

Introduction to the Special Issue on Test Automation: Trends, Benefits, and Costs.

Antonia Bertolino^{a,*}, Guglielmo De Angelis^{b,*}, Maurizio Leotta^{c,*}, Filippo Ricca^{c,*}

^a*ISTI-CNR, Pisa, Italy*

^b*IASI-CNR, Rome, Italy*

^c*DIBRIS, University of Genova, Genova, Italy*

Abstract

Keywords: software testing, automation, trends, benefits, costs, special issue.

It is with a great pleasure that we, as the Guest Editors, can finally present the Virtual Special Issue on “Test Automation: Trends, Benefits, and Costs” for the *Journal of Systems and Software*.

Today software is pervasive in any aspects of our society, and automation
5 across the whole Software Testing process has manifested its powerful contribution to the validation of the systems that software controls. Indeed, research in automating both the synthesis of test cases and their execution has made progress towards meeting the expected adequacy requirements while reducing the high effort and costs that testing activities usually demand.

10 Notwithstanding the huge research effort that has been invested on testing techniques and tools, though, we find that software developers still lack adequate knowledge and frameworks to quantify the resources that test automation actually demands to the existing software production eco-systems (e.g., technological stacks, but also software companies or human organisations).

15 In this sense, we launched this special issue in order to explore current research in trends, benefits and costs of test automation that can effectively contribute to the quality assessment of modern software systems. In addition, being aware of the importance about the readiness levels each novel solution has to compare with, we looked for contributions highlighting the opportunities offered
20 by the proposed techniques, while also discussing how to mitigate issues that potentially impact on their broader adoption. Accordingly, we distributed a detailed call for papers, inviting the research community to propose works in test automation regarding approaches, techniques, tools and experience reports

*Corresponding author

Email addresses: antonia.bertolino@isti.cnr.it (Antonia Bertolino),
guglielmo.deangelis@iasi.cnr.it (Guglielmo De Angelis), maurizio.leotta@unige.it
(Maurizio Leotta), filippo.ricca@unige.it (Filippo Ricca)

| Member | Affiliation(s) |
|----------------------|---|
| Jonathan Bell | Northeastern University (USA) |
| Lionel Briand | University of Ottawa (Canada) University of Luxembourg (Luxembourg) |
| Gabriella Carrozza | Accenture Italia (Italy) |
| Marcelo d'Amorim | Federal University of Pernambuco (Brasil) |
| Serge Demeyer | University of Antwerp (Belgium) |
| Sigrid Eldh | Ericsson AB (Sweden) |
| Emelie Engström | Lund University (Sweden) |
| Michael Felderer | University of Innsbruck (Austria) Blekinge Institute of Technology (Sweden) |
| Robert Feldt | Chalmers University of Gothenburg (Sweden) Blekinge Institute of Technology (Sweden) |
| Moonzoo Kim | Korea Advanced Institute of Science and Technology (South Korea) |
| Mika Mäntylä | University of Oulu (Finland) |
| Leonardo Mariani | University of Milano-Bicocca (Italy) |
| Dusica Marijan | Simula Research Laboratory (Norway) |
| Mike Papadakis | University of Luxembourg (Luxembourg) |
| Patrizio Pelliccione | University of L'Aquila (Italy) Chalmers University of Gothenburg (Sweden) |
| John Penix | Google, Inc (USA) |
| Silvia R. Vergilio | Federal University of Paraná (Brasil) |
| Rudolf Ramler | Software Competence Center Hagenberg GmbH (Austria) |
| Federica Sarro | University College London, UK |
| Daniel Sundmark | Mälardalen University (Sweden) |
| Hong Zhu | Oxford Brookes University (UK) |

Table 1: Distinguished Reviewers Committee

about adopting test automation and improving its effectiveness, as well as empirical studies and case studies discussing the costs and benefits of the proposed solutions.

In addition, we recruited a Committee of Distinguished Reviewers, formed by scientists expert in software testing and its automation. The role of this Committee was to ensure a high quality and consistent review process focused on the main objectives of the call. In Table 1 we report the members of the Distinguished Reviewers Committee with their affiliation (at the moment they accepted our invitation). Of course, along the review process we also invited several other external referees to help review the submissions received. To all the committee members and the external reviewers goes our sincere gratitude for the devoted effort that certainly greatly contributed to improve the quality of this Special Issue.

