Alpuente Frasnedo, M.; Joubert ., C.; Kowalewski, S.; Roveri, M. (2013). Formal methods for industrial critical systems, preface to the special section. Science of Computer Programming. 78(7):775-777. doi:10.1016/j.scico.2012.05.005.

Preface to the special issue on Formal Methods for Industrial Critical Systems (FMICS 2009 + FMICS 2010)[☆]

María Alpuente[a], Christophe Joubert[b,a], Stefan Kowalewski[c], Marco Roveri[d]

[a]*DSIC-ELP, Universitat Politècnica de València, Camino de Vera s/n, Apdo 22012, 46020 Valencia, Spain.*
[b]*Prodevelop, Plaza Don Juan de Villarrasa, 14 - 5, 46001 Valencia, Spain.*
[c]*Embedded Software Laboratory – RWTH-Aachen University, Ahornstr 55, 52074 Aachen, Germany*
[d]*Embedded Systems Unit – Fondazione Bruno Kessler, Via Sommarive 18, 38123 Povo (TN), Italy*

Abstract

This special issue contains improved versions of selected papers from the workshops on Formal Methods for Industrial Critical Systems (FMICS) held in Eindhoven, The Netherlands, in November 2009 and in Antwerp, Belgium, in September 2010. These were, respectively, the 14th and 15th of a series of international workshops organized by an open working group supported by ERCIM (European Research Consortium for Informatics and Mathematics) that promotes research in all aspects of formal methods (see details in http://www.inrialpes.fr/vasy/fmics/). The FMICS workshops that have produced this special issue considered papers describing original, previously unpublished research and not simultaneously submitted for publication elsewhere, and dealing with the following themes:

- Design, specification, code generation and testing based on formal methods.
- Methods, techniques and tools to support automated analysis, certification, debugging, learning, optimization and transformation of complex, distributed,

[☆]This work has been partially supported by the EU (FEDER) and the Spanish MEC TIN2010-21062-C02-02 project, MICINN INNCORPORA-PTQ program, and by Generalitat Valenciana, ref. PROMETEO2011/052.

Email addresses: alpuente@dsic.upv.es (María Alpuente), joubert@dsic.upv.es (Christophe Joubert), kowalewski@embedded.rwth-aachen.de (Stefan Kowalewski), roveri@fbk.eu (Marco Roveri)

real-time and embedded systems.

- Verification and validation methods that address shortcomings of existing methods with respect to their industrial applicability (e.g., scalability and usability issues).
- Tools for the development of formal design descriptions.
- Case studies and experience reports on industrial applications of formal methods, focusing on lessons learned or new research directions.
- Impact and costs of the adoption of formal methods.
- Application of formal methods in standardization and industrial forums.

The selected papers are the result of several evaluation steps. In response to the call for papers, FMICS 2009 received 24 papers and FMICS 2010 received 33 papers, with 10 and 14 accepted, respectively, which were published by Springer-Verlag in the series Lecture Notes in Computer Science (volumes 5825 [1] and 6371 [2]). Each paper was reviewed by at least three anonymous referees which provided full written evaluations. After the workshops, the authors of 10 papers were invited to submit extended journal versions to this special issue. These papers passed two review phases, and finally 7 were accepted to be included in the journal.

---

1. Overview of the Special Issue

The aim of the FMICS workshop series is to provide a forum for researchers who are interested in the development and application of formal methods in industry. In particular, these workshops bring together scientists and engineers who are active in the area of formal methods and are interested in exchanging their experiences in the industrial usage of these methods. These workshops also strive to promote research and development for the improvement of formal methods and tools for industrial applications.

The whole selection process was open to all FMICS themes; however, the final list of papers has a common focus on the automatic verification of systems. This witnesses the current concerns about the importance of automatic verification, which is on the one hand gaining more and more industrial application, especially in the field of interest of FMICS, that is, critical systems: model checking

and static analysis techniques are routinely applied in several industrial domains. On the other hand, application to real systems often stresses such verification techniques to their limits, requiring new insights and techniques for helping scalability of automatic verification to the size of the increasingly complex systems that more and more pervade our daily lives. The collection of papers gathered in this special issue is a good representative of the research carried out in this direction.

2. Selected Papers

The first article by Sami Evangelista and Lars M. Kristensen presents a collection of sound state-space partitioning algorithms for distributed-memory and disk-based state space generation and exploration. It presents a detailed study of heuristics to perform a partition of the state space when carrying on an explicit search using external memory to store the state space. The authors focus on minimizing the network traffic (in a distributed setting) or i/o operations (in external model checking) by minimizing the number of crossing transitions between partitions. The method described in the article starts from a single partition and refines the partitioning schema when a limit was exceeded. This research topic is very relevant. Although it has been studied extensively in the literature, the authors provide new insight by stressing the dynamic nature of their partition refinement.

In the second article, the authors Alwyn Goodloe and Cesar Munoz present a method for developing a compositional proof strategy that supports an iterative design process. The methodology helps to automate important proof steps including finding inductive invariants, and is applied to verification of two interacting communication protocols. The article describes how a protocol stack combining reliable and unreliable communication for remotely controlled aircrafts can be modeled in PVS. It explains how proof scripts support iterative design and how they can be maintained when the protocol changes. The scope is much wider than traditional previous proofs, considering a deeper protocol stack, and a combination of different protocols. The proof of the protocols within a context neatly reuses the correctness proofs for the isolated case. To this end, the authors develop a method to lift invariant proofs to a wider context, based on abstraction and projection of system traces back to component traces.

