Received: Added at production | Revised: Added at production | Accepted: Added at production

DOI: xxx/xxxx

ARTICLE TYPE

Editorial: Machine Learning, Software Process, and Global Software Engineering

Igor Steinmacher[1] | Paul Clarke[2,3] | Eray Tüzün[4] | Ricardo Britto[5,6]

[1]Northern Arizona University, Flagstaff, AZ, USA
[2]Dublin City University, Dublin, Ireland
[3]LERO, Ireland
[4]Bilkent University, Ankara, Turkey
[5]Ericsson AB Karskrona, Sweden
[6]Blekinge Institute of Technology Karskrona, Sweden

Correspondence
*Igor Steinmacher, Northern Arizona University.
Email: igor.steinmacher@nau.edu

Summary

On June 26-28, 2020, the International Conference on Software and Systems Processes (ICSSP 2020) and the International Conference on Global Software Engineering (ICGSE 2020) were held in virtual settings during the first year of the COVID pandemic. Several submissions to the joint event have been selected for inclusion in this special issue, focusing on impactful and timely contributions to Machine Learning (ML). At present, many in our field are enthusiastic about the potential of ML, yet some risks should not be casually overlooked or summarily dismissed. Each ML implementation is subtly different from any other implementation, and the risk profile varies greatly based on the approach adopted and the implementation context. The ICSSP/ICGSE 2020 Program Committees have encouraged submissions that explore the risks and benefits associated with ML so that the important discussion regarding ML efficacy and advocacy can be further elaborated. Four contributions have been included in this special issue.

1 | SPECIAL ISSUE PAPERS

In "Global software engineering in the age of GitHub and zoom" [1], James Herbsleb of the Institute for Software Research at the School of Computer Science at CMU presents a compelling reflection on the advances in software development since the early days of Global Software Development (GSD). This reflection emphasizes various advances from source code control systems (e.g. GitHub) to asynchronous chat applications (e.g. Slack), and Open Source Software (OSS). Together with other advances, for example, the accelerated adoption of video conferencing technologies (e.g. Zoom during the COVID-19 pandemic), the world of distributed software development has been radically disrupted, but it has also radically advanced to a point where the very notion of a distributed software development project is no longer viewed as a cumbersome and challenging prospect; indeed, it might be much closer to a norm at present.

Herbsleb's article acknowledges the role of improved computing power and communication speeds (these both being fundamentally important to the various other application-level advances). It emphasizes the transformation in terms of the effect that software has on our very lives. Herbsleb notes that this latter effect is critically important. Software is nowadays embroiled in the tangled web of social influence and media messaging, to the point where "social epistemology" (concerned with accuracy and thoughtful content) has been deemphasized. Various social media recommendation techniques may emphasize user engagement over content accuracy, which has presented broader society with a significant challenge at this time when claims regarding fake news content are pervasive. This challenge impacts software development professionals, and there are clear ethical considerations that software firms and their employees must address if truth is to be prized as an objective that trumps profit. Truth is elusive, perhaps an ideal, but the progress of society will benefit if it is rigorously pursued. Herbsleb's contribution to this special issue incorporates other important reflections, especially in the context of ethics, trust, and truth; readers are encouraged to review Herbsleb's contribution in its entirety.

The second paper in this special issue, "Sparse Reward for Reinforcement Learning based Continuous Integration Testing"[2], addresses the topic of reinforcement learning and how it might be applied to testing in a Continuous Integration (CI) context. Specifically, the consideration is that without intervention, the sparse rewards associated with test failures in high-frequency CI settings can reduce the effect of reinforcement learning. To improve the effect of reinforcement learning, three rewards are examined: the Historical Failure Density-based reward (HFD), the Average Failure Position-based reward (AFP), and the test occurrence frequency. The reported results demonstrate that through careful adaptation, reinforcement learning can improve the normalized average percentage of faults detected (NAPFD), with direct implications for the efficient identification of failures during testing. Techniques such as these demonstrate the positive effect that ML can have on software testing, and efficiencies obtained could have a substantial impact not just on the elapsed time to identify issues in given software builds but also on the total compute power employed in testing. Advances in this area could lower the cost and environmental footprint of CI efforts.

The third contribution to this special issue, "CrowdAssist: A multi-dimensional decision support system for crowd workers"[3], addresses the area of crowdsourcing software development work, acknowledging that challenges arise when aligning developers with posted tasks and with the identification of appropriate task pricing. Individual developer preferences, past tasks, and tasks performed by similar developers are all considered. Importantly, the identification of appropriate pricing is also incorporated in this work. Based on recommendation techniques incorporated in this study, the alignment of developers with tasks is shown to be improved, as is the alignment of task price.

The fourth and final work incorporated in this special issue, "Improving the Detection of Community Smells Through Socio-technical and Sentiment Analysis"[4], examines the adoption of socio-technical and sentiment analysis techniques to improve the identification of possible issues in active OSS projects. In this work, possible (or latent) issues in OSS projects are examined through the lens of techniques such as sentiment analysis. This approach advocates a multi-labeled learning model to identify 10 common types of so-called “community smells” on active OSS projects. Applying genetic programming, an Ensemble Classifier Chain (ECC) is employed to explore the optimal detection rules for each smell type. Through empirical evaluation incorporating 143 OSS projects, this research proposes an improved community smell identification mechanism compared to existing techniques. Advances in this respect are important for the OSS initiative; improved quality in OSS projects can lead to improved quality in the software and systems incorporating OSS projects.

References

1. Herbsleb J. Global software engineering in the age of GitHub and zoom. *Journal of Software: Evolution and Process* 2021: e2347.

2. Yang Y, Li Z, Shang Y, Li Q. Sparse reward for reinforcement learning-based continuous integration testing. *Journal of Software: Evolution and Process* 2021: e2409.

3. Abhinav K, Kaur Bhatia G, Dubey A, Jain S, Bhardwaj N. CrowdAssist: A multidimensional decision support system for crowd workers. *Journal of Software: Evolution and Process* 2021: e2404.

4. Almarimi N, Ouni A, Chouchen M, Mkaouer MW. Improving the detection of community smells through socio-technical and sentiment analysis. *Journal of Software: Evolution and Process* 2022: e2505.