work of: Hungar, Hardi; Lemke, Oliver; Huhn, Michaela

Full Paper: http://drops.dagstuhl.de/opus/volltexte/2008/1405

Case Study Railway Level Crossing

The Railway Level Crossing case study provides an opportunity to illustrate development processes and techniques on a small, but still realistic, stand-alone dependable system. The case study was taken from the OPRAIL project (2004-2006, sponsored by the bmbf).

Using the case study as a common basis we hope to focus discussions, inspire comparisons, and accelerate the awareness of agreed solutions, opposite approaches and open issues in the seminar.

Keywords: Safety-critical system, railway level crossing, model based design, formal methods

Joint work of: Hungar, Hardi; Lemke, Oliver; Huhn, Michaela; Zechner, Axel

Development Processes and Methods for Aircraft System and Equipment

Gert Döhmen (Airbus - Hamburg, D)

With A380, Airbus has made the step from federated avionics to a standardized data communication and controller design - namely "Integrated Modular Avionics" IMA. The talk explains why and how this step was technically taken on A380 and how this step has influenced the business model and development process for systems. It discussed also the need for more distributed architectures of IMA and the need to set up an appropriate development method and environment to evaluate and optimize such architectures.

A second part of the talk reported on a project which evaluates whether SysML can be used for the design concept of an avionics system.

Keywords: Integrated Modular Avionics, System Architecture Evaluation, System Design with SysML

Some Experiences About Adoption of Model-based Development for Dependable Systems in Industry

Alessandro Fantechi (University of Firenze, I)

In this talk we report some views on the adoption of a model-based development approach, as seen from an academic perspective in several years of collaboration with industry: we will go through the different choices of modelling notation and tools made by the few different companies involved in such collaboration, looking at the many factors, both technical and political, domain-dependent, internal or external to the company, that have driven such choices.

Executing ASM Models with CoreASM

Roozbeh Farahbod (Simon Fraser University, CA)

How can one cope with the notorious problem of establishing the correctness and completeness of abstract functional requirements in the design of control-intensive software systems prior to actually building the system? In this talk we explore the abstract state machine (ASM) paradigm as one possible solution. Combining common abstraction principles from computational logic and discrete mathematics, abstract state machines provide a universal model of computation and an effective instrument for analysing and reasoning about complex semantic properties of real-world systems. We also look into CoreASM as a novel ASM tool environment that makes ASM ground models executable on real machines. We present the CoreASM language and the general architecture of the CoreASM engine together with a high-level description of its extensibility mechanisms.

Keywords: CoreASM, Abstract State Machines, Specification Languages, Executable Specification

Timing Analyse in Safety Critical Applications

Nico Feiertag (Symtavision GmbH - Braunschweig, D)

In this talk I will give a short introduction to scheduling analysis and some key issues regarding safety critical environments. This will include work done with BMW and a study on the determinism of FlexRay.

Keywords: Scheduling, Qualification, FlexRay

Full Paper:
http://www.symtavision.com/download.html

Towards Integrating Model-Driven Development of Hard Real-Time Systems with Static Program Analyzers

Christian Ferdinand (AbsInt - Saarbrücken, D)

Keywords: Timing analysis, Ascet, aiT, StackAnalyzer

Joint work of: Ferdinand, Christian; Heckmann, Reinhold; Wolff, Hans-Jörg; Renz, Christian; Gupta, Manabendra; Parshin, Oleg; Wilhelm, Reinhard

Model-Based Design of Certifiable and Dependable Systems - The Automotive Perspective

Ulrich Freund (ETAS GmbH - Stuttgart, D)

This talk will show the relationship between model-based design, dependability and certification in automotive systems. A motivation w.r.t. Road-Safety will be given. This talk incorporates results from the FP6 STREP EASIS and examples from the tool ASCET

Keywords: Model-Based Design, Ontology, IEC 61508, ASCET

Dependability, Certification, and the Model-Driven Development of Advanced Softwareintensive Systems: Obstacles, Possibilities, and Challenges*

Holger Giese (Universität Paderborn, D)

Model-driven development can be employed to raise the level of abstraction and thus in turn allows us to better deal with problems of higher complexity. This talk will address at first the question which obstacles exists when you want to employ this idea for the development of dependable systems and their certification. Afterwards, we review which new possibilities are opened up by the model-driven development paradigm. We will in particular consider very complex softwareintensive systems with advanced capabilities such as multiple variants or online-reconfiguration. We refer here to a number of results which have been developed mainly in the context of an advanced railway system, in which the individual shuttles autonomously coordinate themselves. Finally, we will discuss some of the challenges which remain for the addressed class of advanced softwareintensive systems.

