Software Quality Journal (2019) 27:497–499
https://doi.org/10.1007/s11219-019-09447-4

Guest Editorial: Special issue on Testing Software and Systems

Husnu Yenigun[1] · Nina Yevtushenko[2,3] · Ana Rosa Cavalli[4]

Published online: 30 May 2019

Testing is one of the most frequently used techniques in practice to assure the quality and the reliability of software systems. It is used not only during the development of such systems but also during their operation. Over the last years, new testing technologies appeared for checking both functional and non-functional requirements of systems. Application areas of these technologies include but are not limited to communicating systems such as protocols, middleware, networks, web services, cloud computing systems, wireless applications, control systems, business information systems, embedded and real-time software, and software product lines.

Albeit the decades of research and practical experience in software testing, the underlying theory, methods and tools, industrial use, and in its systematic combined application with other verification techniques are still very challenging. The purpose of this special issue was to present relevant papers covering theoretical and practical aspects of testing and papers reporting high-quality research on advanced techniques and quality metrics for software testing were invited. Following an open call for the papers, 14 papers were selected to appear in this special issue, among 24 submissions received.

In "Fault Model Driven Testing from FSM with Symbolic Inputs," the authors Omer Nguena Timo, Alexandre Petrenko, and S. Ramesh, discuss the generation of tests which are complete for predefined fault models for specifications of systems. As specifications, they consider deterministic FSMs with symbolic inputs, which are predicates over input variables with possibly infinite domains.

The paper "Fault-Based Refinement-Testing for CSP" by Ana Cavalcanti and Adenilso Simao proposes an online test generation method for the selection of finite test sets for traces refinement from CSP models. The method uses the notion of a fault domain, which allows a test verdict to be reached after a finite number of test executions.

✉ Husnu Yenigun
yenigun@sabanciuniv.edu

[1] Sabanci University, Istanbul, Turkey

[2] Ivannikov Institute for System Programming of RAS, Moscow, Russia

[3] National Research University Higher School of Economics (HSE), Moscow, Russia

[4] SAMOVAR, CNRS, Telecom SudParis, Paris-Saclay University, Paris, France

Petra van den Bos, Ramon Janssen, and Joshua Moerman show how an n-complete test suite can be designed for labeled transition systems. In their paper titled "n-Complete Test Suites for IOCO," they adapt FSM-based test generation methods for labeled transition systems successfully by introducing a new way for distinguishing compatible states.

"Safety-complete Test Suites" by Wen-ling Huang, Sadik Özoguz, and Jan Peleska considers an existing test generation method and adapts it to construct test suites guaranteed to detect all violations of a given set of safety properties. This focus on the given safety properties reduces the test size, at the expense of not detecting erroneous behavior not related to the safety properties.

Tao Ma, Shaukat Ali, Tao Yue, and Maged Elaasar, in "Testing Self-Healing Cyber-Physical Systems under Uncertainty: A Fragility-Oriented Approach," consider self-healing cyber-physical systems, i.e., systems that can detect, diagnose, and recover from errors. In particular, they consider testing of self-healing functionalities under uncertainty, where they try to identify how likely the system fails at a state, which is a piece of information used to guide the testing process to focus on states with a high probability of failure.

Alexandre Petrenko, Florent Avellaneda, Roland Groz, and Catherine Oriat reveal the connection between FSM inference and FSM-based test generation in "FSM Inference and Checking Sequence Construction are Two Sides of the Same Coin." They display that identifying a complete deterministic machine can be accomplished by building a checking sequence for it, by only knowing the number of states and the input alphabet.

In the paper "An Approach for Guiding Developers in the Choice of Security Solutions and in the Generation of Concrete Test Cases," the authors Sébastien Salva and Loukmen Regainia present a method for deriving tests for the security aspects. Their method aims to check both the system vulnerability to attacks and the correct implementation of measures taken against these attacks.

Dimitris E. Simos, Josip Bozic, Bernhard Garn, Manuel Leithner, Feng Duan, Kristoffer Kleine, Yu Lei, and Franz Wotawa, in "Testing TLS using Planning-Based Combinatorial Methods and Execution Framework," are interested in testing of the TLS protocol. In particular, they consider automated test generation and automated test execution for this protocol.

Jorge López, Natalia Kushik, and Djamal Zeghlache estimate the quality of the virtual machine (VM) placement in cloud infrastructures in "Virtual Machine Placement Quality Estimation in Cloud Infrastructures using Integer Linear Programming." They focus on the test generation and monitoring techniques for comparing the placement result of a given implementation with an optimal solution with respect to given criteria.

In "Learning and Statistical Model Checking of System Response Times," the authors Bernhard K. Aichernig, Priska Bauerstätter, Elisabeth Jöbstl, Severin Kann, Robert Korošec, Willibald Krenn, Cristinel Mateis, Rupert Schlick, and Richard Schumi present a technique for performance evaluation based on model-based testing and statistical model checking. Their approach is to learn response-time distributions from test data in order to augment existing behavioral models with time aspects, and then to perform statistical model checking for a performance prediction.

"Differences Between a Static and a Dynamic Test-to-Code Traceability Recovery Method" by Tamás Gergely, Gergő Balogh, Ferenc Horváth, Béla Vancsics, Árpád Beszédes, and Tibor Gyimóthy deals with the correctly forming the link between a unit test and the unit under test. Although some guidelines exist to simplify forming this correspondence, in practice, these guidelines are not always followed. In this paper, a semi-automatic method is presented, which uses both code structure and code coverage information.

Pedro Delgado-Pérez, Louis M. Rose, and Inmaculada Medina-Bulo introduce a new definition of metrics for mutation operators in "Coverage-Based Quality Metric of Mutation Operators for Test Suite Improvement," combining information used for classical code coverage and mutation score. Their purpose is to represent the importance of mutation operators for testing in a quantitative way more correctly.

"Automatically Learning Usage Behavior and Generating Event Sequences for Black-Box Testing of Reactive Systems" proposes a test generation method which uses machine learning. The authors M. Furkan Kıraç, Barış Aktemur, Hasan Sözer, and Ceren Şahin Gebizli show how one can use an initial set of test cases to generate additional test cases.

"Test Descriptions with ETSI TDL," authored by Philip Makedonski, Gusztáv Adamis, Martti Käärik, Finn Kristoffersen, Michele Carignani, Andreas Ulrich, and Jens Grabowski, gives an overview of the Test Description Language (TDL) of ETSI. In addition to some latest language design enhancements on TDL, they present an initial discussion on mapping TDL to TTCN-3.

The special issue was motivated by the 29th IFIP International Conference on Testing Software and Systems, which was held in St. Petersburg, on October 9–11, 2017. Both the extended versions of the selected papers from the conference and the papers submitted directly to our open call have undergone a rigorous peer-review process.

We would like to thank all the authors and reviewers contributing to this special issue. Special thanks go to SQJ Editor-in-Chief Rachel Harrison for her support and cooperation. We are also grateful to Katrina Turner and Vincent Salvo, for their timely support and guidance.