Benefits and Limitations of Automated Software Testing: Systematic Literature Review and Practitioner Survey

Dudekula Mohammad Rafi*, Katam Reddy Kiran Moses*, Kai Petersen*,+
Blekinge Institute of Technology/ Ericsson AB +*
School of Computing
Karlskrona, Sweden
(mrdu09, kika09, kai.petersen)@bth.se

Mika V. Mäntylä
Lund University
Department of Computer Science
Lund, Sweden
mika.mantyla@cs.lth.se

***Abstract*—There is a documented gap between academic and practitioner views on software testing. This paper tries to close the gap by investigating both views regarding the benefits and limits of test automation. The academic views are studied with a systematic literature review while the practitioners views are assessed with a survey, where we received responses from 115 software professionals. The results of the systematic literature review show that the source of evidence regarding benefits and limitations is quite shallow as only 25 papers provide the evidence. Furthermore, it was found that benefits often originated from stronger sources of evidence (experiments and case studies), while limitations often originated from experience reports. We believe that this is caused by publication bias of positive results. The survey showed that benefits of test automation were related to test reusability, repeatability, test coverage and effort saved in test executions. The limitations were high initial invests in automation setup, tool selection and training. Additionally, 45% of the respondents agreed that available tools in the market offer a poor fit for their needs. Finally, it was found that 80% of the practitioners disagreed with the vision that automated testing would fully replace manual testing.**

***Keywords*-automated software testing; benefits; limitations**

I. Introduction

From a research perspective automated testing is a mature research area. Though, at the same time it is recognized that there is a gap between academic research and the benefits and problems actually encountered in applying software testing in industry. In [1] the previous fact is acknowledged by practitioners as well as academicians. The vision of automated testing aims at 100% automtion [2]. However, in practice this vision has not been yet fulfilled (e.g. [3]). The research results are often reported in the form of case studies, experience reports, and experiments, which provides rich descriptions and insights, but at the same time limits generalizability for the software industry at large. This underlines the need to investigate what the industry at large experiences with regard to automated testing benefits and limitations.

We evaluated the following research question: *Are benefits and limitations of empirical studies and experience reports observed in industry at large?*

In order to address the above mentioned research gap and research question, this study makes the following contributions: (1) Identify benefits and challenges that are based on empirical studies and experience reports in literature (Section III); (2) We devise a survey and test whether the observed benefits and limitations are prevalent in industry (Section III). We also present the validity threats (Section IV) and reflect on the results (Section V).

II. Systematic Literature Study

We used systematic literature review [4] to identify benefits and limitations of AST.

A. Conduct

Table I shows the search terms used for title, abstract, and keywords to identify articles related to automated software testing that are based on practical experience. Columns are connected with AND, and items within columns are connected with OR.

Table I
Keywords

Population	Intervention 1	Intervention 2	Outcome
K1: Software	K5: Tool*	K7: Test*	K11: Empirical
K2: Application	K6: Automat*	K8 Quality assurance	K12: Industrial
K3: Program		K9: Validation	K13: Practical
K4: Develop*		K10: Verification	K14: Case Study
			K15: Survey
			K16: Experience*
			K17: Experiment*

The search was focused on the time period 1999-2011 as testing tools have evolved in the last decade and became more powerful, which would affect the evaluation with respect to limitations and benefits.

The databases used for the search were IEEE Explore, Engineering Village (including Inspec and Compendex), Scopus, ACM, and Google Scholar. The search resulted in a total of 24.706 articles. For the selection of articles the following steps were taken (see Table II): First, duplicate abstracts were removed. Second, the titles were read to determine whether the article focuses on AST (including automated test execution, test generation and selection, as well as result evaluation and test quality analysis) in the context of software engineering. Third, the abstracts were read to check whether they contained the search words, had empirical background, and focused on the application of automation methods, tools, techniques, and approaches for AST. Often, it was not clear from the abstract if a paper was relevant, and discussed benefits and limitations of AST. Such papers were not excluded right away. Instead, the introduction, conclusion, and full-text of these papers were read, resulting to rejection of irrelevant papers.