*This work was developed in the course of the Special Research Initiative 614 – Self-optimizing Concepts and Structures in Mechanical Engineering – University of Paderborn, and was published on its behalf and funded by the Deutsche Forschungsgemeinschaft.

Certification - A Challenge for Formal Methods

Hardi Hungar (OFFIS - Oldenburg, D)

Certification is about convincing an authority that a system, which has been constructed, is as safe and reliable as it is supposed to be. There are domain-specific standards, mostly derived from the IEC 61508, which detail the evidence to be provided. This consists in documentation of procedures, verification and validation activities, tools, workforce qualification and so on. This results in a proof of juridical quality of the fact that the development has been performed with adequate care so that one may reasonably assume that the system fits its purpose.

Though the current practice largely seems to achieve its goal - as can be seen in the low percentage of accidents being attributable to design flaws - there are strong arguments to look for improvements. On the one hand, the effort to achieve sufficient confidence is rather high. And on the other hand, formal methods seem to have matured to a state that even a mathematically rigorous proof might become achievable.

To do this constitutes a challenge for the formal methods community with many facets: Not only several sorts of formal arguments (concerning e.g. timing, function and fault probabilities, different design levels, software and hardware and so on) are called for, but also evidence for the trustworthiness will be required. If e.g. a model checker verifies a property, it either must itself be verified or produce a proof for its verdict which can be validated by other means.

To this end, existing approaches will have to be extended and combined into coherent, comprehensive methodologies. Case studies like the level crossing example may be used to demonstrate the role particular methods can play in such a picture.

Keywords: Certification, Standards, IEC 61508, Formal Methods, Proof Checking

The Dagstuhl Level Crossing: A Formalist's Perspective

Stephan Merz (INRIA - Nancy, F)

I present a development of the seminar case study using Lamport's specification language TLA$^+$. In contrast to other notations presented at the seminar, TLA$^+$ emphasizes abstraction through the use of an expressive language: everyday mathematics. I will argue that this helps a developer to concentrate on the essential aspects, gradually introducing detail along the way. In particular, I will start with high-level properties (or "requirements"), then introduce a model of regular system operation before considering the degraded mode of operation. Real-time aspects will be added at the end of the development, a further step could introduce models of the actual sensors and actuators found in the system. Such a stepwise approach can help a developer to question and document design decisions early on, before they get buried beneath large amount of detail in the models. TLA$^+$ is supported by a model checker, for which the case study poses no particular problem.

Our current work on tools for TLA$^+$ is centered around providing a proof environment that combines interactive and automatic deduction tools in a sound and coherent way. Because we do not expect proofs of complex systems or algorithms to be automatic, the tool environment is designed around a declarative, hierarchical proof language.

Model-based development of an axle counter in SCADE

Stefan Milius (Siemens - Braunschweig, D)

In this talk we report about a pilot project for a model-based approach to software development at Siemens Transportation Systems. In our project we use the modelling tool SCADE by Esterel Technologies. SCADE is intended to model synchronous embedded and safety critical software systems. Software is modelled using a combination of graphical representations of the synchronous data flow language LUSTRE and of synchronous state machines. A code generator then produces C-code from the models, which has to be integrated in the rest of the system via a (handwritten) wrapper. To support certification the code generator of the current release SCADE 5.1.1 is certified for DO-178B, Level A, a norm from avionics.

In our pilot project we currently use the pre release SCADE 6 P3 to model a component of the software of an axle counter. Axle counters are used for track vacancy detection in railway crossings and interlockings. In this talk we present our experiences with SCADE 6 so far, we indicate how SCADE fits into our development process, and we present the first results of our pilot project concerning modelling, test and integration.