Concerning the visibility and the interest of the research community to the

Special Issue, we received 20 scientific papers on the proposed topic, which were submitted between December 2020 and May 2021. Among these, 11 submissions
40 were finally accepted for publication within the Special Issue. The healthy number of manuscripts we received even bypassed our expectations and remarks the interest of the research community in the proposed topic; moreover, most of the submissions were relevant and sound, resulting in a good acceptance rate (i.e., 55%) at the end of a rigorous review process.

45 In the following we provide a brief summary of the papers included in the Virtual Special Issue, with a cordial invitation to go and read the full original texts. We provide a full reference of the papers in the bibliography of this guest introduction.

In the work “Automatically generating test cases for safety-critical software
50 via symbolic execution” [1], Kurian and co-authors address the automated generation of test cases for safety critical systems based on symbolic execution. They observe that some linguistic features commonly banned from critical programs because they are believed to downgrade safety are also the main efficiency-blockers for test generation based on symbolic execution. Hence, they show how
55 by leveraging the same programming constraints adopted for the sake of safety, symbolic execution -if specifically tailored on the those restrictions- can become a viable approach in practice for a significant class of industrial safety-critical systems. In particular, they present the TECS test generator tool for SCADE programs and report their real-world experience in using TECS for testing the
60 on-board signaling unit of high-speed railway software. In this case study, their test suites achieved a high model coverage with good efficiency and, after the manual augmentation of assertion-style test oracles, could also identify some previous unknown faults.

In the article “Diversity-driven unit test generation” [2], Kessel and Atkin-
65 son explore whether providing multiple implementations of the functionality of interest is viable and effective to improve results of automated unit test generation. The idea is to exploit the redundancy available in mainstream software repositories. More specifically, they present and evaluate a hybrid approach that combines a mainstream unit test generation tool with a software search
70 engine to generate or amplify test sets. They show that by exploiting the diversity available in software repositories, such a hybrid approach outperforms the standalone use of test generation tools.

Within the research work “Discovering boundary values of feature-based machine learning classifiers through exploratory datamorphic testing” [3], Zhu and
75 Bayley address some of the features that hinder the testing of AI applications. Building on some previous results, the authors propose and assess three exploratory testing strategies for machine-learning applications in the framework of the datamorphism testing methodology. These techniques aim at exploring the data space of a classification or clustering application to discover the
80 boundaries between the defined classes. In addition to including an extensive theoretical study of the addressed problem, the work has also practical relevance, as it helps testers understand precisely the behaviour and function of the software under test. An empirical study of the techniques implemented in the

Morphy tool is conducted on a set of controlled experiments and case studies,
85 which evaluated the capability and cost of the proposed test strategies.

In the article “Test automation maturity improves product quality – Quantitative study of open source projects using continuous integration” [4], Wang and co-authors investigate the relationship between test automation maturity and continuous integration (CI) success. They survey main contributors and
90 mine some repositories (GitHub repository, CI repository, and issue track system) of 37 java-based open source projects. From the analysis they distill three notable aspects: (a) a potential benefit of improving test automation maturity level using standard best practices in the literature is improving product quality and accelerating the release cycle in the CI context, (b) higher levels of
95 test automation maturity are highly significantly associated with higher product quality and, (c) the effect of test automation maturity is more significant than product size, product complexity, product popularity, product age, and team size on product quality.

Within the research work “A co-evolutionary genetic algorithms approach to
100 detect video game bugs” [5], Albaghajati and Ahmed present a solution for the automated testing of video games software, which can notoriously suffer from different challenging types of bugs due to its inherent complexity, concurrency and non-determinism. The proposed approach relies on the collaborative work of two agents based on genetic algorithms (i.e., co-evolutionary): one agent
105 generates buggy states, and the other performs a reachability analysis of such states using a colored Petri nets representation of the software workflow. The work includes several experiments through which the authors show the potential of the proposed automated approach in effectively finding bugs in different video games systems.