Table II
STEPS

Criteria	No. of articles (in → out)
Step 1: Remove duplicates	24706 → 19920
Step 2: Check title	19920 → 9456
Step 3: Check abstract	9456 → 1470
Step 4: Check intro/conclusion	1470 → 227
Step 5: Read full-text	227 → 25

Prior to study selection based on titles and abstracts a test-set of 50 articles was used to determine whether reviewers have the same understanding of the inclusion and exclusion criteria. We calculated the Kappa k value (a measuring for determining the agreement between reviewers [5]) and achieved a value k=0.605, which indicates substantial agreement. Hence, this indicates that the inclusion/exclusion criteria were clearly formulated. Thereafter, the articles were divided between authors for individual inclusion/exclusion.

The data of each paper was extracted using a form containing fields for title, author, year, subject of investigation, research methodology, and relevant area of research (challenges/benefits). The analysis of the review was done using thematic analysis in combination with narrative summaries.

B. Results

Table III illustrates which sources of evidence were used. It is visible that the majority of studies are of empirical nature following a research methodology (experiments and industrial case), while 8 papers are experience reports.

Table III
SOURCE OF EVIDENCE

Method	References	No. of articles
Experiment	[6][7][8][9][10][11][12][13][14][15]	10
Industrial Case	[16][17][18][19][20][21][22]	7
Experience	[23][24][25][26][27][3][28][29]	8

Tables IV and V provide an overview of the benefits and limitations identified in the literature review.

For the benefits the results are well supported by evidence from experiments and case studies. In comparison to experiments and case studies only few experience reports (e.g. [29], [3], [26]) appear. The results relating to limitations are primarily based on experience reports, whereas only few empirical studies support them (such as [21], [19]).

Table IV
BENEFITS

ID	Description of Benefits	References
B1	*Improved product quality:* Quality in terms of fewer defects present in the software product	[11], [21]
B2	*Test coverage:* High coverage of code (e.g. statement, branch, path) is achieved through automation	[6], [8], [9], [11], [14], [21]
B3	*Reduced testing time:* Time required for testing, i.e. the ability to run more tests within a timeframe	[10], [21], [22], [29]
B4	*Reliability:* AST is more reliable when repeating tests as variance in outcomes can be due to the manual tester running the tests in a different way, but can not make use of the knowledge of the tester	[29]
B5	*Increase in confidence:* Increase of confidence in the quality of the system (e.g. as perceived by developers)	[16], [29]
B6	*Reusability of tests:* When tests are designed with maintenance in mind they can be repeated frequently, a high degree of repetition of test cases leads to benefits, not a single execution of an automated test case	[7], [21]
B7	*Less human effort:* Automation reduces human effort that can be used for other activities (in particular ones that lead to defect prevention)	[7], [10], [12], [24], [3]
B8	*Reduction in cost:* With a high degree of automation cost are saved	[7], [9], [20]
B9	*Increased fault detection:* Effectiveness as the ability to detect a large portion of defects in a system due to high fault detection ability	[6], [13], [27], [18], [20], [26]

Table V
LIMITATIONS

ID	Description of Limitations	References
L1	*Automation can not replace manual testing:* Not all testing tasks can be easily automated, especially those that require extensive knowledge in a domain	[21], [3]
L2	*Failure to achieve expected goals:* Organizations are tempted by running tests in a fraction of time, but failed to achieve lasting or real benefits	[29], [28]
L3	*Difficulty in maintenance of test automation:* Change in technology and evolution of software products leads to difficulty in maintaining automated tests	[21], [3], [17], [29]
L4	*Process of test automation needs time to mature:* Creating the infrastructure and tests for automation requires time, hence maturity of automation (and related benefits) requires time	[19]
L5	*False expectations:* Organizations have impractical expectations when it comes to AST with the aim of saving as much cost as possible (e.g. by spending effort on unproductive testing activities)	[3], [29], [25]
L6	*Inappropriate test automation strategy:* An appropriate strategy (e.g. what test levels to automate with what purpose) is hard to decide, hence leading to inappropriate strategies that do not allow to utilize the benefits of AST	[3], [29]
L7	*Lack of skilled people:* To automate tests many skills are needed (e.g. knowledge of test tools, general software development skills, domain and system knowledge)	[26]

III. PRACTITIONER SURVEY

The goal of the practitioner survey was to determine whether the benefits and limitations are of relevance for the software industry at large.

