2014

Special issue: a selection of distinguished papers from the 18th Working Conference on Reverse Engineering 2011

Martin Pinzger

Denys Poshyvanyk
College of William and Mary

Recommended Citation

Pinzger, M., & Poshyvanyk, D. (2014). Special issue: a selection of distinguished papers from the 18th Working Conference on Reverse Engineering 2011. Journal of Software: Evolution and Process, 26(1), 1-2.

Journal of Software: Evolution and Process

JOURNAL OF SOFTWARE: EVOLUTION AND PROCESS
J. Softw. Evol. and Proc. 2014; **26**:1–2
Published online 31 October 2013 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/smr.1638

Editorial

Special issue: a selection of distinguished papers from the 18th Working Conference on Reverse Engineering 2011

Reverse engineering aims at obtaining high-level representations of software systems from existing low-level artifacts, such as binaries, source code, execution traces, and historical information. Reverse engineering methods and technologies are the keys to develop and evolve large and complex software systems, in particular when it comes to comprehend the design and implementation of such systems, as well as software evolution phenomena.

The Working Conference on Reverse Engineering (WCRE) is the premium platform for presenting and actively discussing innovative methods for reverse engineering and experiences from applying them in academic and industrial settings. WCRE follows a discussion-oriented conference style that is reflected in the mix of full and short research papers, industrial experience reports, workshops and tool demonstrations, as well as in the schedule of the conference, which leaves ample time for discussion and debate.

This special issue presents a selection of five distinguished papers from the 18th Working Conference on Reverse Engineering. The papers were significantly extended by the authors with new material and subjected to additional rounds of revisions and reviewing. In the following, we briefly introduce these five papers.

The first paper 'An Exploratory Study of the Evolution of Communicated Information about the Execution of Large Software Systems' [4] presents a case study on two large open sources and one industrial software system, in which the authors explore the evolution of execution logs and logging statements. The results show that these logs change at a high rate, which could lead to fragile log processing applications. Moreover, the results show that up to 70% of these changes could have been avoided, and the impact of 15–80% of the changes could have been minimized through the use of more robust analysis techniques.

Focusing on improving the layout of source code, the paper 'Automatic Segmentation of Method Code into Meaningful Blocks: Design and Evaluation' [5] presents and evaluates a heuristic to automate the delineation between source code segments. The approach leverages program structure and naming information to identify meaningful blocks of source code and separates them by vertical spacing. The authors evaluate the approach with various programmers having different levels of expertise. Overall, the approach receives strong positive feedback from the various programmers.

The paper 'Predicting Dependencies Using Domain-Based Coupling' [3] presents a novel approach for predicting software dependencies using domain-level relationships between software components. Because it solely relies on domain information, the approach can be used for software projects, where the source code is not available. Furthermore, it allows non-technical domain experts to predict the impact of software changes. The evaluation of the approach is with a large-scale enterprise system in which the authors demonstrate how the approach can assist software maintainers in searching for source code, database, and architectural dependencies.

The paper 'Detecting Asynchrony and Dephase Change Patterns by Mining Software Repositories' [2] presents Macocha, an approach using the k-nearest neighbor clustering algorithm to detect two novel change patterns in the history of software projects. The first change pattern is the *macro editorialco-changes* denoting a set of source files that change together within a large time interval. The second change pattern is the *dephase macro co-changes* denoting macro co-changes that always happen with the same shifts in time. The evaluation with the data from seven open source projects shows that macro co-changes can help explain software evolution phenomena and reduce maintenance costs.

Finally, the paper 'Comparing Text-Based and Dependence-Based Approaches for Determining the Origins of Bugs' [1] presents a study with three open source systems to investigate the effectiveness of existing approaches to determine the origin of bugs. The results show that the text-based approach is

more likely to identify the correct origin than the dependence-based approach but at the cost of reduced precision. On the basis of the results, a number of improvements are explored to improve the effectiveness and efficiency of both approaches.

ACKNOWLEDGEMENTS

We would like to thank the members of the WCRE 2011 program committee and in particular the reviewers of this special issue, who provided us with in-depth, high-quality, and timely reviews. We would also like to thank the WCRE 2011 organization committee who helped make WCRE 2011 a success. Furthermore, we would like to thank Alice Wood and Gerardo Canfora for their support while preparing this special issue.

MARTIN PINZGER
Software Engineering Research Group, University of Klagenfurt, Austria

DENYS POSHYVANYK
Software Engineering Maintenance and Evolution Research Unit, The College of William and Mary, USA

REFERENCES

1. Davies S, Roper M, Wood M. Comparing text-based and dependence-based approaches for determining the origins of bugs. *Journal of Software: Evolution and Process* 2013.
2. Jaafar F, Guéhéneuc Y-G, Hamel S, Antoniol G. Detecting asynchrony and dephase change patterns by mining software repositories. *Journal of Software: Evolution and Process* 2013.
3. Aryani A, Perin F, Lungu M, Mahmood AN, Nierstrasz O. Predicting dependencies using domain-based coupling. *Journal of Software: Evolution and Process* 2013.
4. Shang W, Jiang ZM, Adams B, Hassan AE, Godfrey MW, Nasser M, Flora P. An exploratory study of the evolution of communicated information about the execution of large software systems. *Journal of Software: Evolution and Process* 2013.
5. Wang X, Pollock L, Vijay-Shanker K. Automatic segmentation of method code into meaningful blocks: design and evaluation. *Journal of Software: Evolution and Process* 2013.

AUTHORS' BIOGRAPHIES

Martin Pinzger is a full professor of Software Engineering and the head of the Software Engineering Research Group at the University of Klagenfurt, Austria. He received his PhD in Informatics from the Vienna University of Technology in 2005. He was a postdoc at the University of Zurich and an assistant professor at the Delft University of Technology from which he received his tenure in 2012. His research interests are in software engineering with focus on software evolution, software design, software quality analysis, collaborative software engineering, software repository mining, software visualization, and empirical studies in software engineering. In 2012, he won an NWO Vidi grant, one of the most prestigious Dutch individual research grants. In 2013, he received an ICSE 2013 ACM SIGSOFT distinguished paper award and the ICSM 2013 most influential paper award. He is a member of IEEE and ACM.

Denys Poshyvanyk is an assistant professor at the College of William and Mary in Virginia. He received his PhD degree in Computer Science from Wayne State University in 2008. He also obtained his MS and MA degrees in Computer Science from the National University of Kyiv-Mohyla Academy, Ukraine and Wayne State University in 2003 and 2006, respectively. His research interests are in software engineering, software maintenance and evolution, program comprehension, reverse engineering, software repository mining, source code analysis and metrics. He is a member of the IEEE and ACM.

DOI: 10.1002/smr