Exploring the industry's challenges in software testing: An empirical study

Garousi, V., Felderer, M., Kuhrmann, M., Herkiloglu, K., & Eldh, S. (2020). Exploring the industry's challenges in software testing: An empirical study. *Journal of Software: Evolution and Process*. Advance online publication. https://doi.org/10.1002/smr.2251

Published in:
Journal of Software: Evolution and Process

Document Version:
Peer reviewed version

Queen's University Belfast - Research Portal:
Link to publication record in Queen's University Belfast Research Portal

This is the pre-print of a paper that has been published in the Journal of Software: Evolution and Process
DOI: onlinelibrary.wiley.com/doi/full/10.1002/smr.2251

Exploring the industry's challenges in software testing: An empirical study

Vahid Garousi
Queen's University Belfast
Belfast, Northern Ireland, UK
v.garousi@qub.ac.uk

Michael Felderer
Blekinge Institute of Technology, Sweden
University of Innsbruck, Austria
michael.felderer@uibk.ac.at

Marco Kuhrmann
Institute for Applied Software Systems Engineering
Clausthal University of Technology, Germany
kuhrmann@acm.org

Kadir Herkiloğlu
Test Directorate
HAVELSAN A.Ş., Ankara, Turkey
kherkiloglu@havelsan.com.tr

Sigrid Eldh
Ericsson AB, Sweden
Mälardalen University, Sweden
sigrid.eldh@ericsson.com

Abstract:

Context: Software testing is an important and costly software engineering activity in the industry. Despite the efforts of the software testing research community in the last several decades, various studies show that still many practitioners in the industry report challenges in their software testing tasks.

Objective: To shed light on industry's challenges in software testing, we characterize and synthesize the challenges reported by practitioners. Such concrete challenges can then be used for a variety of purposes, e.g., research collaborations between industry and academia.

Method: Our empirical research method is opinion survey. By designing an online survey, we solicited practitioners' opinions about their challenges in different testing activities. Our dataset includes data from 72 practitioners from eight different countries.

Results: Our results show test management and test automation are considered the most challenging among all testing activities by practitioners. Our results also include a set of 104 concrete challenges in software testing that may need further investigations by the research community.

Conclusion: We conclude that the focal points of industrial work and academic research in software testing differ. Furthermore, the paper at hand provides valuable insights concerning practitioners' "pain" points and, thus, provides researchers with a source of important research topics of high practical relevance.

Keywords: Software testing; software industry; challenges; opinion survey

This is the pre-print of a paper that has been published in the Journal of Software: Evolution and Process
DOI: onlinelibrary.wiley.com/doi/full/10.1002/smr.2251

1. Introduction

Software testing is an important phase of software engineering to ensure developing high-quality software [1]. Even if the body of knowledge and the research literature in software testing are vast [3], still, there is a high industry need for more improvements in effectiveness and efficiency of software testing [4]. While industry and academia are working on their software testing activities in a mostly disjoint manner [5], it is often not clear what major challenges the industry is experiencing that needs more research effort from the academic community [5].

Furthermore, understanding specific challenges of the industry in software testing is regularly mentioned as an important issue in expanding the contributions and impact of software testing research in general, e.g., [6]. Based on the above background and motivations, the goal of this study is to explore and characterize industry's challenges in software testing, and the topics that industry would like to suggest as potential research topics to academia. For this purpose, we developed an online survey and a classification of testing activities (i.e., test-case design and test execution) and solicited practitioners' opinions on the challenges they face in those testing activities.

The contributions of this article are four-fold:

(1) characterizing the level (extent) of industry's challenges in different test activities;
(2) in-depth analyses and classification of the reported challenges;
(3) a set of concrete research challenges in software testing as observed by industry; and
(4) a comparison of the reported industrial challenges versus the focus of academic contributions in recent years.

Results of our study will benefit researchers and practitioners alike. As a synthesis of practitioners' opinion, researchers will benefit by better understanding industry's challenges in software testing and possibly plan to conduct some of their research activities in the context of those challenges (topics). Practitioners will benefit from the results (level of challenges and concrete challenges) by getting an overview of the state of practice and also state of challenges in the industry, and for managers which are looking for topics, which need further improvements in their test teams and/or more training of their test engineers.

The remainder of this article is organized as follows: Section 2 presents the background and related work. Section 3 describes the goal and the research method used in this study. Section 4 presents the analysis and the results. Finally, Section 5 concludes the study and states the directions of future work.

2. Background and related work

In the Software Engineering (SE) literature in general, many papers have looked into industry's challenges, e.g., [7-11]. The study [7] explored the challenges of adopting Model-Driven Engineering (MDE) in industry based on experience in two large software companies. A workshop was held in 2003 [8] to explore the challenges of software industry globalization. An experience report [9] explored and reported challenges of software verification and validation in the space industry, in the context of two companies in the European space industry.

Various books and papers have also talked about the challenges and successes of industry in conducting software testing. Three test managers from Microsoft published a book in 2008 about "*How we [they] test software at Microsoft*" [12]. The book reported the testing practices and challenges in that company. Some of the reported challenges we as follows: (1) growing number of tests as a software under test evolves: "*With each wave, most old tests carry forward and many new tests are added.*"; (2) "*with a massive number of testers spread throughout the world, the challenge [of organizing the testers] at Microsoft is monumental*"; and (3) the widespread challenges of developing test oracles.

A similar 2012 book was written by testers in Google, entitled: "*How Google tests software*" [13]. It also reported about testing practices and challenges in that company.

This is the pre-print of a paper that has been published in the Journal of Software: Evolution and Process
DOI: onlinelibrary.wiley.com/doi/full/10.1002/smr.2251

Furthermore, there is a very large amount of discussions about challenges in software testing in the grey literature among practitioners. A Google search for "*challenges software testing*", as of this writing (September 2018), returned about 185 million hits. The auto-complete feature of Google also suggested interesting phrases about this subject, as shown in Figure 1. On this topic, there have been recent discussions in the SE community about using practitioners' opinions, mentioned in the grey literature, as a source of knowledge and evidence for SE research [14-16].

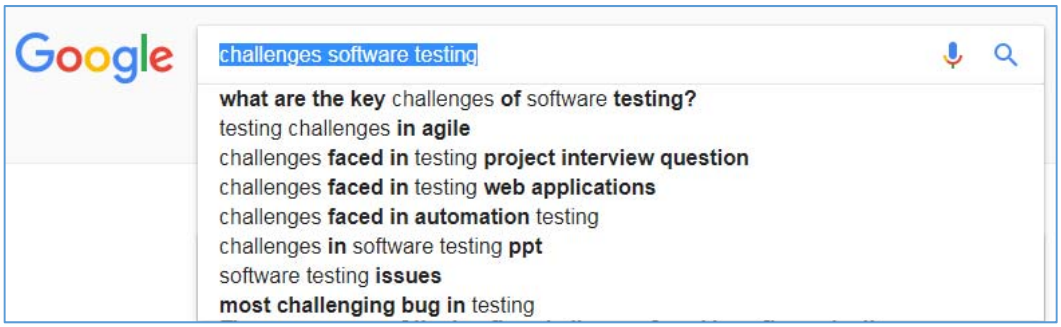


Figure 1- Discussions about challenges in software testing in the grey literature

Cem Kaner, who is a software testing researcher and advocate [17], summarized the fundamental challenges in software testing as follows: *"(1) Complete testing is impossible; (2) Testers misallocate resources because they fall for the company's process myths; (3) Test groups operate under multiple missions, often conflicting, rarely articulated; and (4) Test groups often lack [enough] skilled programmers, and a vision of appropriate projects that would keep programming testers challenged"*.

An IEEE Software paper entitled "*Software testing and industry needs*" [18] gathered the opinions of five "renowned" experts in software testing in response to the following question: Does the practice of software testing effectively meet industry needs? One of the interesting highlights from the paper was as follows: "*Effective testing is a quest and, like any quest, includes intellectual challenge, passionate debate, and the excitement of discovery*". Another co-author of the paper explored the reasons behind why deployment (transfer) of research results in software testing in (to) practice has not been widespread. She argued that: "*On one side, practitioners, who are chronically short of time or resources, tend to perceive systematic testing as a luxury. On the other, testing impacts the whole life cycle, because any testing technique presupposes adequate preparation, modeling, and documentation*", thus those could be seen as some challenges of testing in practice.

A survey on software testing practices in Canada, executed in 2010, was reported in [11], which was based on data from 246 practitioners. One question of the survey was: "*Challenges posed [by practitioners] for the research community*", which gathered 63 data points (challenges) from the practitioners, e.g., "*Automated GUI testing tools are not stable", "[How to determine the right] automation scope", "[need for] better techniques/metrics for deciding what should be tested*", and "*complexity of maintaining and documenting test plans and test results*".

