Software Qual J (2013) 21:99–100
DOI 10.1007/s11219-012-9195-6

Guest editor's introduction to the special section on tests and proofs

Gordon Fraser · Angelo Gargantini

Published online: 9 January 2013

Two important approaches to improving software quality are to prove programs correct or to test software for errors. Traditionally, these two techniques were seen as very different and even competing. Proving people would say: If correctness is proved, what do we need tests for? Testers, on the other hand, would claim that proving is too limited in applicability and testing is the only true path to correctness. As aptly stated by Ed Brinksma in his 2009 keynote at the Dutch Testing Day and Testcom/FATES, "Who would want to fly in an airplane with software proved correct, but not tested?" Indeed, the true power lies in the combination of both approaches. Today, modern testing systems rely on approaches deeply rooted in formal proof techniques, and testing techniques make it possible to apply proof techniques where there was no possibility of this previously.

At a time when even mainstream software engineering conferences start featuring papers with both "testing" and "proving" in their titles, we are clearly on the verge of a new age where testing and proving are not competing but finally accepted as complementary techniques. However, we are not quite there yet, and so in 2007, the first conference on tests and proofs (TAP) was held at ETH Zürich with Bertrand Meyer as Conference Chair and Yuri Gurevich as Program Chair. The idea of this new conference was to provide a forum for the cross-fertilization of ideas and approaches from the testing and proving communities. Following its success, TAP has since been held as an annual event. This special section contains extended versions of selected papers presented at TAP 2010, the 4th International Conference on TAP held during July 12 in Malaga, Spain, as part of the TOOLS Federated Conferences. Out of 12 papers presented at the conference, we invited the authors of four outstanding papers to submit extended versions for this

G. Fraser (✉)
Department of Computer Science, University of Sheffield, Regent Court, 211 Portobello, S1 4DP Sheffield, UK
e-mail: gordon.fraser@sheffield.ac.uk

A. Gargantini
Dipartimento di Ingegneria dell'informazione e metodi matematici, University of Bergamo, viale Marconi, 5, 24044 Dalmine, BG, Italy
e-mail: angelo.gargantini@unibg.it

special section. Each manuscript was reviewed by three reviewers, and after a rigorous selection process, three papers remain for publication.

Research on proof techniques has resulted in many very powerful tools and approaches, and a very successful synergy between testing and proving comes from applying proof-related tools such as SAT and SMT solvers or model finders to the task of test generation. In the paper "Relational analysis of (co)inductive predicates, (co)algebraic datatypes, and (co)recursive functions," Jasmin Christian Blanchette addresses the important issue that the languages used by model finders lack important high-definitional principles. His improvements will allow us to write more expressive specifications.

A related approach to this is when formal specifications are used for test generation; however, the large size of realistic models makes this a difficult task. Jacques Julliand, Nicolas Stouls, Pierre-Christophe Bue, and Pierre-Alain Masson in their paper "B model slicing and predicate abstraction to generate tests" consider the use of abstraction and slicing to reduce model size, which makes it possible to apply test generation even to industrial-sized models.

This idea of model-based testing is used by Ki Yung Ahn and Ewen W. Denney, although their work is an instance where testing techniques are helpful for the proving process and not the other way round. In their paper "A framework for testing first-order logic axioms in program verification," Ahn and Denney use testing to help developers find errors in axiomatizations.

This selection of papers nicely demonstrates how testing and proving can benefit from each other, and we are certain that the future will bring many more successful ideas and approaches benefiting from the combination of testing and proving. We would like to thank all authors and reviewers for their effort and time, and Rachel Harrison, the Editor in Chief of the Software Quality Journal and the editorial staff for their support.