Model-Based Design of the Communication System in an Integrated Architecture

Roman Obermaisser (TU Wien, A)

The DECOS integrated architecture supports the sharing of a single time- triggered network for multiple application subsystems. As its communication infrastructure each application subsystem possesses a virtual network with guaranteed temporal properties, which is realized on top of the physical time-triggered network. We present a solution for the model-based design of virtual networks in order to reduce development time and avoid design faults. A graphical modelling tool enables system engineers to create a virtual network model that captures all relevant properties for instantiating virtual networks in a specific application.

The virtual network model forms the input to a code generator, the output of which can be deployed on the target system together with the application code. The development process is exemplified in an automotive example, which exploits virtual networks for application subsystems derived from present-day automotive architectures.

Keywords: Model-based development, system architectures, real-time systems, dependability

Integrating timed verification in a UML-based development process - the Omega/IFx approach

Iulian Ober (IRIT - Toulouse, F)

In this talk we present our experience with integrating verification in a model-based development process for real-time systems. Our framework consists of a UML profile (Omega UML) and a simulation and model-checking toolbox (IFx). The profile includes concurrency and timing concepts and a designer-friendly property specification mechanism. The toolbox supports model simulation, property verification and some forms of automatic abstraction which aim at reducing verification complexity and improving scalabilty. We present one of the most realistic experiments conducted with this framework, the modelling of a significant part of the Ariane 5 launcher's Flight Program, and the verification of several properties concerning functionality and tasking architecture. We also tackle the "level crossing" case study proposed in this seminar, and we show some unexpected results obtained with the model checker, and some ways to correct the specification to improve safety.

Keywords: Model checking, timed automata, UML, safety, observers

Transformation based Assessment of Dependability

Andras Pataricza (Budapest Univ. of Technology & Economics, H)

Assessing the impacts of potential faults in a dependable system is a core problem both during the construction and certification phase of dependable systems.

The main paradigm if traditional design for dependability policies was the use of rough granular redundancy, like multiplied modular redundancy. Recently, the dominant trend is shifting towards the use of fine granular redundancy both for the reasons of cost and efficient availability policies based on reconfiguration.

However, the use of fine granular redundancy requires more care, than the traditional, rough granular approach, as it cannot rely any more on the strict correlation between functionality and system structure.

The core idea of the recent talk is an examination, how formal methods can be applied to model and evaluate the impacts of faults in order to complement the traditional (and still VERY hard) proof of correctness V&V methods with the evaluation of the faulty cases.

Model based dependability analysis is a prerequisite for creating efficient redundancy patterns in order to create and check a fault-tolerant architecture assuring the proper functioning of a critical application even in the case of the presence of some anticipated fault.

Moreover certification necessitates evidences of guarantees on the behaviour of the system in the case of faults.

The talk will address several phases of fault and impact modelling in a design for dependability work flow all based on the joint principle of qualitative abstractions describing the systems at the abstraction level of fault/error/failure modes.

In particular, fault modelling aims at the modelling of the anticipated set of technology related faults typically occurring in the different types of components. Our approach describes the fault mechanisms anticipated at the meta model level associated to component types, and generates the entire set of candidate mutations by means of a graph transformation based model transformer implemented in the form of the VIATRA tool.

Another main element in the analysis process is the assessment of the impacts of a fault by means of error propagation analysis. However, in general the analysis problem is practically infeasible to be carried out due to the huge model complexity originating in the large cardinality of the faulty mutation population.

Efficient spatial and temporal information compaction methods will be presented:

– Spatial compaction reduces the model dynamics to the propagation of different error modes by using predicate abstraction. Such models can be analysed, for instance by model checking.
– A more compact representation describes the sensitivity of the individual components to errors by means of relations between the input and output error modes. The resulting net corresponds to a discrete space constraint network.

The presentation will illustrate the ideas gained during former industrial projects in the field of railway control and biomedical engineering, and by the objectives of the ongoing DECOS, HIDENETS, SAFEDMI and DIANA EU sponsored projects.