To characterize industry's challenges in SE and software testing, and to bring industry and academia closer together, the authors and their collaborators have published several papers so far, e.g., [4, 19-21], which relate to the study reported in this paper. This article is another step in that line of work by the authors. In [19], we reported our experience, success stories and challenges in 10 industry-academia collaborations (IAC) projects in software testing conducted in two different countries (Canada and Turkey). The study in [22] reported an empirical analysis of challenges, patterns and anti-patterns in a set IAC projects in SE The study was based on opinion survey data from practitioners and researchers.

In [20], we presented an experience report to help systematically selecting the "right" topics for IACs in software testing, in the context of an IAC involving one industrial partner and one research team. The topic-selection process involved interaction with company representatives in the form of both multiple group discussions and separate face-to-face meetings while utilizing grounded-theory to find (converge to) topics which would be 'interesting' and useful from both industrial and academic perspectives.

This is the pre-print of a paper that has been published in the Journal of Software: Evolution and Process
DOI: onlinelibrary.wiley.com/doi/full/10.1002/smr.2251

A paper, published in the "Future of Software Engineering" conference in 2007, talked about "achievements, challenges, and dreams" in software testing research [23]. However, it contained almost no discussion of challenges specific to industry. Three authors from the Airbus Group reported in [24] the challenges in testing of autonomous systems. They especially highlight that the challenge to test autonomous systems "*to ensure their safe and fault-free behavior is not solved yet*".

Another paper explored the challenges of mobile testing in software industry using agile methods [25]. It compilid the following general challenges: Limited capabilities and device evolution; Many standards, protocols and network technologies; Variety of different platforms; Publication approval on various app stores; Specific users' needs and Short time to meet market requirements.

Challenges of motivating software testing personnel were explored in another paper [26]. The paper [27], written by a practitioner, discussed challenges of industrial-strength model-based testing. The challenges listed in the study included: (1) conformance in the synchronous deterministic case; (2) conformance in presence of non-determinism; (3) requirements tracing to the model; (4) automated compilation of traceability data; and (5) test case selection according to criticality.

A group of authors form academia and industry (from the Thales Group) [28] presented the testing challenges of maritime safety and security systems-of-systems. The paper discussed that a first group of testing challenges is raised by the need of accepting the integration of such large systems, and the ability to reconfigure them at runtime. A second group of testing challenges comes from the fact that, generally, not all the sub-systems are designed along the same kind of architecture (e.g. client-server vs. publish-subscribe architecture).

Authors in [29] and [30] reported the experiences and challenges of introducing risk-based testing in an industrial project, on a Web-based application part of a large information system. The challenges reported were rather quite fine-grained and specific, e.g., since the application's architecture was composed by a hierarchical component structure, the risk ratings were most often subjective estimates and the interview participants felt not confident to express these estimates on a more fine-grained scale other than high, medium or low. The paper reported in [31] explored the challenges of testing services and service-centric systems.

In summary, we should highlight that in almost all the above papers, discussions were purely based on authors' opinion and not evidence-based by systematically gathering and analyzing data from industry as done in this paper.

To characterize what industry wants from academia in software testing, we reported the results of an opinion survey in [4], which included data from 28 practitioners. The current article is a major extension of [4], both in terms of the size of the dataset and also the coverage of research questions (to be discussed in Section 3.1). The current article is also related to our experience report in [20], in which practitioners challenges in testing, gathered via group discussions and separate face-to-face meetings in the context of one industrial partner, were analyzed using grounded-theory, to systematically select the "right" topics for IACs in software testing. In this article, instead of one industrial partner, we aim to gather industrial challenges in testing from a larger population of practitioners.

3. Research goal and method

This section presents the study's goal, research questions studied and outlines the research method implemented for this study.

3.1 Goal and research questions

The goal of this study is to explore and characterize industry's challenges in software testing, so as to be able to identify areas which may need further work by researchers, and to encourage more collaborations with the industry on real industrial needs in testing. This "exploratory" study aims at identifying research topics of practical relevance to inspire and strengthen IACs in software testing.

We should emphasize that we are not conveying in any way that researchers should abandon "basic research" in software testing, and to *only* focus and work on industrial challenges in software testing. We believe that there is value for both basic research and (industry-driven) applied research in software testing and in SE in general [32]. Basic research is fundamental,

This is the pre-print of a paper that has been published in the Journal of Software: Evolution and Process
DOI: onlinelibrary.wiley.com/doi/full/10.1002/smr.2251

foundational, theory-oriented and curiosity-driven, while applied research is practical, goal-directed, solution-oriented, and mission-driven [32].

This issue was also discussed in a 2016 book entitled "*The new ABCs of research: achieving breakthrough collaborations*" [32] which advocated for more IACs in all areas of science. One quote in that book stated that: "*Combining applied and basic research produces higher impact research, compared to doing them separately*". We believe that applied and basic research in software testing can be indeed combined and identifying industry's challenges is an important step for doing so.

Based on the above goal, we raise and investigate the following research questions (RQs):

- **RQ 1:** Challenges in test activities:
  - **RQ 1.1:** To what extent do practitioners experience challenges in different test activities?
  - **RQ 1.2:** Does more work experience of a given practitioner in software testing lead to "less" challenges faced by him/her in test activities?
- **RQ 2:** Classification and root-cause analysis of challenges:
  - **RQ 2.1:** What is the nature of challenges? How many of the reported challenges are related to lack of knowledge/training, resource constraints and how many are "real" research challenges? We have observed in our past IAC projects [19] that some of the practitioners' challenges are simply due to lack of knowledge/training, e.g., when a tester does not know about a particular black-box test-design technique, while some challenges are due to resource constraints, e.g., when a test team does not have the time to conduct systematic test-case design. Finally, some challenges are real research challenges, i.e., those which require research efforts to solve the problem, e.g., when a test team cannot figure out when (in the software development lifecycle) and what (which test cases) to automate in testing [33, 34].
  - **RQ 2.2:** What are the concrete software testing challenges experienced by practitioners? For this RQ, we synthesize the challenges reported using the findings of RQ 2.1 and present them as a list of suggested research topics.
- **RQ 3:** How do industrial challenges relate to the recent focus of the academic community? It would be interesting to assess the relationship between industrial challenges and the recent focus of the academic community, as to see the extent of overlap between them.

As discussed above, the current article is an extension of an earlier conference paper [4], in terms of coverage of RQs: that paper [4] only covered RQ 1.1 and only presented the "raw" data for RQ 2 without classifying the challenges and determining their nature (challenges due to lack of knowledge, training, low maturity or awareness about testing, due to resource constraints, versus concrete/real software testing research challenges), as we reported in Section 4.3. However, RQ 1.2, RQ 2.2 and RQ 3 cover new material added in this article. Furthermore, we extended related work and provide a more in-depth discussions throughout the article and especially also in Section 4.5.

3.2 Research method

To address the study's goal and research questions, we designed and conducted an online opinion survey to collect the opinions from practitioners.

In designing the survey, we benefitted from the survey guidelines in SE, e.g., [35, 36]. We followed the recommended guidelines from [35] for the following aspects of the survey process: (1) identify the research objectives, (2) identify & characterize target audience, (3) design sampling plan, (4) design & write questionnaire, (5) pilot test questionnaire, (6) distribute questionnaire, and (7) analyze results and write the article. We also used the recommendations from [37, 38] with respect to characterizing units of observation, units of analysis, establishing the sampling frame and recruitment strategies. For designing and conducting the opinion survey, we also relied on our experience in designing and conducting surveys in the past, e.g., [10, 11].

In Section 3.2.1, we provide an overview of the survey instrument used, and in Section 0, we provide details about the data collection procedure and give an overview of the study's population.

This is the pre-print of a paper that has been published in the Journal of Software: Evolution and Process
DOI: onlinelibrary.wiley.com/doi/full/10.1002/smr.2251

3.2.1 The survey instrument

Table 1 provides an overview of the questionnaire, including question categories and the answer types. Since this survey addresses practitioners, we kept the number of questions to a minimum to ensure getting as many responses as possible [35]. The questionnaire has three question categories. In the first category, we asked the participants to provide demographic and background information (questions D1 and D2).

Table 1-Questionnaire used for the survey.

Group	Question number	Question text	Answer type
Demographics (D)	D1	What is your country of work/residence?	List of countries
	D2	How many years of work experience do you have in software testing (such as 1, 2, ...)? (was added in phase 2)	Integer
Challenges in testing (Q)	Q1	What is the level of challenge that you have in each of the following types of testing activities, in which you would consider collaborations with software testing researchers? (descriptions are provided below, see Table 2)	5-point Likert Scale
	Q2	Please list your concrete challenges in "Test-case design (criteria-based)"	Free text
	Q3	Please list your concrete challenges in "Test-case design (based on human expertise)"	Free text
	Q4	Please list your concrete challenges in "Test scripting"	Free text
	Q5	Please list your concrete challenges in "Test execution"	Free text
	Q6	Please list your concrete challenges in "Test evaluation"	Free text
	Q7	Please list your concrete challenges in "Test-result reporting"	Free text
	Q8	Please list your concrete challenges in "Test management"	Free text
	Q9	Please list your concrete challenges in "Test automation"	Free text
	Q10	Please list your concrete challenges in "Other testing activities"	Free text
Feedbacks (F)	F1	Please feel free to indicate any comments or suggestions that you may have	Free text

The actual research questions (Section 3.1) of our study are addressed in the second question category, which includes 10 questions (Q1-Q10). Question Q1 is concerned with a set of test activities and the challenges participants face in these activities. As shown in Table 2, we synthesized a set of nine test activity types by reviewing popular textbooks on software testing, e.g., [39], guidelines of industry certification bodies on testing, e.g., the International Software Testing Qualifications Board (ISTQB; [40]), discussions with several senior test engineers in industry, and also our recent systematic mapping studies, e.g., [41].