A. Conduct

Sample: The online survey was distributed through forums on AST, Yahoo Groups, Google Groups, and LinkedIn. In addition, the survey was distributed by e-mail to industry contacts. In total, we got 115 valid responses (questionnaire filled in completely and pre-check of whether they have industrial experience).

Survey design: Prior to publishing the survey widely it was tested by 10 respondents who are familiar with the field of AST and the survey was updated based on their suggestions to improve its clarity. The survey [1] consisted of two main parts, namely:

- Demographic questions asking for the role of the respondent, experience level, system type, and development model used.
- Questions were defined using benefits and limitations from the systematic literature results as input. Tables VII, VIII state the questions asked. The questions are in the form of an opinion survey checking the level of agreement/disagreement in response to stated facts.

B. Results

1) Demographics: The majority of respondents (53.91%) has a designated quality assurance role, followed by programmers doing testing (23.48%). Only few other roles, such as system architect (6.96%), system designer (3.48%), researcher (3.48%) and project manager (3.48%), answered the survey.

With respect to total experience in years we can see that the respondents cover a wide range of experience, with 52.17% of them having less than 5 years of experience, 28.69% having 5 to 9 years, 15.65% having 10 to 15 years, and 3.48% having more than 15 years.

For system type (Table VI) we allowed the practitioners to provide several answers (e.g. they might develop embedded systems in the healthcare domain). The results show that all domains are covered, with the majority of them being for Web, Finance, and Healthcare.

Table VI
SYSTEM TYPE

System type	Answers total	Answers %
Web	56	26.31
Finance	37	17.29
Healthare	26	12.15
Mobile	19	8.88
Other	19	8.88
Telecommunication	18	8.41
ERP	17	7.94
Embedded system	11	5.14
Games/Entertainment	8	3.74
Others	3	1.40

The majority of the respondents used agile software development (61.74%), followed by waterfall-process/plan-driven development (25.21%). Lean software development is only used by few respondents (2.61%). Other models were used by 10.43% of the respondents.

[1] https://sites.google.com/site/kaipetersen79/files/AST.pdf?attredirects=0&d=1

2) Benefits: Table VII shows the benefits of AST as they appeared in the survey, and also includes references to the benefits identified in the systematic literature review to illustrate which benefit in the review led to each question. Furthermore, the answers on a 5-point Likert scale are shown, including the total number of answers and the percentages, as well as the median value. The answers are ranked in ascending order with respect to the sum of the number of responses answering agree (4) and completely agree (5).

Overall, it is visible that the benefits of AST that were found in literature are strongly supported by the respondents, with at least half of them agreeing or fully agreeing to 8 out of 9 statements. Only for rank 9 (high fault detection) more than half of the respondents are either indifferent or disagree/strongly disagree.

In the following paragraphs we reflect on each answer, taking the free text answers by the respondents into account.

B.R1: Overall, 86% of the respondents choose to completely agree or agree on that benefit of high reusability of test cases. In the free text answers it was highlighted that the success is conditional upon right test strategies, which e.g. relate to challenges such as skills of testers with respect to technical and tool knowledge, as well as investment in tools (see Table VIII).

B.R2: This question is relatively similar to the previous one (B.R1), but asks more explicitly about repeatability in connection to saving time. In total 84% agree or fully agree on that benefit. However, it was highlighted that this should not be the motivation/goal for testing. The motivation should be to *“rather to do better testing in the time provided by the project stakeholders”*.

