Empir Software Eng (2013) 18:1043–1046
DOI 10.1007/s10664-013-9273-9

EDITORIAL

Introduction to the special issue on mining software repositories

Tao Xie · Thomas Zimmermann · Arie van Deursen

Published online: 17 August 2013

The Mining Software Repositories field analyzes the rich data available in software repositories to uncover interesting and actionable information about software systems and projects. Thanks to the ready availability of software configuration management, mailing list, and bug tracking repositories from open source projects, it has gained popularity since 2004 with the first instance of the MSR workshop (now conference) and continues to be one of the fastest growing fields in the area of software engineering.

Researchers in this field empirically explore a range of software engineering questions using software repository data as the primary source of information. Some commonly explored areas include software evolution, models of software development processes, characterization of developers and their activities, prediction of future software qualities, use of machine learning techniques on software project data, software bug prediction, analysis of software change patterns, and analysis of code clones. There has also been a stream of work on tools for mining software repositories, and techniques for visualizing software repository data.

In recent years the importance of the field has further increased as today's society and businesses become more data-driven. Industry has a strong interest in transforming software repository data into actionable insights to inform better development decisions. Analytics is already commonly used in many businesses—notably in marketing, to better reach and understand customers (May 2009). The application of analytics to software development is becoming more popular.

For those wishing to develop a broad understanding of the software repository mining research field, there are several resources.

- Tao Xie and Ahmed E. Hassan provide an overview of software repository mining research areas and methods in the tutorial notes to their Mining Software Engineering Data tutorial, given at several recent software engineering conferences (notes are available at https://sites.google.com/site/asergrp/dmse/).

T. Xie (✉)
University of Illinois at Urbana-Champaign, Urbana, IL, USA
e-mail: taoxie@illinois.edu

T. Zimmermann
Microsoft Research, Redmond, WA, USA
e-mail: tzimmer@microsoft.com

A. van Deursen
Delft University of Technology, Delft, The Netherlands
e-mail: arie.vandeursen@tudelft.nl

- Kagdi et al. provide a survey of software repository mining techniques in (Kagdi et al. 2007) that focuses on software evolution research, but which provides broad coverage of a wide range of research methods and repository types used in the field.
- Hassan provides a survey of the entire field along with future research challenges in (Hassan 2008). Together, these three sources give a strong general introduction to software repository mining research.
- The PROMISE repository (http://promisedata.org/) is a collection of 140+ software engineering data sets related to defect prediction, effort estimation, model-based software engineering and many other topics.
- The Working Conference on Mining Software Repositories (MSR) is co-located every year with the ACM/IEEE International Conference on Software Engineering (ICSE). The MSR conference brings together researchers who share an interest in advancing the science and practice of software engineering via the analysis of data stored in software repositories.

Papers in the Mining Software Repositories field tend to take a quantitative empirical approach to exploring research questions. As a consequence, it is natural to select the best papers from the MSR conference for inclusion in Empirical Software Engineering. The papers in this special issue provide a good cross section of the topics and approaches recently explored in the mining software repositories community (MSR 2011).

In the first paper, "*Adoption and use of Java generics*", Parnin, Bird, and Murphy-Hill aim at obtaining insight in the usefulness of Java generics. To that end, they analyze actual usage of Java generics throughout the history of 40 popular open source Java programs. Their observations help us understand how new language features are adopted, and to what extent developers believe that the benefits of generics outweigh the pain of adoption. Key insights include that projects are cautious, and usually do not adopt a large-scale conversion of raw to parameterized types. Furthermore, generics are usually adopted by a single champion in the project, rather than by all committers.

In the paper "*How do open source communities blog*?" Pagano and Maalej analyzed the behavior of 1,100 bloggers in four large open source communities to understand how software communities use blogs compared to conventional development infrastructures. Among other findings they observed an intensive usage of blogs with one new entry every eight hours. Their findings call for a hypothesis-driven research to further understand the role of social media in dissolving the collaboration boundaries between developers and other stakeholders and integrate social media into development processes and tools.

In the paper "*Automated topic naming*", Hindle, Ernst, Godfrey, and Mylopoulos propose an automated solution to suggest topics that describe and relate software artifacts. Previous work attempted to summarize, categorize, and relate software artifacts by using machine-learning techniques such as Latent Dirichlet Allocation (LDA). However, results produced by the previous work are difficult to interpret without meaningful summary labels. The solution proposed by Hindle extracts topics using LDA from commit-log comments and then labels these topics from a generalizable cross-project taxonomy. Their study results on the MySQL, PostgreSQL and MaxDB projects show that their solution can produce appropriate, context-sensitive, insightful labels relevant to these projects.

In the paper "*How (and why) developers use the dynamic features of programming languages: the case of Smalltalk*", Callaú, Robbes, Tanter, and Röthlisberger perform an empirical study of a large Smalltalk codebase to assess how much dynamic and reflective features are actually used in practice, whether some are used more than others, and in which kinds of projects. In addition, they uncover reasons that drive people to use dynamic features, and how these dynamic feature usages can be removed or converted to safer usages. These results are useful to make informed decisions about which features to consider when designing language extensions or tool support.

In the paper "*Software Bertillonage*", Davies, German, Godfrey, and Hindle recover the provenance of software entities with a fast, simple, and approximate technique. The provenance of components included in a deployed software system (e.g., external libraries or cloned source code) is often not clearly documented, causing technical and ethical concerns when managing software assets. To address such issue, Davies et al. have developed a fast, simple, and approximate technique called *anchored signature matching* for identifying the source origin of binary libraries within a given Java application. To demonstrate the validity and effectiveness of their technique, they conducted an empirical study on 945 jars from the Debian GNU/Linux distribution, as well as an industrial case study on 81 jars from an e-commerce application.

We hope you enjoy the papers in this special issue.

Acknowledgments We are grateful to the continuous support and encouragement offered by the Editorial board for the Journal of Empirical Software Engineering and by the Editor-in-Chief Lionel Briand. This issue is the result of a great deal of effort by the reviewers, authors, and attendees of MSR 2011. We thank the authors for keeping up with the review schedule and the reviewers for their detailed and constructive comments which helped shape the papers.

References

Hassan A (2008) "The road ahead for mining software repositories," In Frontiers of Software Maintenance, held with the 2008 I.E. International Conference on Software Maintenance, Beijing, China, pp. 48–57

Kagdi HH, Collard ML, Maletic JI (2007) A survey and taxonomy of approaches for mining software repositories in the context of software evolution. J Softw Maint 19(2):77–131

May T (2009) The new know: innovation powered by analytics. Wiley

MSR (2011) 8th Working Conference on Mining Software Repositories. Waikiki, Honolulu, Hawaii, May 21–22, 2011. http://2011.msrconf.org/

Tao Xie is an Associate Professor in the Department of Computer Science at University of Illinois at Urbana-Champaign. His research interests include software engineering, particularly software testing, program analysis, and software analytics. He received his PhD degree in computer science from the University of Washington at Seattle in 2005.

Thomas Zimmermann is a researcher in the Research in Software Engineering Group at Microsoft Research, adjunct assistant professor at the University of Calgary, and affiliate faculty at University of Washington. His research interests include empirical software engineering, mining software repositories, software reliability, development tools, and social networks, and computer games. He received his PhD degree from Saarland University, Germany in 2008.

Arie van Deursen is a professor at Delft University of Technology, The Netherlands, where he is leading the Software Engineering Research Group. His research interests include program comprehension, collaborative software engineering, software architecture, and software testing. He received his PhD from the University of Amsterdam, The Netherlands, in 1994.