For each activity type, the questionnaire asked participants to rate the level of challenge and the need for collaborations with academics (researchers) on a 5-point Likert scale (rubric), which was defined as follows:

- 0 = No challenges at all and thus no need for collaborations with academia (we are doing very well)
- 1 = Some challenges in the testing activity
- 2 = Average level of challenges in the testing activity
- 3 = Many challenges
- 4 = Lots of challenges (collaborations with academia needed)

Having rated the level of challenges, questions Q2-Q10 asked the participants for concrete challenges for each specific activity types. This detailed information was used to collect specific research topics in the nine test activities that practitioners wanted to pose for the research community. The last question category asked the participants to provide feedback if they had any (question F1).

This is the pre-print of a paper that has been published in the Journal of Software: Evolution and Process
DOI: onlinelibrary.wiley.com/doi/full/10.1002/smr.2251

Table 2-Overview of the test activity type evaluated in the survey

No.	Activity Type	Description
1	Test-case design (criteria-based)	Designing test suites (set of test cases) or test requirements to satisfy coverage criteria, e.g., line coverage
2	Test-case design (based on human expertise)	Designing test suites (set of test cases) based on human expertise (e.g., exploratory testing) or other engineering goals
3	Test scripting	Documenting test cases in manual test scripts or automated test code
4	Test execution	Running test cases on the software under test (SUT) and recording the results
5	Test evaluation	Evaluating results of testing (pass or fail), also known as test verdict
6	Test-result reporting	Reporting test verdicts and defects to developers, e.g., via defect (bug) tracking systems
7	Test management	Encompasses activities related to test management, e.g., planning, control, monitoring, etc.
8	Test automation	Automating any test activity
9	Other testing activities	Includes activities other than those discussed above, e.g., regression testing, and test prioritization

When designing the survey, we were aware of common threats to internal validity in surveys [35, 36], e.g., (1) Lack of a common understanding of the questions among subjects; (2) Bias introduced by researchers in selection of or framing of the questions (for instance, leading questions, omissions, and so forth); (3) Insufficient questions to test consistency for constructs across multiple questions; (4) Ordering of the questions [42]. We took the following decision w.r.t. each of them, with the goal of reducing their negative impacts on internal validity.

To ensure that our survey subjects (participants) would come to a common understanding of the survey questions, we made sure the questions were grounded in conventional software-testing concepts that practitioners were more likely to relate to and understand. This was achieved by using the terminology and concepts used/defined in popular textbooks on software testing, e.g., [39] as discussed above, and also the guidelines of industry certification bodies on testing, e.g., ISTQB [40]. Furthermore, to ensure the quality and clarity of the survey questions, we piloted the questionnaire with five test engineers in the industry. All five of them mentioned that the questions and the terminology used were already clear. One feedback that we received during the piloting phase was to provide a brief explanation of the concepts for each survey question, which we took into account to improve the questionnaire accordingly. We provided the short definitions as a reminder to prevent potential confusions about terminology and to ensure that participants clearly understood the questions. We show a partial screenshot of the finalized online survey in Figure 2, in which the reader can see the brief explanations.

The second common threat in the above list is possible (unwanted) bias introduced by researchers in selection of or framing of the questions. An online article in Wikimedia [43] highlights the importance of suitable framing of survey questions as follows: "*When people misunderstand survey questions, they often don't answer them the way the survey creator intended. By the time the survey creator realizes this, it's usually too late*". Two example suggestions provided in [43] are: "*When asking multiple-choice questions, make sure you provide choices for all the major types of answers you expect to receive, at the right level of specificity*", and "*Ask open-ended questions in a way that encourages people to elaborate, rather than give one word answers*". By keeping our questions short but concise, we believe we framed the questions with good quality. We ensured designing the wording of our questions in "neutral" tones, e.g., "*What is the level of challenge that you have in each of the following types of testing activities?*". By this, we wanted to ensure that respondents would fairly provide the actual level of challenges that they have been experiencing in their test activities.

Another design issue (trade-off) for the survey was its length versus having multiple questions for the same construct, which could increase the survey's reliability" [36] (the third common threat in the above discussion). As we discussed above, since this survey addresses practitioners, we kept the number of questions to a minimum to ensure getting as many responses as possible [35]. Thus, we designed the survey instrument to be short, which is an advantage in obtaining compliance but we understood the drawback that it lacked multiple questions for the same construct. A standard test of question reliability is Cronbach's alpha [36], or the correlation among responses to the related questions. We were aware of that the inability to apply these metrics would limit the internal validity, but in the context of this trade-off, we decided to keep the number of questions to a minimum to ensure getting as many responses from practitioners as possible.

This is the pre-print of a paper that has been published in the Journal of Software: Evolution and Process
DOI: onlinelibrary.wiley.com/doi/full/10.1002/smr.2251

The last common threat in the above list is the ordering of the questions, a topic about which specific papers have been published, e.g., [42, 44]. The expression "order effect" refers to the well-documented phenomenon that different orders in which the questions (or response alternatives) are presented may influence respondents' answers [44]. This issue was not complex to decide for us since we essentially had three groups of questions, the last being just a feedback, leaving us with two main question groups only (demographics, and challenges in testing), as shown in Table 1. Most SE surveys starts with demographics questions thus most respondents are quite used to seeing demographics questions first. Thus, we followed this routine. The next issue was to decide on the suitable order of questions in the "challenges in testing" group. We had in this group a question asking for the level of challenges for each type of test activity, and the second sub-group asking concrete challenges for each type of test activity. The design was again quite straight-forward since it was logical to first ask the more general Likert-scale questions on the level of challenges, and then the concrete challenges for each type of test activity.

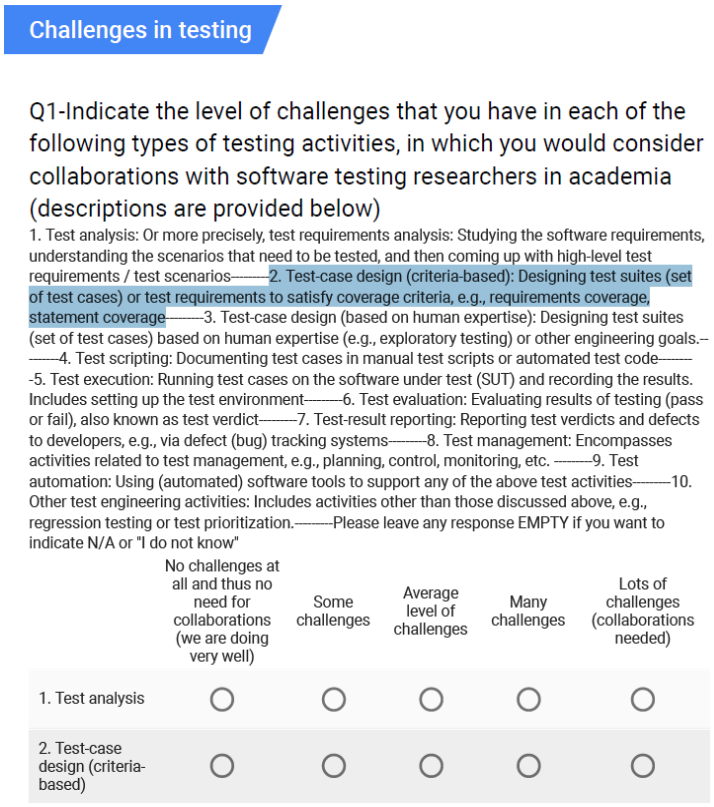


Figure 2- A screenshot of the online survey, which shows the explanations of the concepts provided to respondents, to ensure a common understanding of the questions

After careful design of the survey based on the above discussions, we implemented the questionnaire of Table 1 in the Google Forms online tool and then proceeded to data collection, which we discuss next.

3.2.2 Data sources and data collection

We executed the survey in two rounds (phases). The first round was executed during Fall 2016 and the second round was executed during Fall 2017. Compared to phase #1 of the survey, we added one extra question in phase #2 to improve the aspects that we gathered the opinions about: years of work experience in software testing.

