1-1-2010

Software Test Automation

Phillip Laplante
The Pennsylvania State University

Fevzi Belli
Information Technology University

Jerry Gao
San Jose State University

Greg Kapfhammer
Allegheny College

Keith Miller
University of Missouri-St. Louis, millerkei@umsl.edu

See next page for additional authors

Authors
Phillip Laplante, Fevzi Belli, Jerry Gao, Greg Kapfhammer, Keith Miller, W. Eric Wong, and Dianxiang Xu

Hindawi Publishing Corporation
Advances in Software Engineering
Volume 2010, Article ID 163746, 2 pages
doi:10.1155/2010/163746

Editorial

Software Test Automation

Phillip Laplante,[1] Fevzi Belli,[2] Jerry Gao,[3] Greg Kapfhammer,[4] Keith Miller,[5] W. Eric Wong,[6] and Dianxiang Xu[7]

[1] *Engineering Division, Great Valley School of Graduate Professional Studies, Penn State, 30 East Swedesford Road, Malvern, PA 19355, USA*
[2] *Department of Electrical Engineering and Information Technology, University of Paderborn, 33095 Paderborn, Germany*
[3] *Computer Engineering Department, San Jose State University, One Washington Square, San Jose, CA 95192-0180, USA*
[4] *Department of Computer Science, Allegheny College, Meadville, Pennsylvania, USA*
[5] *Department of Computer Science, University of Illinois at Springfield, One University Plaza, UHB 3100, Springfield, IL 62703, USA*
[6] *Department of Computer Science, The University of Texas at Dallas, 800 West Campbell, Richardson, TX 75080, USA*
[7] *National Center for the Protection of the Financial Infrastructure, Dakota State University, Madison, SD 57042, USA*

Correspondence should be addressed to Phillip Laplante, plaplante@psu.edu

Received 31 December 2009; Accepted 31 December 2009

Software testing is an essential, yet time-consuming, and expensive activity. Therefore, automation of any aspect of software test engineering can reduce testing time and, in the long-run, reduce costs for the testing activity. While there are many research directions in testing automation, from theory through application, the main focus of this special issue is on partially or fully validated tools, techniques, and experiences.

In response to the call for papers a broad range of submissions were received. Consistent with the standards of a highly regarded journal, all submissions received several rounds of review and revision and only outstanding papers were selected, yielding an acceptance rate of 60%. The resultant collection provides a number of important and useful results.

For example, in "A tester-assisted methodology for test redundancy detection" Koochakzadeh and Garousi propose a semiautomated methodology based on coverage metrics to reduce a given test suite while keeping the fault detection effectiveness unchanged. They then validate their approach on Allelogram, an open source Java program used by biological scientists for processing genomes. The results of the experiments confirm that the semiautomated process leads to a reduced test suite with the same fault detection ability as the original test suite.

In "A strategy for automatic quality signing and verification processes for hardware and software testing," Younis and Zamli give a technique for optimizing the test suite required for testing both hardware and software in a production line. Their strategy is based a "Quality Signing Process" and a "Quality Verification Process." The Quality Signing Process involves parameter interaction while the Quality Verification Process is based on mutation testing and fault injection. The novelty of the proposed strategy is that the optimization and reduction of test suite is performed by selecting only mutant killing test cases from cumulating t-way test cases. The results demonstrate that the proposed strategy outperforms traditional block partitioning with the same number of test cases.

There are considerable costs involved in regression testing, which often cause practitioners to short-circuit the activity. In "Automated test case prioritization with reactive GRASP" Mai et al. propose the use of the Reactive GRASP (Greedy Randomized Adaptive Search Procedures) metaheuristic for the regression test case prioritization problem. They compare this metaheuristics with five other search-based algorithms previously described in the literature. Five programs were used in the experiments and the results demonstrate good coverage performance with respect to many of the compared techniques and a high stability of the results generated by the proposed approach. Their work also confirms some of the previous results reported in the literature concerning the other prioritization algorithms.

Automated GUI test case generation is a highly resource intensive process. In "A proposal for automatic testing of GUIs based on annotated use cases," Mateo Navarro et al. describe a new approach that reduces the effort by automatically generating GUI test cases. The test case generation process is guided by use cases describing behavior recorded as a set of interactions with the GUI elements. These use cases are then annotated by the tester to indicate interesting variations in widget values and validation rules with expected results. Once the use cases are annotated, this approach uses the new values and validation rules to automatically generate test cases and validation points, expanding the test coverage.

The next paper by Darwin, "AnnaBot: a static verifier for Java annotation usage," describes one of the first tools permitting verification of correct use of annotation-based metadata in Java. This verification becomes especially important both as annotations become more widely adopted and as multiple implementations of certain standards become available. The author describes the domain-specific language and parser for AnnaBot, which are available for free from the author's website

Finally a study of the software test automation practices in industry was conducted and also the results given in "Software test automation in practice: empirical observations" by Kasurinen et al.. Both qualitative interviews and quantitative surveys of 55 industry specialists from 31 organizational units across 12 software development organizations were conducted. The study revealed that only 26% of the organizations used automated testing tools, primarily in quality control and quality assurance. The results also indicated that adopting test automation in software organization is a demanding effort

We hope you enjoy this eclectic set of works relating to software testing automation. The editors also wish to thank the authors for their excellent contributions and the reviewers for their outstanding efforts in helping to select and improve all of the papers in this special issue.

Phillip Laplante
Fevzi Belli
Jerry Gao
Greg Kapfhammer
Keith Miller
Eric Wong
Dianxiang Xu