B.R3: 75% agree on the positive relation between automation resulting in better test coverage and hence improved product quality. Practitioners provided recommendation of how better test coverage should be understood in this case. One respondent pointed out that *“at times I want to drop most of my automated tests when they become a too heavy burden. They don’t improve the quality by better coverage, but by sufficient coverage”*. Furthermore, one respondent pointed out that only tests should be automated that provide good results in a repeatable fashion (e.g. considering the life-cycle of the test). It was also highlighted that prior to automation coverage criteria need to be clearly defined as this reduces the overall number of tests that need to be automated and maintained.

B.R4: This question relates to re-running tests and with that saving time and cost and that in comparison to manual testing, which 72% of the respondents agree with. Here it was pointed out that if the same bugs are introduced it would be useful to rerun the same test, however, practitioners doing manuel testing can use their experience and knowledge of changes to focus testing effort. Furthermore, specific types of tests take longer time to automate, and there human

Table VII
SURVEY RESULTS FOR BENEFITS

Rank (4+5)	Questions related to benefits	Answers on Scale[1]					Median
		5	4	3	2	1	
B.R1	B6: High reusability of tests makes automated testing productive	46	53	13	3	0	4
		40%	46%	11%	3%	0%	
B.R2	B6/B3: AST enables the repeatability of tests, which gives the possibility to do more tests in less time	38	59	10	6	2	4
		33%	51%	8%	6%	2%	
B.R3	B1/B2: AST improves product quality by better test coverage.	28	59	20	7	1	4
		24%	51%	18%	6%	1%	
B.R4	B8/B3/B6: AST saves time and cost as it can be re-run again and again and hence is much quicker than manual testing	29	54	15	15	2	4
		25%	47%	13%	13%	2%	
B.R5	B5: AST provides more confidence in the quality of the product and increases the ability to meet schedules	25	51	19	16	4	4
		22%	44%	17%	14%	3%	
B.R6	B7: The investment in application-specific test infrastructure can significantly reduce the effort that automation requires from testers	13	63	28	11	0	4
		11%	55%	24%	10%	0%	
B.R7	B7/B8: By having complete automation the cost of AST is dramatically reduced and facilitates continuous testing	22	52	17	10	4	4
		19%	45%	16%	17%	3%	
B.R8	B4/B7: AST requires less effort on the developers side, but cannot find complex bugs as manual testing does	19	49	19	24	4	4
		17%	43%	16%	21%	3%	
B.R9	B9: Automated testing facilitates high fault detection	9	38	35	29	4	3
		9%	33%	30%	26%	3%	

[1]5=completely agree, 4=agree, 3=neutral, 2=disagree, 1=completely disagree

intervention is more useful, one respondent saying that *“Tests executed several times save time and cost, but some kinds of automated tests take much more time than manual (especially GUI tests)”*.

B.R5: The association of AST with higher confidence in product quality and the ability of meeting schedules was agreed on by 66% of the respondents. In order to gain the confidence, respondents pointed out that this *“requires a lot of preparation and appropriate usage”* and on *“how good the automated test software is”*.

B.R6: This benefit relates to that application-specific test infrastructure helps to reduce the effort that automation requires from testers. In total 66% agreed to that, while no specific explanations were provided by the practitioners.

B.R7: 64% of the respondents agreed that complete automation reduces testing cost and at the same time facilitates continuous testing. In relation to that statement two respondents pointed out that full test automation is not possible, saying that *“First off, you will never have ”complete” automation. Also, there can never be a replacement for manual testing. It can reduce the cost ...”*. With respect to continuous integrating one respondent recommended to *”automate what makes sense and yes, if properly implemented automation (different types like Unit, Smoke) can be part of continuous integration/testing.”*

B.R8: This question asks for a trade-off between effort from the perspective of developers in relation to the lack of ability to catch complex bugs, that could be better found with manual testing. In total 60% of the respondents agree to that statement. At the same time there is a substantial part (24%) of the respondents disagreeing. Two answers provided in that context were that *“In fact proper automated testing can find much more complex bugs”* and *“I think it is possible for automated testing to find as many complicated bugs”*.