The sampling method used for the survey was "convenience sampling", which is the most widely used survey approach in SE [45]. Albeit its drawbacks, as a non-probability sampling method, convenience sampling has been used in many studies

This is the pre-print of a paper that has been published in the Journal of Software: Evolution and Process
DOI: onlinelibrary.wiley.com/doi/full/10.1002/smr.2251

and has been found an acceptable approach [46], e.g., in our previous Canada-wide surveys [47, 48] and also by other researchers, e.g., [49]. In convenience sampling, "*Subjects are selected because of their convenient accessibility to the researcher. These subjects are chosen simply because they are the easiest to obtain for the study. This technique is easy, fast and usually the least expensive and troublesome. ... The criticism of this technique is that bias is introduced into the sample.*" [50]. In all forms of research, it would be ideal to test the entire population or a large ratio of it, but in most cases, the population is just too large that it is impossible to include every individual. The challenges are to identify the target population and obtain a suitably random sample. Albeit its drawbacks and bias in the data, this does not mean that convenience sampling is generally inappropriate. For example, Ferber [51] refers to the exploratory, the illustrative, and the clinical situations in which convenience sampling may be appropriate. Convenience sampling is also common in other disciplines such as clinical medicine and social sciences. We realize and acknowledge that while convenience sampling could have some limitations,, however given its wide-spread use in the literature, its advantages [45], and the limitation of our resources in this project to conduct more sampling approaches, we decided to use it in our data collection.

We sent invitations to testing practitioners in our contact network. We also posted messages in our social media (Twitter) profiles. In total, we received 28 responses in the first round of the survey, as reported and analyzed in [4], and 30 additional responses were received in the second round of the survey's execution. We did not use "snowball" sampling [35] in our data collection, since it could potentially amplify bias in sampling. Snowball sampling is a non-probability sampling technique where existing study subjects recruit future subjects from among their acquaintances [35]. To prevent this bias, we did not explicitly ask the respondents to also forward our survey invitation in their networks.

To complement the data that we gathered from the online survey, we augmented the dataset with data from one additional data source. Two of the authors have been involved in a set of 14 IAC projects recently [19], in which concrete industrial challenges were identified and solved via joint IAC projects. Based on these 14 IAC projects, we created 14 data points and added them to the dataset. Using qualitative coding [52], we mapped the challenges addressed in these projects to our classification of test activities as listed in Table 2. For example, given a project with the following need: "*Need for a systematic approach to decide what test cases to automate in software testing*", we mapped it to test activities "Test automation" and "Test management", with Level-of -Challenge=3 in the Likert scale (out of 4). Needless to say, for the 14 data points generated from those 14 projects, we only filled the dataset under the challenges-in-testing columns (see the questionnaire design in Table 1), and left the demographics (D) and feedbacks (F) sections empty. Such an approach did not have any negative impact on the survey data, since the challenges-in-testing columns were consider the must-to-fill columns and the other columns were also used as optional data. We show in Table 3 the characteristics of the 14 IAC projects and the data points generated from them. Readers can follow how we have derived the level of challenges (Likert scale 0-4) from the industrial need of each project.

In total, we collected 72 (28+30+14) data points from the two survey phases and our past IAC projects, as discussed above. We believe that the multiple data sources provide different angles on industry's challenges in software testing as well as external (generalization) validity of our dataset and the study [53] as findings are drawn not from one data source only, but from a variety of inputs.

To provide full traceability, replicability and transparency, the full versions of the survey questions (for both phases) can be found online in [54] and [55]. The dataset can also be found online in [56].

Table 3-Three selected examples from the 14 IAC projects and the data points generated from them (when a cell is empty, the corresponding level of challenge is 0)

Project ID	Industrial need in the project	Level of challenges in each test activity type (Likert scale 0-4)								
		1. Test-case design (criteria-based)	2. Test-case design (based on human expertise)	3. Test scripting	4. Test execution	5. Test evaluation	6. Test-result reporting	7. Test management	8. Test automation / tools	9. Other testing Challenges
CA5-testing-Pason	Need for a systematic approach to							3	3	

This is the pre-print of a paper that has been published in the Journal of Software: Evolution and Process
DOI: onlinelibrary.wiley.com/doi/full/10.1002/smr.2251

	decide what test cases to automate in software testing,									
CA6-testing Siemens IBM	Need for stress testing of distributed real-time systems based on UML models	3								
TR1-testing-YTB	Need for systematic software performance testing									3

4. Results and analysis

This section presents the study's results. We first provide an overview of the demographics, followed by the elaboration of the research questions, recommendations, and a discussion of the potential threats to validity.

4.1 Demographics

The question D1 asked for the respondent's country of residence to get a sense of the distribution (and the representativeness) of our dataset. Figure 3 shows the answers per country represented in the dataset. In total, we see that eight countries are present in our dataset, and that the authors' countries of residence have influenced the geographical composition of the dataset. In our past surveys on testing practices in Canada and Turkey [10, 11], and by comparing them to results of other surveys in the literature (e.g., [57, 58]), we have not observed major differences in the general profiles of software testing communities and their challenges in different countries. Hence, we consider the dataset sufficiently representative in terms of general challenges in software testing.

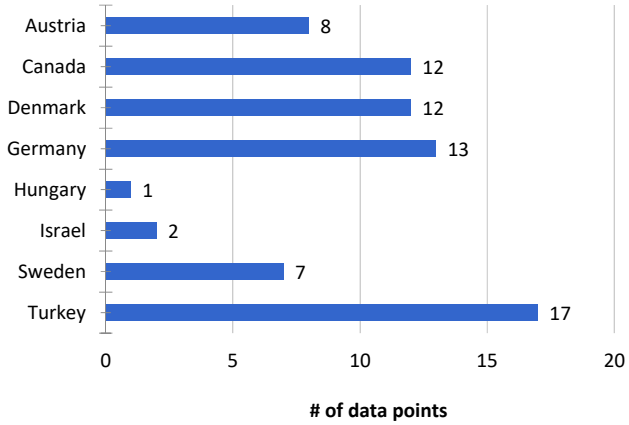


Figure 3- Breakdown of data points by countries

The question D2 of the survey (Table 1) asked for the respondent's years of work experience in software testing. Figure 4 shows the histogram of data points regarding this question. As we can see, the majority of the respondents have between 5 and 15 years of work experience. Four respondents had more than 20 years of experience. Thus, we can say that the pool of respondents provides a good mix of both junior and senior software testers.

This is the pre-print of a paper that has been published in the Journal of Software: Evolution and Process
DOI: onlinelibrary.wiley.com/doi/full/10.1002/smr.2251

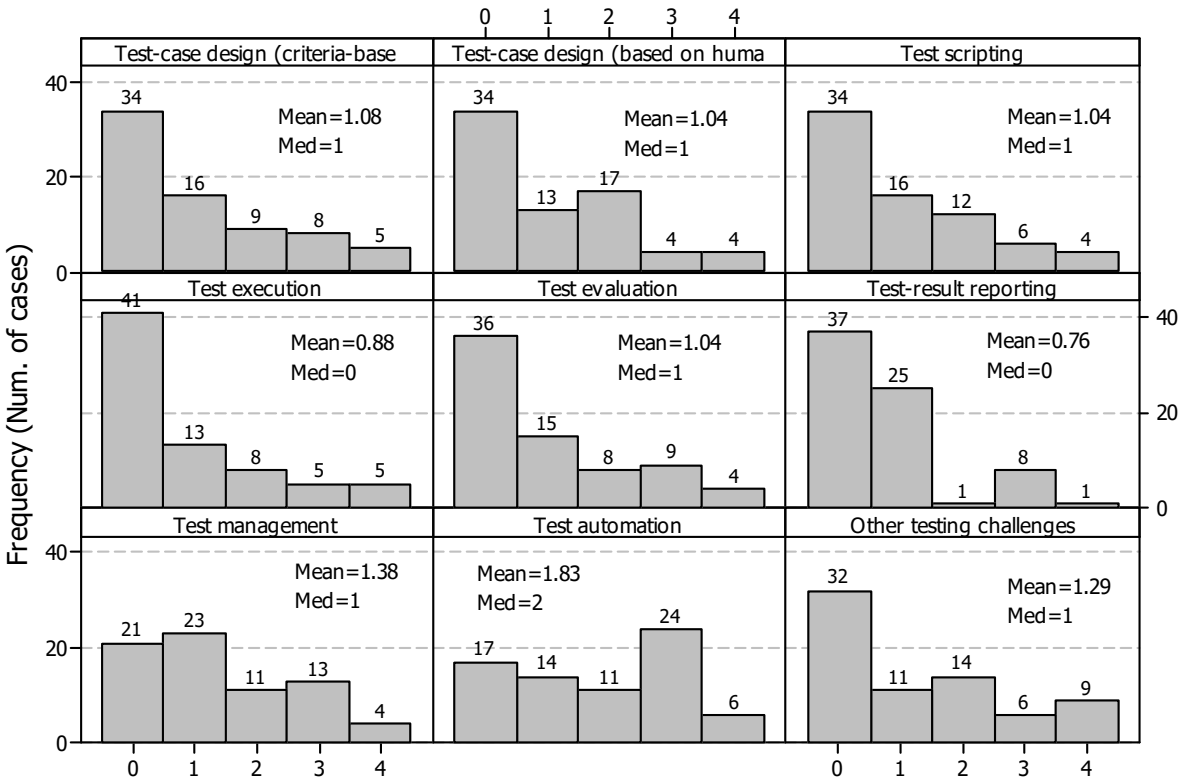
Figure 4- Breakdown of data points by years of work experience in software testing

4.2 RQ 1: Challenges in test activities

In the following, we discuss the research questions 1.1 and 1.2.

