SOFTWARE TESTING, VERIFICATION AND RELIABILITY
Softw. Test. Verif. Reliab. 2020;**30**:e1757
Published online in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/stvr.1757

Editorial

Editorial for the special issue of STVR on the 10th IEEE International Conference on Software Testing, Verification, and Validation (ICST 2017)

The 10th International Conference on Software Testing, Verification, and Validation (ICST 2017) was held on March 13 to 17, 2017, in Tokyo, Japan. The aim of the ICST conference is to bring together researchers and practitioners who study the theory, techniques, technologies, and applications that concern all aspects of software testing, verification, and validation of software systems.

In the main research program, the ICST 2017 program chairs, Ina Schieferdecker and Hironori Washizaki selected 36 full papers and 8 short papers for inclusion in the proceedings from among 135 submissions based on the recommendation of the program committee. All papers were refereed by at least three program committee members. Of the 36 full papers accepted, we selected seven papers for consideration for this special issue of STVR. These papers were extended from their conference version by the authors and were reviewed according to the standard STVR reviewing process. We thank all the ICST and STVR reviewers for their hard work. Three papers successfully completed the review process and are contained in this special issue. The rest of this editorial provides a brief overview of these three papers.

The first paper - "Choosing The Fitness Function for the Job: Automated Generation of Test Suites that Detect Real Faults" by Alireza Salahirad, Hussein Almulla, and Gregory Gay - studies the effectiveness of different fitness functions in search-based unit test generation for detecting faults. Experiment results on real faults from the Defects4J database reveal that the branch coverage fitness function is the most effective. The study also reveals that the most important factor related to the likelihood of detection is the satisfaction of the chosen criterion's test obligations.

The second paper - "Complexity Vulnerability Analysis using Symbolic Execution" - Kasper Luckow, Rody Kersten, and Corina Pasareanu - presents a symbolic execution technique for analyzing the worst-case complexity of programs. The technique uses path policies to guide the symbolic execution towards worst-case paths. The evaluation shows that the technique can detect complexity vulnerabilities in realistic software as well as standard implementations of classic algorithms.

The third paper - "Model-based Testing of Apache ZooKeeper: Fundamental API Usage and Watchers" by Cyrille Artho, Kazuaki Banzai, Quentin Gros, Guillaume Rousset, Lei Ma, Takashi Kitamura, Masami Hagiya, Yoshinori Tanabe, and Mitsuharu Yamamoto - presents a model-based testing technique to generate test cases for concurrent client sessions executing against ZooKeeper. The technique defines the semantics of watchers in ZooKeeper and the test oracle to handle the chain of events that eventually leads to the watcher being triggered. The evaluation shows that the technique can detect known defects as well as seeded mutations that implement possible flaws.

Ina Schieferdecker
Fraunhofer FOKUS, Berlin

Atif Memon
University of Maryland

Hironori Washizaki
Waseda University