B.R9: Only few participants agree (33%) with that automated testing facilitates high defect detection. In particular, the answer to this question seems inconclusive as around 1/3 of the respondents agrees, 1/3 is uncertain, and 1/3 disagrees. One explanation was e.g. *“The fact that the testing is automated does not increase the fault detection rate. It is the tester creating the tests which facilitates high fault detection. ...it depends on how it is used”*.

3) Limitations: The limitations are presented in the same fashion as the benefits (see Table VIII). The table shows that the limitations are also strongly supported by practitioners.

L.R1: The first question relates to initial investment needed in automated testing and that it requires time to mature until benefits are seen. The limitation is recognized by 89% of the respondents. In relation to manual testing the investment in automated testing is perceived as worthwhile, as one respondent points out *“Manual testing, if repeated over and over, is a very large waste of money. Automated testing is an investment, and requires only time to be far more rewarding than manual regression tests”*.

L.R2: In the beginning the cost for AST is higher and requires maturation. This statement finds strong support by 88% of the respondents. In fact, one respondent states that *“Test automation needs at least as much maintenance as the developed software with regards to the Technical Debt”*. In order to avoid some of the initial costs, one respondent points out that *“If you plan early you can create manual test scripts that can also be used for automated testing, hence let time required for maintenance and /or conversion”*.

L.R3: With respect to successful automation and required skills 81% agree to that challenges. However, some respon-

Table VIII
SURVEY RESULTS FOR LIMITATIONS

Rank	Questions related to limitations	Answers on Scale[1]					Median
		5	4	3	2	1	
L.R1	L4: Compared with manual testing, the cost of AS is higher, especially in the beginning of the automation process. However, automated software testing can be more productive after a period of time	42	60	8	4	1	4
		37%	52%	7%	3%	1%	
L.R2	L3/L4: Automated testing needs extra effort for designing and maintaining test scripts.	37	64	7	6	1	4
		32%	56%	6%	5%	1%	
L.R3	L6/L7: Testers should have enough technical skills to build successful automation	40	53	12	9	3	4
		35%	46%	10%	8%	3%	
L.R4	L5: Compared with manual testing, AST requires a high investment to buy tools and train staff to use the tools	32	56	12	11	6	4
		28%	49%	10%	10%	5%	
L.R5	L5: AST requires less effort on the developers side, but cannot find complex bugs as manual testing does	19	49	19	24	4	4
		17%	43%	17%	21%	3%	
L.R6	L3: Most of the testing tools available in the market are incompatible and do not provide what you need or fits in your environment.	11	40	30	24	10	3
		10%	35%	26%	21%	9%	
L.R7	L1: Automated testing fully replaces manual testing.	1	6	16	49	43	2
		1%	5%	14%	43%	37%	

[1] 5=completely agree, 4=agree, 3=neutral, 2=disagree, 1=completely disagree

dents put that challenge in perspective, e.g. highlighting that *"Not all testers have these technical skills... not every tester can be an automater"*. Furthermore, another tester said: *"Completely agree, and completely disagree. I have met outstanding testers who did not have automation skills. You do not need each tester on your team to be a clone. You want a variety of skills in your testers, So, I would say the test team should have enough skills to build successful automation. Maybe your tester without automation skills has spent their time on developing powerful systems thinking skills instead"*.

L.R4: 77% agree that in comparison with manual testing AST requires high investment to buy tools and train staff to use the tools. However, as pointed out by a respondent, the actual measured cost is harder to track, saying that *"I would say completely agree, except that the investment for purchasing automation software and training is more visible than that for manual. Manual training is still expensive in terms of time spent by mentors, which is harder to track, and less visible"*. Further comments related to L.R1 highlighting that the cost will be balanced out later, e.g. due to that regression testing becomes easier.

L.R5: This statement is also listed here as it raises a drawback of not being able to find as complex defects (for detailed description see B.R8).