In the article “SuMo: A mutation testing approach and tool for the Ethereum
110 blockchain” [6], Barboni and co-authors propose a novel mutation testing strategy, and a corresponding infrastructure, for assessing the test suites developed to evaluate smart contracts written for the Ethereum blockchain. Mutation testing is *per se* a rather expensive activity that however has been demonstrated to be
115 one of the most effective ways to improve the quality of a test suite. In order to reduce the cost of such an activity, a set of design choices has been made so to reduce, for instance, the generation of possible stillborn mutants (i.e., invalid mutants that generate compilation errors). At the same time, the framework makes it possible the selection of a set of operators so to reduce the number of
120 generated mutants. The authors evaluated the proposed strategy on a wide set of real smart contracts, demonstrating the usefulness of the approach.

In the work “Cost-effective load testing of WebRTC applications” [7], Gortázar and co-authors investigate novel strategies for non-functional assessment of the real-time video communication standard WebRTC¹, and present an open source
125 framework that can simplify the load testing WebRTC applications in the cloud. Specifically, for each of the proposed strategies the work studies their perfor-

¹see at: <https://webrtc.org/>

mance and costs expressed in terms of number of users emulated, as well as CPU and memory used. In addition, the observed outcomes are also compared with several traditional browser-based strategies.

130 The contribution titled “Selenium-Jupiter: A JUnit 5 extension for Selenium WebDriver” [8] by García and co-authors proposes a novel open-source framework for end-to-end testing of web applications. Specifically, Selenium-Jupiter is a Java-based toolkit that provides seamless integration between the JUnit 5 ² programming model and the features of Selenium WebDriver ³, by expanding
135 the latter and reducing the code that has to be written by developers. The work includes an empirical study highlighting the potentials of the proposed framework on realistic industrial settings. However, the empirical validation also reveals few drawbacks of the current version of the framework: the authors critically address these aspects by discussing their motivations and giving
140 precise suggestions on how to investigate them more in detail.

Within the research work “Automated test-based learning and verification of performance models for microservices systems” [9], Camilli and co-authors mitigate performance and scalability issues in applications based on microservices. Specifically, the work leverages load testing sessions that are driven by realistic
145 workload conditions. Starting from individual performance estimations for each microservice, the proposed approach feeds a Bayesian inference model in order to incrementally learn the performance of the whole application under several operational profiles and alternative deployment scenarios sampled from production. The performance model learned during such in vitro load testing sessions
150 is then verified in order to assess if the microservices application under test meets its non-functional requirements. The work includes an empirical study evaluating benefits and costs of the proposed approach against a benchmark application.

The article titled “Scalability testing automation using multivariate characterization and detection of software performance antipatterns” [10] by Avritzer
155 and co-authors focuses on benefits and costs on the inclusion of a machine learning-based approach for the detection of Software Performance Antipatterns (SPAs) in traditional Continuous Integration (CI) and Continuous Delivery/Deployment (CDD) pipelines. The work identifies parts of system showing the
160 largest impact on scalability by leveraging load testing experiments and multivariate statistical analysis on their outcomes. While the study accepts that both the performance analysis and the SPAs characterization are performed offline, the detection of SPAs is considered an online procedure and the authors provide recommendations for the efficient integration of the latter into the CI/CDD
165 pipelines.

In the article “Automated Web application testing based by pre-recorded test cases” [11], Sunman and co-authors discuss some of the drawbacks of fully automated techniques for testing Web Applications. In this context they pro-

²see at: <https://junit.org/junit5/>

³see at: <https://www.selenium.dev/>

pose a pragmatic and semi-automated approach based on exploratory testing
170 practice. Specifically their solution takes as input a set of interaction sequences
captured during exploratory testing activities; by means of such configurations
the proposed framework automatically generates further test cases by guiding
the exploration of the target Web Application. Despite a small reduction of au-
tomation, the study highlights that the experience of human testers can be easily
175 exploited in order to improve the effectiveness of the whole testing process.

Before concluding, we would like to thank first the Editors-in-chief of the
Journal of Systems and Software, Paris Avgeriou and David C. Shepherd, for
accepting our proposal, and then Raffaella Mirandola, the assigned “Special Is-
sue Editor”, for being always responsive and comprehensive all along the long
180 editorial process. Once again we thank all the many reviewers for their quali-
fied service. And, finally, of course we are grateful to all the authors (of both
accepted and rejected papers) who trusted this initiative and consigned their
work in our hands.

