Preface

Selected and extended papers from FACS 2016

The component-based software development approach has emerged as a promising paradigm to cope with the complexity of present-day software systems by bringing sound engineering principles into software engineering. However, many challenging conceptual and technological issues still remain in this area, theoretically as well as practically. Moreover, the advent of cloud computing, cyber-physical systems, and of the Internet of Things has brought to the fore new dimensions, such as quality of service, reconfiguration, and robustness to withstand inevitable faults, which require established concepts to be revisited and new ones to be developed in order to meet the opportunities offered by those architectures.

The program of FACS 2016 included three invited talks, from Holger Giese, Kung-Kiu Lau, and Franck Le Gall, and 14 research papers, including one tool paper, and 2 application and experience papers. The annals of the conference, containing revised versions of the papers, were published in volume 10231 of LNCS. Then the authors of 8 papers were selected and invited to submit to this special issue. After an extensive and rigorous reviewing process, in which each paper was reviewed by at least three reviewers, seven of them were selected to appear in this special issue and are summarized next.

Constrained Synthesis from Component Libraries, by Antonio Iannopollo, Stavros Tripakis, and Alberto Sangiovanni-Vincentelli, is dedicated to the synthesis problem of a specific component composition from component libraries, which is undecidable in a general setting. The authors consider a bounded version of this problem with the goal to synthetize a composition of components satisfying a given specification, while minimizing a composition cost which is based on individual costs of components. The counter example-guided induction synthesis paradigm (CEGIS) is used to elaborate the problem solution. The paper also reports on the developed tool and experimental results for two non-trivial case studies.

Ch*ecking multi-view consistency of discrete systems with respect to periodic sampling abstractions*, by Maria Pittou, Pete Manolios, Jan Reineke, and Stavros Tripakis, presents a method to test consistency between several models of a system obtained by periodic sampling of the system: each retains information about every n-th action of the system. The paper provides an algorithm to decide consistency of models (finite transitions systems or specific Büchi automata) that yields a canonical, i.e., the most general, model that merges all the initial models if they are consistent. The paper also includes an interesting discussion over the expressive power of transition systems with or without private variables, showing that private variables can be necessary when merging models.

pCSSL: a Stochastic Extension to MARTE/CCSL for Modeling Uncertainty in Cyber Physical Systems, by Dehui Dua, Ping Huang, Kaiqiang Jiang, and Frederic Mallet, explores the modeling, simulation, and verification of Cyber Physical Systems interacting with an uncertain physical environment. Relying on the standardized SysML/MARTE, the authors use pCCSL, a stochastic extension of the Clock Constraints Specification Language CCSL, to model the interaction between the discrete and deterministic control part of the system with the environment. Statistical model checking is then applied to explore the system behaviour and drive the refinement process. The approach is illustrated using energy usage strategies of a smart-building system.

Reasoning about Connectors Using Coq and Z3, by Xiyue Zhang, Weijiang Hong, Yi Li, and Meng Sun, presents a logical representation of REO connectors, building predicates encoding complex connectors from axioms defining basic channels and composition operators. The Coq theorem prover is

then used to prove properties of the connectors. When Coq fails, the authors search (bounded) counter-examples using the SMT engine Z3. The paper includes examples of proofs of equivalence, refinement, and of behavioural properties using Coq, and of counter-examples generation in Z3, on a number of non-trivial use-cases.

Checking Business Process Evolution, by Ajay Krishna, Pascal Poizat, and Gwen Salaün, presents a formal approach for checking the evolution relations of business process models expressed in BPMN standard. The model evolution expressed by a branching relation between two distinct evolutions of a same process, is concerned e.g. with refactoring or with feature insertion, at static time. The objective is to check e.g. if important properties of the original behaviour are preserved by the evolved process. The proposed approach is based on process algebras (LNT) to formalise the semantics of BPMN processes. This enables the use of the CADP toolset for the automated verification.

Unifying Modal Interface Theories and Compositional Input/Output Conformance Testing, by Lars Luthmann, Stephan Mennicke, and Malte Lochau, presents a theory extending the IOLTS formalism, and the I/O conformance relation, "ioco", with modal (may/must) capacities and interface automata specification model. The authors propose a formal foundation for I/O-conformance testing theory based on a modified version of Modal Interface Automata with Input Refusals (IR-MIA). Then they prove correctness and compositionality properties of the corresponding modal I/O-conformance relation, called modal-irioco, w. r. t. a collection of different operators on IR-MIA.

Hierarchical Featured State Machines, by Vanderson Hafemann Fragal, Adenilso Simao, and Mohammad Reza Mousavi, addresses the design of software product lines (SPLs) by defining a formalism and a specification language, Hierarchical Featured State Machines (HFSM), motivated by complex systems' design and analysis, the size of which makes them hard to build and to maintain. The article reports on a tool for editing and validating specifications to guarantee their correctness, and on a case study to illustrate its benefits.

Finally, our most sincere thanks go to all the people who have made this special issue possible. To the authors, who trusted the FACS conference to discuss and publish their work, agreed to write extended versions of their papers and later incorporated all the corrections and improvements required by a careful refereeing process. To the referees, who have donated their time and effort to guarantee the highest quality for each submission. Our special thanks to Jan Bergstra, Bas van Vlijmen and the editorial staff at Elsevier, for agreeing to publish this special issue as a volume of the Science of Computer Programming journal, and for all their help in bringing this special issue to publication.

March 2019

Olga Kouchnarenko, University of Franche-Comté, Besançon, France
Eric Madelaine, Inria, Université Côte d'Azur, Sophia Antipolis, France