Keywords: Modeling, model transformation, abstraction, model based evaluation, fault and error modeling

Analysis of Message Sequence Charts

Doron A. Peled (Bar-Ilan Univ. - Ramat-Gan, IL)

We present several algorithms and tools for the analysis of message sequence charts. The idea of checking for race conditions was suggested, analysed and implemented a decade ago. It reflects the ability to describe an MSC where two messages may appear one after the other, but there is no guarantee that this is the case. We define a new notion called Discord, where message in subsequent MSCs in an MSC graph (HMSC) may migrate because of the partial order to appear in out-of-order manner. We present complexity and algorithms for calculating discords. We suggest a programming methodology based on using HMSCs (or LSCs) as a specification formalism, and using the discord calculation for parallelising the system without violating given constraints.

Level A/SIL-4-Compliant Automated Tests and Static Analysis for Avionic and Railway Control Systems

Jan Peleska (Universität Bremen, D)

In this talk we will consider a - from the scientific perspective rather “tame” - model-based development scenario, which from our experience represents a rather realistic view of the state of the art for many suppliers of the railway and avionic domains. In this scenario, a UML2.0 model for the system to be developed has been created, where automated code generation could be performed from UML2.0 Statecharts, code frames have been generated for classes and their methods, but conditions and actions have not been formally specified (say, with OCL); instead, they have been directly coded using C/C++.

For this scenario we consider verification activities which are crucial to gain certification credit according to RTCA DO178B or CENELEC EN 50128: (1) It is discussed how appropriate assertions can be specified, in order to perform (semi-) automatic model validation. The objective of this step is to replace conventional informal software requirements and design reviews. (2) We describe our approach and tool support to model-based functional and code-based structural testing: Using constraint generators and solvers based on interval analysis and additional theories for special data types like bit vectors and strings, the tool generates test data for MC/DC structural coverage for the system under test (SUT). To this end, C++ code is translated into control flow graph (CFG) representations where statements are expressed in 3-address code. Atomic discrete and floating point variables, strings, compound data types involving structures and arrays, as well as pointers are associated with a memory model so that pointer arithmetic and several variants of type casts can be handled. If the reachability of an edge in the CFG can neither be proven (by constructing appropriate test data) nor disproven, user interactions are possible to guide the solver by introducing additional input constraints and suggesting appropriate execution paths. Since the code generator only creates code from Statecharts, while guard conditions and actions consist of hand-coded C++ code anyway, we advocate to test both the generated and the hand-written code according to the requirements of applicable standards, thereby avoiding the costs for the qualification of the Statechart code generator. The objective of this step is to automate and improve the quality of module and software integration tests, while avoiding the qualification of code generators. (3) It is shown how the algorithms and data structures needed for automated test case / test data generation are equally useful for a wide range of static analysis objectives, typically handled using abstract interpretation, such as the identification of potential run-time errors or the “light-weight” verification of post conditions and intermediate assertions: To this end, negated assertions and post conditions are mapped to auxiliary guard conditions extending the CFG, so that the verification of these properties corresponds to disproving the reachability of certain CFG edges. The objective of this step is to automate and improve the quality of code inspections and analyses.

As outlined above, all the automated verification steps described replace some activity which could - according to the applicable standards - also be performed manually and in an informal way. As a consequence, the implementation of these steps can already be considered as successful if they improve the quality and efficiency when compared with their manual equivalents. As a consequence we advocate light-weight verification support in the sense of "bug-finding support" without necessarily proving the absence of certain types of errors. This contrasts, for example, with the strategy of Cousot and others whose goal is to provide sound proof support for the absence of runtime errors and other verification objectives. Our somewhat "unambitious" attitude is based on the experience with industrial development projects in the avionic and railway domain, which results in the following assessment: (1) even for safety-critical applications wide-spectrum modelling support is needed, because the growing complexity of systems require a wide variety of data types and algorithmic features. As a consequence, a combination of UML2.0 and C/C++ is better suited for modelling many applications where more restrictive formalisms - while possibly offering stronger verification support - may lack expressive power to model all features of the system to be developed. (2) Many C/C++ constructs complicating automated verification (in particular, unions and pointer utilisation in combination with type casts) cannot simply be "banned" by coding standards, because their utilisation is necessary or at least extremely helpful when programming low-level software like device drivers. As a consequence, non-exhaustive bug-finding tools which do not restrict the preferred programming style have a fair chance of being accepted by software developers and will have a considerably higher error detection rate than the manual code inspections they replace.