L.R6: 45% agree that testing tools available on the market are incompatible or do not fit well to tool requirements, which indicates that this drawback is not as strongly supported as the ones previously presented. One respondent pointed out that *"there are multiple tools out there, yes. But you need the right one for your situation. There isn't, and never will be a one size fits all tool"*, and another adding that *"there are a large number of tools available for different cost levels, pricing, etc., as well as different targeted feature sets and capabilities. No tool is going to be good for everyone but there are many good tools on the market, both free and for cost. A good test manager and testing group will spend the time to research and select the right tool for the environment"*. Furthermore, the skill of the tester also affects the choice of tool, as one respondent puts it: *"Most automated test tools are not as useful as having a skilled tester. It's the application of the tester's skills that makes a tool valuable, and if the tool prevents the tester from doing what they want, that tool is less valuable"*.

L.R7: The answer to this question clearly shows that testers do not believe in full automation (only 6% of the respondents), and hence this remains a challenge . Practitioners stated that new development requires manual testing, one saying *a large amount of manual testing is required to deal with new development and to uncover complex problems*. Furthermore, automated testing should be seen as a complement to manual testing, as one respondent puts it: *"NO! It is a tool only and should be used as a complement to manual testing. You automate to allow your manual testing to work on high value tasks, and not be stuck in the mud with re-running other mundane tasks. Think of automation as a way to get efficiency gains for your overall testing effort"*.

4) Overall Satisfaction: The overall satisfaction relates strongly to false expectations (see L5 in Table V) as if testers would be generally unhappy with using automation, they expectations would be largely disappointed. Overall, Table IX shows that the clear majority of the respondents were satisfied, and highly satisfied.

IV. VALIDITY THREATS

Given that the literature review is subject to interpretation in all its steps there is a risk of bias. To reduce this threat we analyzed and refined our criteria for study selection and only

Table IX
OVERALL SATISFACTION WITH AST

Satisfaction	Answers total	Answers %
Highly satisfied	45	39.13
Just satisfied	52	45.22
Yet to see real major benefits of automation	17	14.78
Not at all satisfied	4	0.87

when having a high level of agreement separated the work to make sure that everyone has a similar understanding. All borderline cases where there was a doubt in selection and analysis were discussed to further reduce bias. Overall, this threat is still present, however, actions have been taken to reduce bias in study selection.

In the survey there is a risk that the questions might be misunderstood. Therefore, the survey was tested for understandability prior to making it public. However, questions might still leave room for interpretation, which could not be completely ruled out. Another common risk in surveys is that they are biased towards a specific population. This threat is partially under control as answers came from a variety of domains, but the web domain was clearly the domain with the highest number of responses. The main threat in this survey is that there was no distinction made between purpose of automation (i.e. test data generation, test case generation, and test execution), i.e. it is unknown whether the results are skewed with respect to purpose.

V. CONCLUSION

This paper makes three contributions. First, we performed a systematic review of software test automation benefits and limitations in academic literature. We collected 24.706 papers, which were reduced to 25 research works (see Table II). Thus, the amount of evidence on these matters is quite shallow as many benefits and limitations are backed up by only one or two sources. Furthermore, we found that while benefits often came from stronger sources of evidence (experiments and case studies), limitations were more frequently reported on experience reports. We think that this is caused by publication bias regarding the benefits. We believe that important further work on this area is to assess the limitations of test automation with rigorous empirical studies, i.e. case studies and experiments.

Second, we conducted a survey of the practitioners' view of software test automation benefits and limitations. The results showed that the main benefits of test automation are reusability, repeatability and effort saved in test executions. These results support the superiority of test automation when several regressions testing rounds are needed. Furthermore, the practitioners indicate that automation improves test coverage, which means that automation has benefits even when excessive regression testing is not needed. Regarding the limitations, we found that automation bares a high initial cost in designing the test cases, buying a test automation tool, and training the staff. Non-surprisingly, the maintenance of automated test cases was also perceived as problematic. Also 45% of the practitioners think that current test automation tools offer a poor fit for their needs. The limitations of test automation perceived by the practitioners should outline important future research directions. For example rather than trying to come up with tools that maximize test coverage researchers would better serve the practitioners by trying to come up with tools that:

1) Have an easy learning curve. This would mitigate the high initial investment required for test automation.

