Introduction to the Special Issue on Automation of Software Test and Test Code Quality

Antonia Bertolino, Istituto di Scienza e Tecnologie dell'Informazione "A. Faedo", CNR, Pisa, Italy
E-mail: antonia.bertolino@isti.cnr.it

Shin Hong, Handong Global University, South Korea
E-mail: hongshin@handong.edu

Aditya P. Mathur, Purdue University, USA and Singapore University of Technology and Design, Singapore
E-mail: aditya_mathur@sutd.edu.sg

We are pleased to present the papers selected for inclusion in the special issue devoted to the 1st International Conference on Automation of Software Test, which was held virtually in colocation with the 42nd IEEE/ACM International Conference on Software Engineering (ICSE 2020).

Software testing is an integral and important part of the software engineering (SE) discipline. Over the past decades, a significant amount of SE research has focused on automation of software test (AST), including the automation of test case generation, test case selection and prioritization, test execution, test verdict analysis and debugging. AST practice has also moved forward significantly, and in recent years many test tools, frameworks, and methodologies have been developed and have significantly enhanced the quality assurance and the productivity in software engineering practices.

Despite the significant achievements, AST remains challenging. To achieve total automation of software testing, a huge amount of code for the test cases and the supporting test infrastructure is needed, which yields itself in turn to large maintenance costs. However, if on the one side it is now generally accepted that disciplined procedures and quality standards should be applied in production code development, on the other side, comparable levels of rigour and quality are not demanded for the code written for testing that production code. Indeed, several recent empirical studies point to how test code is affected by many problems, bugs, unjustified assumptions, hidden dependencies from other tests or environment, flakiness, and performance issues. Because of such problems, test effectiveness is impacted and several false alarms are raised in regression testing that increase the test costs.

Recently, both researchers and practitioners have proposed solutions toward this problem by identifying test code smells or test code quality issues and providing techniques to automatically detect and repair test code bugs and flakiness. In consideration of this active research thread, the 1st ACM/IEEE International Conference on Automation of Software Test (AST 2020) featured “Who Tests The Tests?” as the conference theme. AST 2020 was run for the first time in the format of a colocated conference to ICSE, after a successful series of 14 workshops under the same name.

This special issue offers a venue for researchers and practitioners to share the advances in software test automation, especially on test code quality. In particular, we invited the authors of the best papers of AST 2020 to submit extended versions of their conference publications. Moreover, we also encouraged the authors and participants of the AST series, and the broader software engineering community to submit novel original papers on the themes of software test automation and test code quality, including approaches for: understanding the dimensions and characteristics of problems with unreliable, low-quality test code; identifying and preventing test code smells, test code bugs, flaky tests; impact of test code quality in scaling up test automation of very large, complex system; metrics for test code quality and robustness; automated repair of test code bugs and flakiness; employing Artificial Intelligence and Machine Learning methods to help test automation.

Each of the 10 submitted papers underwent a rigorous review process by independent referees to ensure that the paper had a sound, novel and original contribution to the field of software test automation. Finally, five papers were accepted for inclusion in the special issue; three of these are extended versions of the best papers of AST 2020, and two are novel external submissions.

In particular, the paper titled “Quantum Software Testing – state of the art”, by Antonio García de la Barrera, Ignacio García-Rodríguez de Guzmán, Macario Polo and Mario Piattini, provides a systematic mapping study assessing the state of the art in testing of quantum computing applications, indeed an emerging new paradigm that promises exponential speed up in solving highly-demanding computational problems.

In “An empirical study on how Sapienz achieves coverage and crash detection”, Iván Arcuschin, Juan Pablo Galeotti and Diego Garbervetsky report the results from an empirical study aiming at better understanding how the main features of Sapienz, a powerful tool for automated testing of Android applications using evolutionary algorithms, impact its effectiveness.

Many program analysis and testing tools need to incorporate string solver mechanisms. After observing that adequate tools and benchmarks for comparing existing solvers were lacking, in “ZaligVinder: A generic test framework for string solvers”, Mitja Kulczynski, Florin Manea, Dirk Nowotka and Danny Bøgsted Poulsen propose an extensible framework gathering several string solver benchmarks that can be used for analysis and debugging purposes.

“ExVivoMicroTest: Ex-Vivo Testing of Microservices” by Luca Gazzola, Maayan Goldstein, Leonardo Mariani, Marco Mobilio, Itai Segall, Alessandro Tundo, and Luca Ussi, focuses on ex- vivo testing of microservices during deployment. The authors claim that regression testing of such services may not be adequate prior to deployment due to a lack of knowledge regarding new use scenarios. The authors then propose a technique, named ExVivoMicroTest, that analyzes the behavior of deployed microservices and generates test cases for testing future updates.

In “Fight Silent Horror Unit Test Methods by consulting a TestWizard”, Maura Cerioli, Giovanni Lagorio, Maurizio Leotta, and Filippo Ricca focus on a practical problem, i.e., the existence of incorrect tests. While a totally automated solution to identify invalid tests seems impractical, the idea of TestWizard is to assess individual tests' quality from the point of view of their coherence to specifications.

We would like to thank the Editors-in-Chief of the Journal of Software: Evolution and Process for giving us the opportunity to publish this special issue. We appreciate all reviewers for their efforts in providing thoughtful and constructive comments to improve the quality of the publications. We also thank the authors of all submissions and publications of this special issue.