4.2.1 RQ 1.1: Level of challenges in test activities

For RQ 1.1, we analyzed the level of challenge for each test activity type. Figure 5 shows the histograms of the level of challenges for each type of test activity. Figure 5 provides the distribution including the mean (average) and the median for each type of test activity (only for illustrative purposes).



This is the pre-print of a paper that has been published in the Journal of Software: Evolution and Process
DOI: onlinelibrary.wiley.com/doi/full/10.1002/smr.2251

Figure 5 – Histograms of the level of challenges in each testing activity (0=no challenges ... 4=lots of challenges)

Note that, only for illustrative purposes, we converted the 5-point Likert scale values into numerical values in the range from 0 to 4 in Figure 5. Since we are dealing with ordinal variables in this case (the Likert scale), we were aware of the nature of ordinal variables and limitation of analysis using these qualitative data. As it has been widely discussed in the literature and the statistics community, e.g., in [59], when dealing with ordinal variables, analysis using average alone is usually not appropriate, and is only permissible with certain caveats, e.g., when the ordinal data are normally distributed. But as we can see in Figure 5, the ordinal data in our case are not normally distributed.

The widely-accepted recommendation in the statistics community is to "*use the mode for nominal data, the median for ordinal data, and the mean for metric data*" [60]. To analyze the level of challenges in an aggregated manner, we calculated the average (mean), median and mode values for level of challenges, and show them in Figure 6.

In Figure 5 and Figure 6, the average values of challenges in all test activity types are between [0.76, 1.83] (out of 4) and their median values are all either 1 or 0. This shows that, for most of the test activity types, the majority of the respondents reported low levels of challenges (i.e., see the number of '0' and '1' entries in the histograms). Furthermore, we observe that all histograms in Figure 5 tend to be left skewed. We argue that a combination of the following possibilities plays a role in this outcome:

1. Relatively high technical maturity/capability of respondents and their teams in the testing activities;
2. Simplicity of testing tasks in their contexts; and/or:
3. Respondents being unaware of having challenges or unaware of being able to conduct testing in a more effective/efficient manner

While any of the above possibilities may play a role, we elaborate on each of the above possibilities next. As one would expect, there is a wide spectrum of technical maturity/capability among different test engineers, teams and companies, e.g., [61]. Even, there are systematic approaches to measure test maturity in the industry [62], using models such as Test Maturity Model integration (TMMi) [63]. Thus, it is natural to see some respondents reporting no or low level of challenges and some report high level of challenges in their testing projects (possibility 1. above).

Furthermore, industrial domains, contexts and complexity of SUTs vary among different practitioners, e.g., testing a mobile app game would perhaps be less complex than testing a safety-critical control system. Such factors would expectedly impact the level of challenges among different respondents (possibility 2. above).

Another possibility (3. above) could be that a tester would be unaware of having challenges, due to not having a chance to do self-refection on her/his technical activities, or unaware of the existence of better test approaches for conducting testing in a more effective/efficient manner.

When we look at the data for different types of test activities in Figure 5 and Figure 6, we see that test management, automation and "other" test activities have been generally rated higher by practitioners concerning the level of challenges; compared to the other activity types test-case design, scripting, execution, evaluation (the test oracle problem) and test-result reporting. We provide next some explanations for the differences among the levels of challenges for different types of test activities.

Test execution is seen less of a challenge, since one can assume that the more mature testing is, the more automated it is and, thus, test execution becomes less challenging for a practitioner. In the presence of test automation, the challenge usually "shifts" from the actual test execution to test scripting, i.e., how to best "program" the test scripts and to co-maintain them as the SUT changes, a topic which has been phrased as "test-code engineering" [64].

Test scripting is technically separate from test-case design, but in industry, as per observations of authors, most practitioners conduct test-case design while implementing test scripts. As studies such as [65] have shown, most practitioners know the names of most test-case design techniques, but many cannot or, even if they can, do not apply them (rigorously), which indicates a general lack of knowledge, lack of awareness regarding systematic test-case design techniques, or lack of resources to apply them. For example, test-design techniques such as search-based testing and mutation testing are novel

This is the pre-print of a paper that has been published in the Journal of Software: Evolution and Process
DOI: onlinelibrary.wiley.com/doi/full/10.1002/smr.2251

test design technologies, but as per our international experience [19, 66], they have limited usage in industry in general. Other studies have also identified such as phenomenon, e.g., [5, 67].

As test automation is a key aspect of agile software development practices, many practitioners automate their test suites to meet short delivery cycles [68, 69]. Even if a practitioner has many years of experience in test automation, the problem cannot be viewed as fully solved from a practitioner's point of view, since many challenges remain in the "full" automation of all test activities for a given SUT. Practitioners may especially perceive test automation as challenging as it is quite technically compared to many other test activities. The high degree of technicality also implies that test automation is also a relative dynamic area because the underlying system and test technologies as well as the system itself continuously change. Furthermore, the ever increasing amount of automated test cases induces additional changes, for instance to prioritize test cases and to handle test case dependencies.

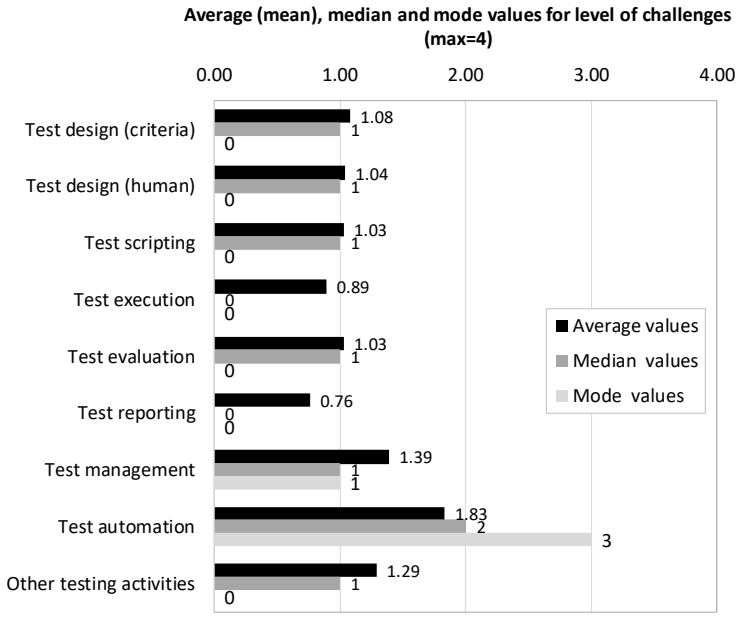


Figure 6 - Average (mean), median and mode values for level of challenges (max=4)

4.2.2 RQ 1.2: Impact of work experience on observing challenges

For RQ 1.2, we were curious to see if there are any correlations among a practitioner's level of challenge with her/his years of work experience in testing. The expectation is that with more years of work experience in testing, a typical tester is expected to gain more experience/knowledge about testing and thus faces less challenges. We aggregated all the level of challenges on the nine test activity types (see Table 2) for a respondent to a single measure by summing up all the nine challenge values. Since each challenge level in each test activity type could be between 0 and 4, the sum values could be between 0 and 36. Figure 7 shows the correlation of the sum of challenges with the number of years of work experience. Only 27 of the respondents provided their years of work experience and thus Figure 7 shows 27 data points.

In Figure 7, we also show the quadratic fit (regression) curve, and the correlation values and the respective p-values using both Pearson and Spearman correlation. Since one of the variables (sum of challenges) is an ordinal data, the Spearman Rho should be mainly used for correlation analysis. Although the points are not clearly scattered on a line or a curve, we can see that the regression fit curve indicates a slight decrease of the sum of challenges with increasing work experience, which is supported by a weak negative correlation (-0.06 and -0.13 for the two correlation metrics) between the sum of challenges and the years of work experience in testing. When starting as a software tester, an engineer usually observes a certain number of challenges, as in every new field. However, even experienced testers still observe challenges, but to a slightly lower

This is the pre-print of a paper that has been published in the Journal of Software: Evolution and Process
DOI: onlinelibrary.wiley.com/doi/full/10.1002/smr.2251

extent. This seems like an interesting finding because it suggests that day to day work of software tester does not generally provide the necessary conditions for improving expertise. This is where academic assistance might fill the gap, e.g., by education, corporate training or via industry-academia collaborations in software testing [19].