The next article by Jos Bacelar Almeida, Manuel Barbosa, Jorge Sousa Pinto, and Bárbara Vieira addresses the verification of a security property for a C function in the NaCl cryptographic library. The authors propose a sound and useful method to formalize and prove non-inference properties for real code. The article starts with a good explanation on how non-interference properties and other

security related properties can be proved. After an introduction of the theory, it is shown in detail how the theory can be applied to analyze and prove that a C function is correct for non-interference, as well as for its functionality. The explanation of the whole chain from theory to real life application is very valuable and shows how verification of non-inference properties can be achieved with off-the-shelf tools. This is an important step for software engineers developing security solutions.

The fourth article by Alessio Ferrari, Alessandro Fantechi, Gianluca Magnani, Daniele Grasso, and Matteo Tempestini describes an experience of integration of formal methods in the industrial life cycle by discussing the application of Stateflow/Simulink and Polyspace to the design of the automatic train protection (ATP) of Metro Rio. They target two kind of analysis: code generation process and run-time error removal. For the code generation process they rely on a subset of the Stateflow/Simulink language having a clean semantics. The Stateflow/Simulink model and associated tools are then used for model-based testing and code validation (via co-simulation of the models with the generated code). For run-time error analysis they rely on Polyspace (a tool based on static analysis and abstract interpretation). The approach is refined in two steps in order to improve the elimination of statements that could lead to run-time errors but not certainly as identified by Polyspace. Both analyses enable for the certification of the code generator and of the resulting code according to a proven-in-use strategy recommended by safety standards in the railway signaling area.

The fifth article by Radu Mateescu and Wendelin Serwe describes an interesting application of formal methods for supporting quantitative analysis of mutual exclusion protocols. In the paper several well known mutual exclusion protocols, like e.g. Peterson's or Dekker's, are considered. All the protocols are formalized in Lotos NT (a process algebraic language), while all the properties to be used for the validation of the protocols are formalized in MCL (a model logic that extends μ-calculus with regular expressions over transitions). All the verification analyses are carried out with the CADP verification toolbox (CADP is a successful state-of-the-art toolbox for the design and analysis of communication protocols). The results of this paper complement the thorough studies on formal verification of mutual exclusion protocols with formal approaches to the verification of their non-functional requirements.

The sixth article by Jörg Brauer, Andy King and Stefan Kowalewski describes an approach to performing static program analysis of machine-code. The main focus of the paper is in modelling and analyzing bounded integer computer arithmetic. To this purpose they considered the conjunction of two complementary

abstraction techniques, namely interval abstraction and linear congruences. The paper provides three main contributions. First, it describes how to automatically generate transformers for the two considered abstractions. Second, it provides techniques for the synthesis of branching conditions at machine-code level (a very challenging problem because differently from high-level languages, in machine-code branching decisions are not performed in a single atomic step). Finally, it provides automatic refinement techniques that enable to use information from one abstraction to refine the other. This work represents an important step in the verification of microcontroller-code aiming at improving its quality and correctness.

The last article by Alexei Iliasov, Elena Troubitsyna, Linas Laibinis, Alexander Romanovsky, Kimmo Varpaaniemi, Dubravka Ilic, and Timo Latvala discusses a formal approach to the development and refinement of mode-rich systems. First, a general approach based on Event-B is discussed, and then it is instantiated and applied to the design and development of the Attitude and Orbit Control System component of a satellite. The main contribution of the paper consists of a conceptual model of complex mode transitions and in its formalisation captured in a Mode Manager specified in modular Event B. The approach recognizes and addresses some fundamental complications. In particular, mode transitions of layered systems involve many components, and cannot happen instantaneously, due to the properties of electro-mechanical parts; and mode transitions can be interrupted while in progress, but still guarantees certain mode invariants. Design decomposition corresponds to B-refinement, which are checked formally with the Rodin platform. As a result, the final mode-transition system resulting from refinements is fully verified. This approach provided a means for fighting design and verification complexity.

Acknowledgments

We would like to thank all the authors, the members of the program committees and the external referees of the two workshops, and the reviewers of the journal versions for their hard work in reviewing papers. We also especially thank Bas vas Vlijmen for his valuable support in the whole revision and editing process. Finally, we would like to thank very specially Jan Bergstra, editor in chief of Science of Computer Programming.

References

[1] M. Alpuente, B. Cook, C. Joubert (Eds.). Formal Methods for Industrial Critical Systems (14th International Workshop, FMICS 2009, Eindhoven,

The Netherlands, November 2-3, 2009), in: Lecture Notes in Computer Science, vol. 4916, Springer, 2009.

[2] M. Roveri, S. Kowalewski (Eds.). Formal Methods for Industrial Critical Systems (15th International Workshop, FMICS 2010, Antwerp, Belgium, September 20-21, 2010), in: Lecture Notes in Computer Science, vol. 6371, Springer, 2010.

María Alpuente
Christophe Joubert
Stefan Kowalewski
Marco Roveri