In our role of Guest Editors we are very happy with the obtained result. In
185 fact, this long journey of collecting scientific papers, coordinating their review
process and summarizing them here has led us to discover many interesting
research directions and to ascertain, once again, how the automation of Software
Testing, in all its facets, is important within the field of software engineering.

We sincerely hope you enjoy the papers in this special issue!

190 References

- [1] E. Kurian, D. Briola, P. Braione, G. Denaro, Automatically generating test
cases for safety-critical software via symbolic execution, *Journal of Systems
and Software* 199 (C) (mar). doi:10.1016/j.jss.2023.111629.
URL <https://doi.org/10.1016/j.jss.2023.111629>
- 195 [2] M. Kessel, C. Atkinson, Diversity-driven unit test generation, *Journal of
Systems and Software* 193 (C) (nov 2022). doi:10.1016/j.jss.2022.
111442.
URL <https://doi.org/10.1016/j.jss.2022.111442>
- [3] H. Zhu, I. Bayley, Discovering boundary values of feature-based machine
200 learning classifiers through exploratory datamorphic testing, *Journal of
Systems and Software* 187 (2022) 111231. doi:10.1016/j.jss.2022.
111231.
URL <https://doi.org/10.1016/j.jss.2022.111231>
- [4] Y. Wang, M. V. Mäntylä, Z. Liu, J. Markkula, Test automation maturity
205 improves product quality - quantitative study of open source projects
using continuous integration, *Journal of Systems and Software* 188 (2022)
111259. doi:<https://doi.org/10.1016/j.jss.2022.111259>.
URL [https://www.sciencedirect.com/science/article/pii/
S0164121222000280](https://www.sciencedirect.com/science/article/pii/S0164121222000280)

- 210 [5] A. Albaghajati, M. Ahmed, A co-evolutionary genetic algorithms approach
to detect video game bugs, *Journal of Systems and Software* 188 (2022)
111261. doi:10.1016/j.jss.2022.111261.
URL <https://doi.org/10.1016/j.jss.2022.111261>
- [6] M. Barboni, A. Morichetta, A. Polini, SuMo: A mutation testing approach
215 and tool for the Ethereum blockchain, *Journal of Systems and Software*
193 (2022) 111445. doi:10.1016/j.jss.2022.111445.
URL <https://doi.org/10.1016/j.jss.2022.111445>
- [7] F. Gortázar, M. Gallego, M. Maes-Bermejo, I. Chicano-Capelo, C. Santos,
220 Cost-effective load testing of WebRTC applications, *Journal of Systems and*
Software 193 (2022) 111439. doi:10.1016/j.jss.2022.111439.
URL <https://doi.org/10.1016/j.jss.2022.111439>
- [8] B. García, C. D. Kloos, C. Alario-Hoyos, M. M. Organero, Selenium-
Jupiter: A JUnit 5 extension for Selenium WebDriver, *Journal of Systems*
and *Software* 189 (2022) 111298. doi:10.1016/j.jss.2022.111298.
225 URL <https://doi.org/10.1016/j.jss.2022.111298>
- [9] M. Camilli, A. Janes, B. Russo, Automated test-based learning and verifi-
cation of performance models for microservices systems, *Journal of Systems*
and *Software* 187 (2022) 111225. doi:10.1016/j.jss.2022.111225.
URL <https://doi.org/10.1016/j.jss.2022.111225>
- 230 [10] A. Avritzer, R. Britto, C. Trubiani, M. Camilli, A. Janes, B. Russo, A. van
Hoorn, R. Heinrich, M. Rapp, J. Henß, R. K. Chalawadi, Scalability test-
ing automation using multivariate characterization and detection of soft-
ware performance antipatterns, *Journal of Systems and Software* 193 (2022)
111446. doi:10.1016/j.jss.2022.111446.
235 URL <https://doi.org/10.1016/j.jss.2022.111446>
- [11] N. Sunman, Y. Soydan, H. Sözer, Automated web application testing driven
by pre-recorded test cases, *Journal of Systems and Software* 193 (2022)
111441. doi:10.1016/j.jss.2022.111441.
URL <https://doi.org/10.1016/j.jss.2022.111441>