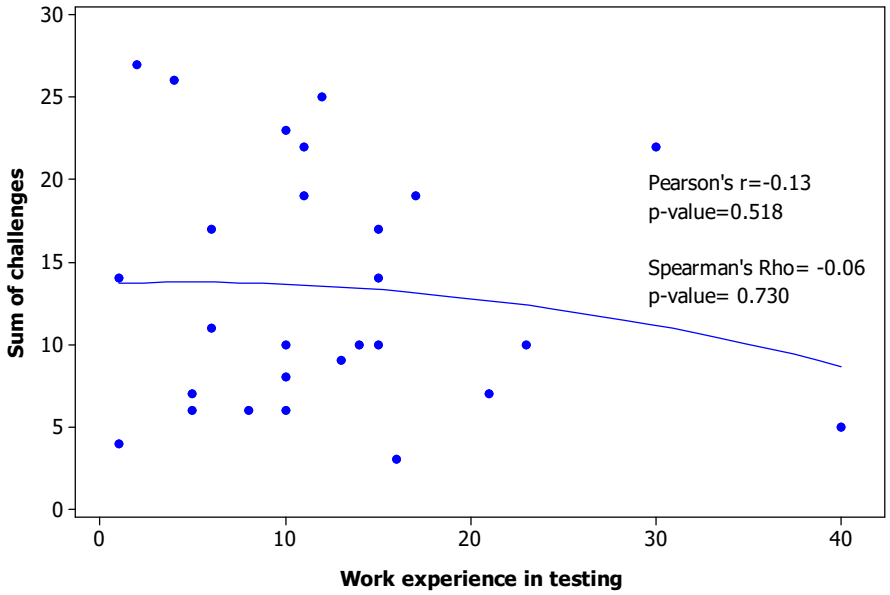


Figure 7 - Correlations between level of challenges versus years of work experience

4.3 RQ 2: In-depth analysis of challenges and their nature

For RQ 2, we raised two sub-questions, which are analyzed and addressed next. In RQ 2.1, we classify the (raw) reported challenges into several categories, e.g.: (1) those related to lack of knowledge/training, (2) resource constraints, and (3) "real" research challenges. In RQ 2.2, we synthesize and present the concrete research topics posed by industry.

4.3.1 RQ 2.1: Classification of challenges

Once we had the dataset, we looked at the raw reported challenges and noticed that, as expected, different respondents have used different terminologies in reporting their challenges. The reported data were qualitative, and thus, we had to properly process and synthesize the qualitative data before reporting.

To process and classify the reported challenges, we used qualitative coding [52], which is the widely-used approach to process qualitative data. In our process, we conducted "open" and "axial coding" [52]. Through qualitative coding, the following classifications "emerged" from the reported challenges: (1) concrete/real software testing research challenges; (2) non-technical challenges or challenges outside testing; and (3) unclear responses, too generic, or not challenges really. We furthermore classified non-technical challenges or challenges outside testing into these sub-categories: (1) challenges due to lack of knowledge, training, low maturity or awareness about testing, (2) challenges due to resource constraints, (3) technical challenges outside testing (e.g., requirements), and (4) other challenges. Qualitative coding of the reported challenges was conducted by one researcher and then peer reviewed by all other involved researchers (authors). There were very few disagreements which were discussed until consensus was reached for all cases.

To provide insights on how we conducted the qualitative coding, we provide a few examples next. For the challenges under the test scripting activity type, one respondent had provided: "*We are not mature enough with [test script] documentation. Not managed properly. It would be nice to get additional manpower to help*". To us that statements were clearly not real software testing research challenges. We thus classified them as follows: "*We are not mature enough with [test script] documentation. Not managed properly*." was moved to the category: "Due to lack of knowledge, training, low maturity or

This is the pre-print of a paper that has been published in the Journal of Software: Evolution and Process
DOI: onlinelibrary.wiley.com/doi/full/10.1002/smr.2251

awareness". Furthermore, we moved the provided challenge: "*It would be nice to get additional manpower to help*" to the category: "Due to resource constraints".

Another example for our qualitative coding approach is as follows: one respondent had provided the following comment under challenges for test-case design (criteria-based): "*Communication between developers and clients can be a problem*". This phase to us clearly was not a real software testing research challenge. We thus moved it under the classification of non-technical challenges outside testing.

Based on the result of our qualitative coding process, we provide the breakdown of the challenge categories in Figure 8. In total, 168 challenge items were provided by the 72 data points, i.e., a single data point provided by a practitioner could contribute zero or many challenges. From the 168 reported challenges and respective explanations, we could classify 104 (61.9%) as concrete/real software testing research challenges in one of the nine test activities and 44 (26.2%) as challenges outside testing (see Figure 8). The remaining 20 items were no real challenges.

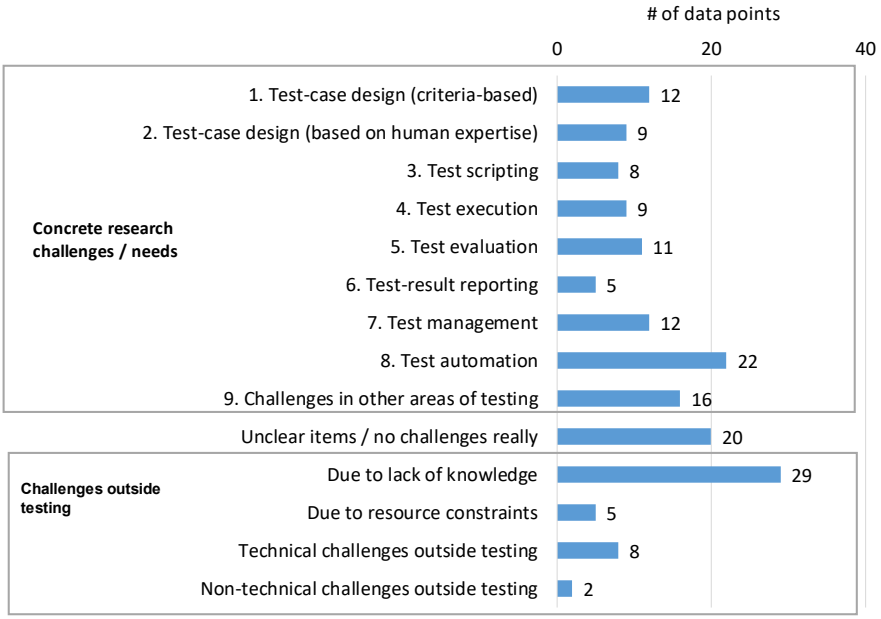


Figure 8 - Classification of the reported challenges

As shown in Figure 8, 44 out of 168 challenges (26.2%) were classified as challenges outside testing. Of those challenges, 29 were identified reporting challenges due to lack of knowledge and training, as for instance stated by the participants: "*Project managers lack basic knowledge about and are unaware of test automation*", "*We are not mature enough with test documentation*", "*[Test process] not managed properly*", "*we need expertise in automated software testing*", "*there is a need of education of developers and testers", "Basically [most] practitioners are uneducated in testing",* and *"People know the theory and name of test design techniques, but very few can translate this into an applicable automated test".*

It is unquestionable that SE education (e.g., [70-72]) and software testing education and training (e.g., [73-75]) have been important and active areas of research and practice. However, from our dataset, we can infer that, still, the community sees the need for better software testing education and training, as many practitioners still admit to not have reached high test maturity in their teams, which is mostly due to lack of education and training. There are large international associations working on such activities, such as the International Software Testing Qualifications Board (ISTQB; [40]), but more effort regarding education and training is needed; ideally by integrating testing education into modern SE and computer science curricula as well as by making testing knowledge accessible to practitioners, e.g., via online courses.

This is the pre-print of a paper that has been published in the Journal of Software: Evolution and Process
DOI: onlinelibrary.wiley.com/doi/full/10.1002/smr.2251

Another five statements from the challenges outside testing reported challenges due to resource constraints, such as: "It *would be nice to get additional manpower to help [in testing]", "Time is a problem. So if the academia can help with even quicker [test] processes, it would be great", "Insufficient time allowed [for testing]",* and *"Testing team not given sufficient time*".

Eight statements from the challenges outside testing were identified reporting technical challenges outside testing (e.g., requirements): "*Poor (low quality) requirements", "compromising between the stakeholders", "Bad shaped requirements affect the test-case design and the processes", "User stories or requirements sometimes are not testable",* and *"understanding and improving testability on various levels and artifacts"*. Finally, two statements reported non-technical challenges outside testing, specifically: "*Test politics (resistance among testers and developers)",* and *"Communication between developers and clients can be a problem"*.

Of the reported challenges, 20 (11.9%) were unclear responses, too generic, or not challenges at all, which we grouped in their respective group, e.g., *"Finding good oracles w.r.t. requirements", "Customers don't want to pay for tests", "Acceptance Testing, and Behavior-Driven Development", "We need help in creating a general structure that is able to get a superior view about the whole",* and *"Non-functional tests approaches. Better metrics in test"*.

Once we partitioned the set of reported challenges, we had the set of reported concrete research challenges, which we report next.

4.3.2 RQ 2.2: The set of reported concrete research challenges

As discussed above, we looked at the type/nature for each of the 168 reported challenges and their explanations. In total, we classified 104 (61.9%) under real software testing research challenges using the nine test activity types (Figure 8). Classification shows that the top-3 challenging areas are: (1) test automation (reported by 22 data points), (2) challenges in the group of "other" testing areas (reported by 16 data points), and (3) test management and test-design (each in 12 data points). In general, we saw that the higher level of challenges each participant had reported, the more concrete research challenges she/he reported.

