Guest Editorial: Special Section on Mining Software Repositories

Lin Tan¹ · Abram Hindle²

Published online: 6 June 2019

© Springer Science+Business Media, LLC, part of Springer Nature 2019

Mining software repositories as a research area arose from the availability of open source code and information about open source software development. Many open source projects from 2000 on started publishing not only versioned releases of their software, but they were enabling open access to their version control systems, configuration management systems, wikis, mailing lists, and issue trackers. This open access lowered the barriers for researchers of all sorts to start evaluating theories of software development against projects in the wild, as well as discover methods of software development in the wild without the heavy constraint of industrial partnership. Mining software repositories started as a data mining exercise to extract as much as possible from software projects to aid development or reason about software engineering in general. This bolstered empirical software engineering which sought to study software development and software engineering from a fundamentally quantitative and qualitative point of view—this was different than much of the prior research which was tool building or prescriptive recommendations for processes.

This special section on Mining Software Repositories (MSR) serves to highlight and elaborate on papers from the International Conference on Mining Software Repositories 2017. A workshop on mining software repositories was founded in 2004 and grew into an international conference, which still honors its roots in extraction of software source code and metadata from various sources.

This special section serves to showcase and elaborate on the full papers and data showcase papers of particular interest to the MSR reviewers. Many papers were nominated and a few were chosen at the conference and invited. The special section gives our authors more space to be more thorough and concise, explore deeper into the topics discussed at the conference, and do better science in general. Each paper was rigorously reviewed by three or more reviewers,

✉ Abram Hindle
hindle1@ualberta.ca

Lin Tan
lintan@purdue.edu

¹ Purdue University, West Lafayette, Indiana, USA

² Department of Computing Science, University of Alberta, Edmonton, Alberta, Canada

and the authors worked especially hard as every paper faced one major and one or more minor revisions.

The following papers were invited, reviewed, and accepted:

“Mock Objects For Testing Java Systems - Why and How Developers Use Them, and How They Evolve” is a study on the use and evolution of mock objects used in testing. The authors survey developers about mock object use and engage in mining of projects that use mock objects to characterize self-reported and available mock object use and evolution. This study is one of the first studies on mock objects and mocking is a popular and controversial testing topic among developers.

“Classifying code comments in Java software systems” tries to study tens of thousands of topics via manual labelling and machine learning. The authors employ a taxonomy of topic line-types to characterize what exists within topics. This kind of research can aide automated systems that must parse comments for copyright, purpose, technical debt and other concerns.

“Cross-Project Code Clones in GitHub” is an ecosystem wide study of code reuse and why code is copy and pasted. They find that there are projects who act as a clone seeds which enable other projects to follow suite. The authors also propose a clone onion model about the transmission of clones through layers of modularity from with project, to within domain, to within the global software ecosystem.

“Empowering OCL Research: A Large-Scale Corpus of Open-Source Data” is an extension of a data showcase paper that addresses one of the greatest challenges to model driven engineering (MDE) research: empirical data, and MDE based projects. Without existing MDE projects to study much MDE research becomes prescriptive rather than relying on empirical evidence. Not only is a dataset described in great detail and shared, some initial investigations into OCL projects is provided.

“High-level Software Requirements and Iteration Changes: A Predictive Model” presents multiple industrial case studies of software development processes and software requirements undergoing change over numerous iterations of product development. This work provides evolutionary insight into requirements shifting overtime and impact on processes that must address this dynamism.

As guest editors, we express our deep gratitude to both reviewers and authors for making this such a high quality special section on Mining Software Repositories. It would be impossible without community support from our MSR 2017 program committee to help referee these papers to completion. At the same time without interesting and novel work from our community, this would not be possible.

We hope you gain much from this special section on Mining Software Repositories!

Lin Tan and Abram Hindle

Guest Editors

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Lin Tan is an Associate Professor of Computer Science at Purdue University. She received her PhD from the University of Illinois, Urbana-Champaign. One of her research focuses is on leveraging machine learning and natural language processing techniques to improve software dependability, and using software approaches to improve the dependability of machine learning systems. Prior to joining Purdue, she was a Canada Research Chair and an associate professor at the University of Waterloo. She was the program co-chair of MSR 2017, ICSE-NIER 2017, and ICSME-ERA 2015. She is an associate editor of IEEE Transactions on Software Engineering (2017-present) and an editor of the Springer Empirical Software Engineering Journal (2015-present). Dr. Tan's co-authored papers have received ACM SIGSOFT Distinguished Paper Awards at MSR in 2018 and FSE in 2016; and IEEE Micro's Top Picks in 2006. Dr. Tan was a recipient of Canada Research Chair, an NSERC Discovery Accelerator Supplements Award, an Ontario Early Researcher Award, an Ontario Professional Engineers Award — Engineering Medal for Young Engineer, two Google Faculty Research Awards, a Facebook research award, and an IBM CAS Research Project of the Year Award.

Abram Hindle is an associate professor of Computing Science at the University of Alberta. His research focuses on problems relating to mining software repositories, improving software engineering-oriented information retrieval with contextual information, the impact of software maintenance on software energy consumption, and how software engineering informs computer music. He likes applying machine learning in music, art, and science. Sadly Abram has no taste in music and produces reprehensible sounding noise using his software development abilities. Abram received a PhD in computer science from the University of Waterloo, and Masters and Bachelors in Computer Science from the University of Victoria. Contact him at abram.hindle@ualberta.ca.