Our work has been influenced by practical verification, validation and testing projects and consultancy with respect to certification activities performed by Verified Systems International since 1998 for the avionic and railway domains (Level-B and C systems for Airbus A318, A340, A480, SIL-4 interlocking and train control components, control software for the Shanghai MAGLEV levitation train developed by Siemens Transportation Systems).

Several of the scientific results presented here are based on the cooperation between the first author's research group at the University of Bremen and Martin Fränzle's team at the University of Oldenburg.

Keywords: Safety-critical systems, model-based development, UML2.0, test automation, static analysis, abstract interpretation

Joint work of: Peleska, Jan; Zahlten, Cornelia

Challenges in the railway domain: Is Model-based development a good answer?

Ralf Pinger (Siemens - Braunschweig, D)

Developing products for rail automation leads to several challenges for processes and methods. Firstly, we have to develop products on a very high safety level. For achieving this safety level a high amount of testing, reviewing and certification is needed to assure the confidence in the technical system. Secondly, typical life cycles of rail automation products last more than 25 years. With the change from relay technology to computer based systems we have to deal with another problem: the underlying hardware platform with its electronic devices will change during the life cycle time of the product. In this talk we will present some problems arising from railway products and provide some insights on how model based software development might help reducing those problems. Furthermore we give some suggestions for a tool framework on model based development.

Property-Based Development of Dependable Systems

Amir Pnueli (Courant Institute - New York, USA)

Reacting to the emphasis of the seminar on “model-based” approach, I play devil’s advocate and present the competing approach of “property-based” formal development.

We start with a general comparison of these two approaches to specification, where the major distinction is that a model-based specification is structured according to modules of the implementing system (intra-object view), while a property-based specification is structured according to the properties the system should satisfy (inter-object view).

We then review recent developments in automatic synthesis of running code (or design) from a property specification, given in LTL or PSL. Time permitting, we will illustrate the approach on the “Railway Level Crossing” case study.

Keywords: Specification, property-based, code synthesis

Mondex Case Study

Peter H. Schmitt (Universität Karlsruhe, D)

The Mondex Case study is still the most substantial contribution to the Grand Challenge repository. It has been the target of a number of formal verification efforts. Those efforts concentrated on correctness proofs for refinement steps of the specification in various specification formalisms using different verification tools. In this paper we report on a Javacard implementation of the Mondex protocol and on proving its correctness using the KeY tool. The safety properties to be proved are formalized in the Java Modeling Language and follow as closely as possible the concrete layer of the previous **Z** specification.

Keywords: Formal methods, verification, specification, Z specification language, JML theorm proving

Full Paper:
http://www.ubka.uni-karlsruhe.de/cgi-bin/psview?document=ira/2007/4

See also: Tonin, Isabel. Verifying the Mondex Case Study - The KeY Approach, Interner Bericht 2007-4 der Fakultät fuer Informatik, U.Karlsruhe, ira/2007/4

Modular Functional Descriptions

Bernhard Schätz (TU München, D)

The construction of reactive systems often requires the combination of different individual functionalities, thus leading to a complex overall behaviour. To achieve an efficient construction of reliable systems, a structured approach to the definition of the behaviour is needed. Here, functional modularization supports a separation of the overall functionality into individual functions as well as their combination to construct the intended behaviour, by using functional modules as basic paradigm together with conjunctive and disjunctive modular composition.

Keywords: Function, behaviour, module, composition, decomposition, refinement, verification

Verics: a model checker for real-time and multi-agent systems

Maciej Szreter (ICS PAS & University of Podlasie, PL)

A tool for symbolic verification of real-time and multi-agent will be presented. Symbolic model checking algorithms constituting the computational engine of the tool: Bounded and Unbounded Model Checking, will be described in more detail. Some case studies corresponding to several aspects of presented scenarios will be shown.

Keywords: Model checking tool, SAT-based verification, verification of multi-agent systems