Table 4 provides a subset of the research topics posed by industry for each test activity type. Note that, for brevity, we provide only a subset of the topics in Table 4, while the entire list of all the posed research topics is available in the online dataset [54]. As we can see, a variety of research topics is suggested by practitioners, e.g., reusable test-case design, and optimizing the number of test cases. We would like to invite the research community to consider this list as catalog for potential topics to work on. However, on many of these topics research has already been performed. Is it has also be further investigated why the available research results are not accessible or not applicable for practitioners.

Table 4-A subset of specific research topics posed by practitioners

Test activity	Example research topics suggested
Test-case design (criteria-based)	• Need for better test-case data generator tools • Need to check redundancy among test cases • What is the minimum test that can test the maximum number of requirements when we have a reasonably mature platform? • Reusable test-case design
Test-case design (human expertise)	• Dependence of quality of exploratory testing on human testers • Exploratory testing is hard to document or track • How to ensure that exploratory testing captures all test cases systematically? • We face challenges when dealing with manual test data because a manual tester obviously does not use the system (exactly) the same way every time. • Exploratory testing need to be better defined and studied closer to see what it actually can contribute in for example integration testing • The whole topic of exploration [exploratory testing] is not embraced enough by academia; exploration is much more motivating and seems to have much better results than scripted test cases yet is seen as a "toy method".
Test scripting	• Need for automated unit-test code-generation based on source-code • Need for better tool support for automated test-script generation form test specification

This is the pre-print of a paper that has been published in the Journal of Software: Evolution and Process
DOI: onlinelibrary.wiley.com/doi/full/10.1002/smr.2251

Test activity	Example research topics suggested
	• Need for domain-specific languages (DSL) to simplify documenting manual test cases • Help is needed for writing mock-driven unit tests (which are tedious [to write]) • Lean test documentation, document-less test documentation (video, audio etc.), efficient test code generation, reusable test code
Test execution	• [Automated] set up of embedded devices and test environment; virtualization of devices and environment; simulation of environment • Immaturity of tools for test execution for real-time embedded systems (issues such as time-deterministic execution) • Lots of efforts for setting up test environments for the same SUT for different clients • It is hard to involve external resources and people [in test execution]. Only if they have experience from the same field, it is useful. • Execution related to massive variation of test environment is a huge problem for many industries. There is an "explosion" of configurations here. In that aspect, there is much to do for researchers. • Test environment setup has been always a big issue for test engineers. Some standards and operation procedures are needed to improve the process
Test evaluation	• Automating test oracles for embedded software systems • Automation tools don't always provide the expected output. It is difficult to evaluate upon wrong [unreliable] outputs • Test verdicts are hard to be determined when results are analogue, varied or non-functional requiring pure analytic • Need for test oracle for complex systems such as GPS software • Need for approaches based on machine learning in test evaluation • How to process massive amounts of log data and automatically detect potential unknown functional and non-functional problems in regression testing
Test-result reporting	• Complexity of maintaining and documenting test results • Need for better metrics, and better visualisation in support of test-result reporting • Test Reports should have more Key Performance Indicator (KPI)'s beyond test results • Visual display of test results
Test management	• Need for better coordinating of testing with Release Management ROI calculation • Need for better test-related management reports • Which tests to select for each stage of development project to balance test effort and finding defects early? • Objectively assessing the effectiveness and efficiency of test management • Test size and effort estimation • Figuring out what test approach is relevant • Help in prioritising tests • Need for test process maturity assessment and improvement • Better techniques/metrics for deciding what should be tested • Determining the value of testing vs. cost of testing
Test automation	• Maintenance of test automation • Need for test automation across the entire test process (not just execution) • How to make test automation more reliable? • How to compare different test frameworks? • How can one determine automation scope? • Tools are not easily customizable • What is the right amount of automation?
Other testing challenges	• Regression testing of complex legacy software • Design testability • Need for more effective tool-support for traceability analysis, especially test artifacts, in embedded software • Fidelity of testing when compared to actual, real world use (e.g., it passed all the tests, but *still* failed in the field) • How to make legacy code more testable • SW visualization in support of testing (e.g., of data and control flow), Heat mapping/visualization of test cases and code coverage for xUnit based test frameworks such as Google Test

4.4 RQ 3: Industrial challenges versus the recent focus of the academic community

The aim of RQ 3 was to “contrast” industrial challenges versus the research focus of the academic community in recent years.

This is the pre-print of a paper that has been published in the Journal of Software: Evolution and Process
DOI: onlinelibrary.wiley.com/doi/full/10.1002/smr.2251

To be able to address RQ 3 in a quantitative manner, we used the data for the level of challenges of different test activities in industry (RQ 1.1) as one source. To contrast that data, we needed a dataset to quantify the level of activity by academia on different test activities, i.e., how much challenge industry has on test automation versus how much academia has focused on test automation. Since testing is a large research field and clearly several thousand papers that have been published in this area [76, 77], we did not see any data source providing such metrics for us. As a proxy, we had access to data from two systematic mapping studies in two sub-areas of software testing: web application testing [41] and testing embedded software [78], as the first author was involved in those studies. These two systematic mapping studies had done classifications of all papers published on those sub-areas of software testing w.r.t. different test activities, thus we could use them. Although these two datasets were not perfect (ideal) since the level of activity by academia on different test activities in these two particular sub-areas of software testing may be different from the entire field of software testing. However, we still think these two proxy datasets provide an interesting snapshot.

We considered the number of papers studied in each of the mapping studies [41, 78], for each of the nine test activity types (see Table 2) as a proxy to quantitatively measure the level of recent focus of the academic community.

We compared the normalized level of challenges for each test activity type (as the quantitative proxy of level of industrial challenges) to the number of papers metrics taken from these two systematic mapping studies [41, 78], which is illustrated in Figure 9 using two scatter plots. The X-axes in both cases represent the normalized level of challenges for each test activity type (max=1.0, as calculated in Section 4.2.1) and the Y-axes are the number of papers concerning the test activity types in the two domains (web application testing [41] and testing embedded software [78]).

Note that there are slight differences between the classifications used in the two previous studies and the article at hand: the "Test scripting" and "Test-result reporting" categories have not been used in the previous studies. Furthermore, the classifications used in the systematic mapping study on testing web applications [41] did not have the "Test management" category. Therefore, we omit such cases in Figure 9.

By looking at Figure 9, we observe a noticeable difference between the two communities (industry versus academia) in both cases. To provide a big picture, we divided the charts into the two sectors excessive (academic) research, and shortage of research. The former denotes testing areas in which there is more academic research than the level of existing industrial challenges, and the latter denotes where further academic research is needed. Interestingly, both testing topics (web testing and embedded software testing) reveal similarities as shown in Figure 9, e.g., there is slight shortage of research on test automation in both testing domains. Also, there is excessive research on topics such as test design (criteria-based) and test evaluation on both testing topics. In the area of testing embedded software, fewer research has been performed on test management issues, where practitioners see more challenges.

We should mention at this point that "some" degree of mismatch is expected between the two communities (industry versus academia), given the different goals and timelines of industry and academia [79]. Academia works on long-term research problems, e.g., finding better approaches for test-case design, while industry faces challenges to increase efficiency and effectiveness of testing via approaches such as test automation. We believe that a main reason behind the "mismatch" of industrial challenges versus the recent focus of the academic community is low industry-academia collaboration in SE in general and in software testing in particular. This opinion is based on the authors' years of fruitful experience in conducting such collaborations, e.g., [19, 66, 80, 81], and also experience-based opinions of many other researchers in the field [79].

As Wohlin mentioned in [82]: "*Collaboration between industry and academia supports improvement and innovation in industry and helps to ensure industrial relevance in academic research*". The lack of such collaborations (the disconnect between research and practice) would hurt both sides: (1) it hurts researchers since their research topics tend to have low practical relevance [83, 84], their research output does not get the cane to have industrial impact, and most papers are only read and cited by a few other researchers only [85]; and (2) it also hears the software industry as they do not get the opportunity to use novel SE techniques developed in academia for improvement and innovation in industry. Of course, further discussions, implications and root causes of the gap are topics, which have been discussed widely in SE and other fields, are controversial and beyond the scope of this article.

