Special section on software clones

Nils Göde¹ · Yoshiki Higo² · Rainer Koschke³

Published online: 16 November 2016
© Springer Science+Business Media New York 2016

Software clones are identical or similar pieces of code, designs, models, requirements, or other artifacts. Clones are known to be closely related to various issues in software engineering, such as software quality, complexity, architecture, refactoring, evolution, licensing, plagiarism, and so on. Various characteristics of software systems can be uncovered through clone analysis, and system restructuring can be performed by merging clones. Clones are the most prominent type of bad smell. Studies found that the average degree of cloned source code is about 9 % if the requested minimal length of a clone is 30 lines of code to be considered.

Research on software clones is published in virtually every conference and journal related to software engineering, and in particular, at the International Workshop on Software Clones (IWSC), an annual workshop dedicated to solidify and give shape to this research area and community. This special section is the result of an open call for contributions on all aspects of software clones with an emphasis on software quality aspects. We announced our open call on various mailing lists related to software engineering and in addition to that invited the best contributions of IWSC 2014. The received submissions were thoroughly reviewed by international experts in the field. Two papers were finally accepted. One paper looks into synchronizing changes to clones and different types of clones in source code, the other one addresses the detection of clones in models rather than source code.

✉ Nils Göde
goede@cqse.eu

✉ Yoshiki Higo
higo@ist.osaka-u.ac.jp

✉ Rainer Koschke
koschke@informatik.uni-bremen.de

¹ CQSE GmbH, Garching, Germany

² Osaka University, Suita, Japan

³ University of Bremen, Bremen, Germany

In *A Comparative Study on the Intensity and Harmfulness of Late Propagation in Near-Miss Code Clones*, Mondal et al. report investigation results on how late propagation differs by clone types, which are identical clones, clones with parameter substitutions, and near-miss clones, that is, clones with further modifications. In late propagation, one clone in a set of clones is modified, causing the clones to become inconsistent. The cloned code fragments are then re-synchronized in a later version. Earlier research suggests that late propagation is related to defects. The authors analyzed thousands of revisions of eight diverse open-source systems and found that late propagation occurs more frequently in near-miss clones compared to the other two types of clones. The authors also found that there is a higher probability that near-miss clones will experience buggy late propagations compared to the other two types of clones.

In *Clone Detection in Matlab Stateflow Models*, Chen et al. present an approach that extends the current state of clone detection in models. While there have been a number of studies on detecting clones in Simulink models, these have so far not considered Stateflow, an important extension of Simulink to model state charts. Especially in the automotive industry, Stateflow is a widely used and important extension of Simulink. While Stateflow charts are stored in the same files as the Simulink models that host them, their representation differs. This prevents the application of current model clone detection approaches to detect clones in Stateflow charts. The contribution of Chen et al. is a transformation of the Stateflow charts that allows the existing model clone detector SIMONE to detect clones in the state charts. The clone information from the state charts can additionally be used to improve the accuracy of clones detected in the hosting Simulink model. The authors' validation shows promising results.

We hope that you will find these papers interesting. We also would like to thank the reviewers who did a great job in helping to prepare the special section and the editors of this journal who gave us the opportunity to publish research on this practical and relevant research topic.

Nils Göde received his diploma in computer science from the University of Bremen, Germany, in 2008. From 2008 to 2011 he was working as a researcher studying the evolution of duplicated code—code clones. In 2011, he received his doctoral degree in software engineering from the University of Bremen. His research interests include code clones, static analysis, software quality and maintainability. Since 2012 he works at CQSE GmbH helping development teams to maintain and improve the quality of their source code.

Yoshiaki Higo received his master's degree and Ph.D. degree in information science and technology from Osaka University, Japan, in 2004 and 2006, respectively. He was an assistant professor at Osaka University from 2007 to 2015 and has been an associate professor since 2015. His research interests include code clone analysis, software repository mining, and refactoring support techniques. He is a member of the IEEE, the IEEE Computer Society, IPSJ, and IEICE.

Rainer Koschke is a full professor for software engineering at the University of Bremen in Germany and head of the software engineering group. His research interests are primarily in the fields of software engineering and program analyses, including architecture recovery, feature location, program analyses, clone detection, and reverse engineering. He is one of the founders of Axivion GmbH, an organization that develops methods and tools to support software maintainers in their daily job through continuous code quality monitoring. He holds a doctoral degree in computer science from the University of Stuttgart, Germany.