2) Utilize test cases that are highly maintainable and robust. The maintenance burden of automated testing is likely to increase in the future as we have already seen systems were the amount of test code exceeds the amount of production code. One could for example look which techniques and tools developed for software maintenance and evolution can be applied for test code. Furthermore making test cases more robust can be aided for example with tools that automatically "fix" test cases when the production system changes.

3) Allow an efficient creation of test cases. Capture-replay tools already allow the efficient production of test code, but their major shortcoming is that the code they produce is un-maintainable and fragile. Future research could look how one could make capture-replay tools to produce highly maintainable test code.

4) Can be easily configured and fitted to the various software development environments and ways of working. This research direction is important as if the tools do not fit the practitioners they are likely to remain shelf ware.

5) Support an incremental delivery of test automation. The key question here is how can we go from the current state where test automation first requires high investment and then (maybe) provides high reward, to a model where test automation requires first low investment and provides a lower reward. Tools supporting such incremental adaption would help to mitigate from the current high-risk high-reward scenario.

Third, based on the survey and the literature review we highlight some differences in the views between academic literature and practitioners. Overall, the views of the practitioners and academic literature are largely aligned of the benefits and limitations of test automation. First, of two major differences was that while many academic sources provide evidence that test automation increases fault detection, still 58% of the practitioners do not agree with this. This was further explained by the open questions where a practitioner pointed out that *it is the tester that facilitates high defect detection*, and that one can get high defect detection with both manual and automated testing depending how they are used. Second, the vision that software testing would be fully automated was rejected by 80% of the respondents of our survey. Naturally, the practitioners cannot be aware of the latest development in the area of test automation, but still given the strong practitioners opposition one should reflect whether the vision is even worth chasing when practitioners think that the main test automation problems lie on more practical matters outlined above.

In future work context needs to be considered, in particular the purpose of automation (test execution, test case generation and selection, test analysis).

Acknowledgment

This work has been supported by ELLIIT, the Strategic Area for ICT research, funded by the Swedish Government.

References

[1] R. L. Glass, R. Collard, A. Bertolino, J. Bach, and C. Kaner, "Software testing and industry needs," *IEEE Software*, vol. 23, no. 4, pp. 55–57, 2006.

[2] A. Bertolino, "Software testing research: Achievements, challenges, dreams," in *Proceedings of the Workshop on the Future of Software Engineering (FOSE 2007)*, 2007, pp. 85–103.

[3] S. Berner, R. Weber, and R. K. Keller, "Observations and lessons learned from automated testing," in *Proceedings of the 27th International Conference on Software Engineering (ICSE 2005)*, 2005, pp. 571–579.

[4] B. Kitchenham and S. Charters, "Guidelines for performing systematic literature reviews in software engineering," Software Engineering Group, School of Computer Science and Mathematics, Keele University, Tech. Rep. EBSE-2007-01, July 2007.

[5] K. Berry and P. Mielke, "A generalization of cohen's kappa agreement measure to interval measurement and multiple raters," *Educational and Psychological Measurement*, vol. 48, no. 4, pp. 921–933, 1988.

[6] F. Saglietti and F. Pinte, "Automated unit and integration testing for component-based software systems," in *Proceedings of the International Workshop on Security and Dependability for Resource Constrained Embedded Systems (D4RCES'10)*, 2010, pp. 5:1–5:6.

[7] J. A. Dallal, "Automation of object-oriented framework application testing," in *Proceedings of the 5th IEEE GCC Conference and Exhibition 2009)*, 2007, pp. 425–434.

[8] R. P. Tan and S. Edwards, "Evaluating automated unit testing in sulu," in *Proceedings of the First International Conference on Software Testing, Verification, and Validation (ICST 20089)*, 2008, pp. 62–71.

[9] M. Alshraideh, "A complete automation of unit testing for javascript programs," *Journal of Computer Science*, vol. 4, no. 12, pp. 1012–1019, 2008.

[10] L. du Bousquet and N. Zuanon, "An overview of lutess: A specification-based tool for testing synchronous software," in *Proceedings of the 14th Conference on Automated Software Engineering (ASE'99)*, 1999, pp. 208–215.

