JOURNAL OF SOFTWARE MAINTENANCE AND EVOLUTION: RESEARCH AND PRACTICE
*J. Softw. Maint. Evol.: Res. Pract.* 2008; **00**:1–7 *Prepared using* smrauth.cls *[Version: 2003/05/07 v1.1]*

**Editorial**

# Introduction to the Special Issue on Program Comprehension through Dynamic Analysis (PCODA)

Andy Zaidman[1,*,†], Abdelwahab Hamou-Lhadj[2], Orla Greevy[3]

[1] *Delft University of Technology, The Netherlands*
[2] *Concordia University, Canada*
[3] *University of Bern, Switzerland*

This special issue on program comprehension through dynamic analysis is tightly related to the international workshop on program comprehension through dynamic analysis (PCODA) series. The aim of PCODA is to bring together researchers and practitioners using dynamic analysis as a basis for their program comprehension and reverse engineering technique(s). Within the reverse engineering community much attention is focused on static analysis, the analysis of the source code of a software system, while dynamic analysis focusing on runtime properties of software systems, has often been neglected. Nevertheless, dynamic analysis is recognized as yielding a more precise analysis in the face of polymorphism, a language feature widely used in object-oriented software systems.

PCODA was first co-located with WCRE 2005, the 12th Working Conference on Reverse Engineering, in Pittsburgh, and over the past three years has proved to be a very successful event, attracting a constant number of attendees and high quality submissions. We chose to adopt a unique format for the half day PCODA workshop. The authors do not present their own papers, each paper is assigned in advance to another participant who then presents a summary of the paper at the workshop. Experience has shown that the key advantage of adopting this format is that authors are given the opportunity to see an external interpretation of their work which in turn leads to interesting and lively discussions among the authors, the presenter and the audience. Thus it is not surprising that during the PCODA workshop an equal amount of time is devoted to discussion and presentation.

Three highly successful PCODA workshops have led to this special issue of the *Journal of Software Maintenance and Evolution: Research and Practice* devoted to the topic of program comprehension through dynamic analysis. Authors of two papers from the PCODA 2007 workshop were invited to submit extended versions of their papers. An additional 10 papers

*Correspondence to: Andy Zaidman, Delft University of Technology, Mekelweg 4, 2628 CD Delft, The Netherlands
†E-mail: a.e.zaidman@tudelft.nl

were submitted through an open call. Each paper was subjected to a rigorous reviewing process, involving three independent reviewers with expert knowledge in the area and two rounds of revisions. Subsequently four papers were accepted for inclusion in this special issue.

In the paper 'Mining Temporal Rules for Software Maintenance' Lo, Khoo and Liu describe a technique to mine statistically significant temporal rules of arbitrary length to describe system behaviour from sets of traces. They represent their rules as temporal logic expressions that then serve as input to formal analysis toolkits supporting program comprehension, program verification, debugging and specification mining. They demonstrate the scalability of their technique by applying it to two open source case study applications. A key contribution of this work is that each condition of a rule can consist of multiple events.

In the paper 'An Automated Approach for Abstracting Execution Logs to Execution Events' Jiang, Hassan, Hamann and Flora present a technique for abstracting execution logs generated from output statements inserted by developers in the source code. They apply clone detection techniques, the static and dynamic part of each log line is detected, and subsequently each log line is abstracted to its corresponding execution event.

In the paper 'Improving Dynamic Software Analysis by Applying Grammar Inference Principles' Walkinshaw, Bogdanov, Holcombe and Salahuddin propose to improving dynamic analysis by applying grammar inference principles. The authors' approach aims at tackling the fact that dynamic analysis can only provide a partial view of the system. Large traces can be analyzed to infer properties about a program's behavior but it is difficult to obtain a complete state of traces that covers all possible execution paths. Grammar inference is an example of an other field that suffers from the problem of incomplete samples for inferring grammar rules. This paper argues that many solutions in grammar inference that produce reliably accurate approximations of regular grammars can be applied with similar effect to improve dynamic analysis techniques. The authors perform three experiments that show the effect of adopting particular grammar inference principles on the accuracy of dynamic analysis techniques.

In the paper 'A Survey and Evaluation of Tool Features for Understanding Reverse Engineered Sequence Diagrams', Bennett, Myers, Ouellet, Storey, Salois, German, and Charland present a thorough study of the features supported by several tools that focus on the exploration of sequence diagrams, reverse engineered from large execution traces. They have developed a prototype tool, called OASIS, to evaluate the usefulness of these features in understanding the behavior of a software system. The result of the experiment confirms that most existing features are indeed useful in a variety of reverse engineering tasks. In addition, the paper presents the results of a user study that focuses on understanding how existing tools can be improved. Several improvements have been proposed such as the ability to save the state of the session, the ability to navigate between the source code and the extracted sequence diagram, etc. Another important contribution of the paper consists of a rich discussion on how to improve cognitive support in reverse engineered sequence diagram tools.

We hope that readers will enjoy this special issue and through these papers gain useful insights into the domain of dynamic analysis. We would like to thank all the authors who submitted papers to the PCODA workshop series and to this special issue. A special thank you goes to the external reviewers who helped in making this special issue a highly qualitative one. Finally, we would like to thank Aniello Cimitile, the editor in chief of the *Journal of Software*

*Maintenance and Evolution: Research and Practice*, and the publisher Wiley for providing us with the opportunity to devote an issue of this journal to PCODA.

**ACKNOWLEDGEMENTS**

The organization of the PCODA workshop series and this special issue has been sponsored by: the Netherlands Organisation for Scientific Research (NWO) through the "Jacquard RECONSTRUCTOR" project (2005 - 2009), the Swiss National Science Foundation through the project "Analyzing, capturing and taming software change" (SNF Project No. 200020-113342, Oct. 2006 - Sept. 2008), and the Natural Sciences and Engineering Research Council of Canada (NSERC) for the project "Program Comprehension through Dynamic Analysis" (Apr. 2007 - Mar. 2012) led by the DASS (Dynamic Analysis of Software Systems) research group at ECE, Concordia University, Canada.