[11] M. Malekzadeh and R. Ainon, "An automatic test case generator for testing safety-critical software systems," in *Proceedings of the 2nd International Conference on Computer and Automation Engineering (ICCAE)*, 2010, pp. 136–167.

[12] A. Leitner, I. Ciupa, B. Meyer, and M. Howard, "Reconciling manual and automated testing: The autotest experience," in *Proceedings of the 40th Hawaii International International Conference on Systems Science (HICSS-40 2007)*, 2007, p. 261.

[13] S. Kansomkeat and W. Rivepiboon, "Automated-generating test case using uml statechart diagrams," in *Proceedings of SAICSIT 2003)*, 2003, pp. 296–300.

[14] J. Burnim and K. Sen, "Heuristics for scalable dynamic test generation," in *Proceedings of the 23rd IEEE/ACM International Conference on Automated Software Engineering (ASE 2008)*, 2008, pp. 443–446.

[15] K. C. Choi and G. H. Lee, "Automatic test approach of web application for security (autoinspect)," in *Proceedings of the International Conference on Computational Science and Its Applications (ICCSA 2006)*, 2006, pp. 659–668.

[16] B. Haugset and G. K. Hanssen, "Automated acceptance testing: A literature review and an industrial case study," in *Proceedings of the Agile Development Conference (AGILE 2008)*, 2008, pp. 27–38.

[17] C. Liu, "Platform-independent and tool-neutral test descriptions for automated software testing," in *Proceedings of the 22nd International Conference on Software Engineering (ICSE 2000)*, 2000, pp. 713–715.

[18] D. Hao, L. Zhang, M.-H. Liu, H. Li, and J. Sun, "Test-data generation guided by static defect detection," *J. Comput. Sci. Technol.*, vol. 24, no. 2, pp. 284–293, 2009.

[19] M. F. Bashir and S. H. K. Banuri, "Automated model based software test data generation system," in *Proceedings of the 4th International Conference on Emerging Technologies (ICET 2008)*, 2008, pp. 275–279.

[20] L. Shan and H. Zhu, "Generating structurally complex test cases by data mutation: A case study of testing an automated modelling tool," *Computing Journal*, vol. 52, no. 5, pp. 571–588, 2009.

[21] K. Karhu, T. Repo, O. Taipale, and K. Smolander, "Empirical observations on software testing automation," in *Proceedings of the Second International Conference on Software Testing Verification and Validation (ICST 2009)*, 2009, pp. 201–209.

[22] T. Wissink and C. Amaro, "Successful test automation for software maintenance," in *Proceedings of the 22nd IEEE International Conference on Software Maintenance (ICSM 2006)*, 2006, pp. 265–266.

[23] W. E. Perry and R. W. Rice, *Surviving the top ten challenges of software testing: a people-oriented approach*. New York: Dorset House Pub, 1997.

[24] P. Pocatilu, "Automated software testing process," *Economy Informatics*, vol. 1, 2002.

[25] B. Pettichord, "Seven steps to test automation success," in *Proceedings of the STAR West Conference (STAR West 1999)*, 1999, pp. 136–167.

[26] M. A. Fecko and C. M. Lott, "Lessons learned from automating tests for an operations support system," *Software: Practice and Experience*, vol. 32, no. 15, pp. 1485–1506, 2002.

[27] R. Coelho, E. Cirilo, U. Kulesza, A. von Staa, A. Rashid, and C. J. P. de Lucena, "Jat: A test automation framework for multi-agent systems," in *Proceedings of the 23rd IEEE International Conference on Software Maintenance (ICSM 2007)*, 2007, pp. 425–434.

[28] C. Persson and N. Yilmaztürk, "Establishment of automated regression testing at abb: Industrial experience report on 'avoiding the pitfalls'," in *Proceedings of the 19th IEEE International Conference on Automated Software Engineering (ASE 2004)*, 2004, pp. 112–121.

[29] M. Fewster and D. Graham, *Software test automation: effective use of test execution tools*. Harlow: Addison-Wesley, 1999.