This is the pre-print of a paper that has been published in the Journal of Software: Evolution and Process
DOI: onlinelibrary.wiley.com/doi/full/10.1002/smr.2251

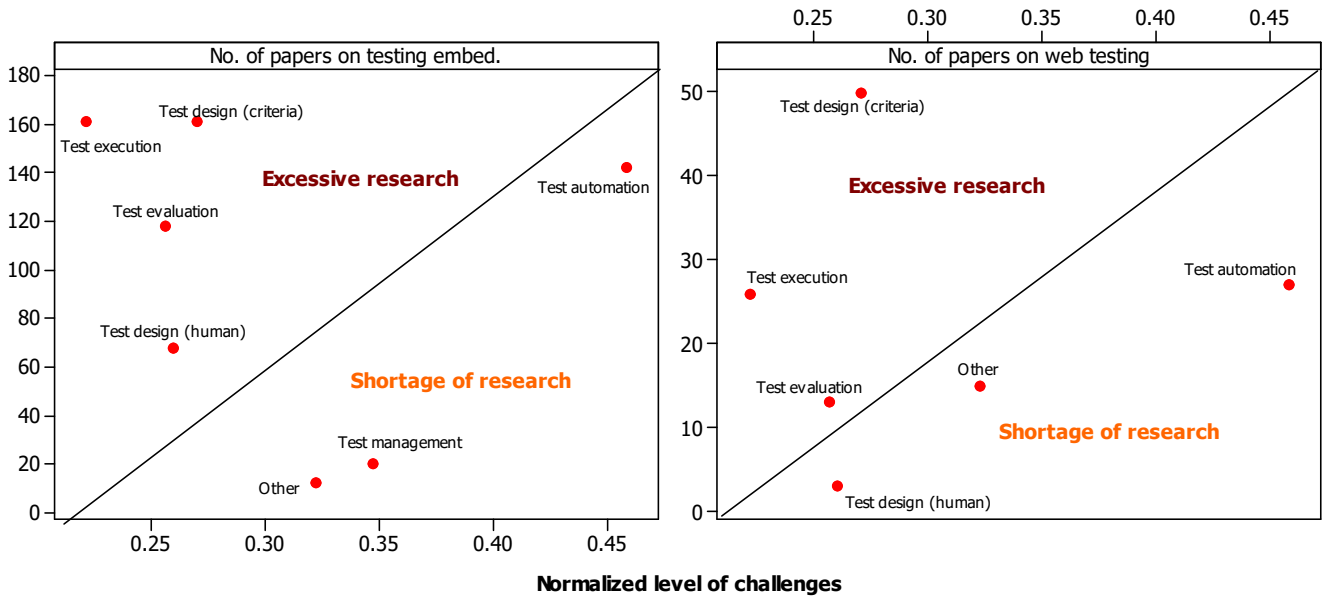


Figure 9 - Normalized level of challenges in each test activity type (max=1) versus number of papers in two example testing domains (web testing and embedded software testing) reviewed in two survey studies

The trends shown in Figure 9 also seem to confirm our findings from [5]: while "*researchers in academia feel excited by theoretically-challenging issues*", e.g., search-based test-case design, practicing test engineers would like to find ways to improve effectiveness and efficiency of testing (e.g., by better support for managing test automation) and are generally less interested in sophisticated methods or techniques, which they consider to be complicated and labor-intensive.

Furthermore, in our IEEE Software paper [5], we found that, when practitioners talk about test automation, they regularly refer to automating the test execution phase. However, academic work on automated approaches is mostly concerned with test-case generation and test oracles. Regarding the 10 most common phrases in the industrial testing conferences, practitioners discuss testing in its relationship to software quality quite frequently. Also, management issues and testing in the context of agile development are hot topics. Mobile testing, Cloud testing and performance testing are also popular issues. Finally, testing in context of continuous integration and delivery is a popular topic in industry [68, 69] and is frequently covered in talks at industrial conferences [5].

On the other hand, it seems that researchers in academia usually feel excited by "theoretically challenging" issues such as combinatorial testing and search-based test-case design, while practitioners just seek ways to improve effectiveness and efficiency of testing (e.g., better test automation) without the need for "fancy" methods or techniques, which they consider too complicated and time-consuming to implement and deploy into practice [79]. Model-based testing is quite popular in academia and seems to have some relevance in specific industry sectors, e.g., in testing automotive software, medical devices, and safety-critical systems in general. Yet, according to our data, model-based testing not widely used in most industry sectors. Mutation testing is widely discussed in research but seems to attract low attention in industry. To sum up, the topics of interest in industry and academia are quite different. Thus, we can state that the two groups "live in two different worlds", since their areas of interest and focus are quite different and we have seen that this is one main reason for low interaction among the two communities [5]. Hence, the findings of the current study support [5] and, moreover, both studies complement each other.

4.5 Discussion and recommendations

We summarize our discussions in this section, provide a critical overview on the state of the practice and then present a few recommendations.

This is the pre-print of a paper that has been published in the Journal of Software: Evolution and Process
DOI: onlinelibrary.wiley.com/doi/full/10.1002/smr.2251

4.5.1 Challenging test-activity types

As a key outcome, our results show that practitioners consider test management, test automation as well as other test activities the most challenging test activity types. Other test activity types such as test-case design, test execution, evaluation and test-result reporting have been seen less challenging.

Regarding test management, some of the main challenges raised by practitioners are related to assessing the effectiveness and efficiency of testing. Practitioners need guidance for selection of suitable testing approaches for a given context (testing challenge at hand) as well as the estimation of their return on investment (ROI). Such a challenge is typically seen in a process and organizational context that is often abstracted away when measuring cost-effectiveness of testing approaches in most of research studies [86]. There is a general shortage of studies measuring the ROI of different testing approaches. The good news is that there exist a handful of such studies, e.g., an ROI analysis was performed for model-based testing in [87] and jointly performed by practitioners and researchers in the context of the European Space Agency. Cost/benefit analyses of test automation in two industrial contexts were reported in [88, 89]. We should note that academic experiments performed in clearly-defined contexts are of high value, but at the same time, there is also a need for meta-studies on the effectiveness and efficiency of testing approaches based on primary studies, which could provide decision-support for practice.

Furthermore, similar to evidence-based medicine [90], there is the issue of efficacy versus effectiveness. Efficacy is defined as the performance of an intervention under ideal and controlled circumstances, whereas effectiveness refers to its performance under 'real-world' conditions. As nicely summarized in a blog-post [91] in the grey literature of evidence-based medicine, "*Tightly controlled trials are poor predictors of real-world outcomes*". While most industrial papers in SE (including software testing) are often "case studies", a medical effectiveness study attempts to collect data from a very large set of cases from varying contexts. This distinction, and the large-scale real-world trial are largely absent in the SE literature. Key challenges regarding test automation are maintenance of test automation scripts, devising "proper" strategies for test automation across the entire test process, and also decision-support for choosing test frameworks. The latter two challenges are also related to test management. Therefore, key challenges especially arise at the interface of test automation and management. The raised test automation challenges are linked to the issue that the term "test automation" is often used and interpreted slightly different in industry and academia [5]: while practitioners mainly refer to automated test execution that is essential for modern agile and continuous software development [92], researchers mainly consider automated test-case design.

Other testing challenges are mainly related to testing legacy systems as well as testability aspects (as discussed in Section 4.3.1). Testing legacy systems is an aspect largely neglected both in research and in education where "greenfield" development is preferred to "brownfield" development [93, 94]. However, in practice, challenges often arise in the context of continuous co-evolution of software systems and their test suites, which is also related to test automation maintenance, and also when deploying and testing new software systems in the presence of existing (legacy) software systems.

4.5.2 State of the software-testing practice: a critical overview

In retrospective, after exploring and analyzing the industry's challenges in software testing, a few questions arise, e.g., (1) whether industry, in general, is doing the "right" things, i.e., the techniques, generally used in practice, are generally "good enough" for purpose or should be enhanced, and (2) whether industry is doing the things well enough (adequate training, skill, and resourcing). In our opinion, the answer is: "It depends", an answer which is applicable to many SE-related questions [95, 96], as we discuss below.

The above two questions are mostly related of maturity of testing practices in a given software company, and various studies / models have been published to assess such a maturity. A recent multivocal literature review (MLR) compiled a list of 57 models for software test maturity assessment and test process improvement [62]. An example maturity model is the Test Maturity Model integration (TMMi) [63], which has been inspired by and is based on the Capability Maturity Model Integration (CMMI) [97], which itself, is used for maturity assessment of general SE practices.

This is the pre-print of a paper that has been published in the Journal of Software: Evolution and Process
DOI: onlinelibrary.wiley.com/doi/full/10.1002/smr.2251

Several survey studies have been published by the TMMi foundation, e.g., [61], which provide a high-level picture on the general test maturity of the software industry. We can see that, as one would expect, there is a varying spectrum of test maturity among different companies, e.g., in the level 1 (ad-hoc) to level 5 (optimization) of TMMi [63]. It is expected that if a given test team or company has low maturity, it would experience many challenges, and the reverse being expected for companies with higher test maturity. We were explicitly aware of this phenomenon and design and analysis of our survey, and for this reason, we classified the reported challenges in Section 4.3.1 into three categories: (1) concrete/real software testing research challenges; (2) non-technical challenges (e.g., due to resource constraints) or challenges outside testing; and (3) unclear responses, too generic, or not challenges really. In our opinion, the challenges reported in category (1) above are really "valid" test-related challenge, since for example when a respondent mentioned her/his company cannot conduct systematic performance testing due resource constraints, it is just a concrete organizational resource issue and not a technical challenge for us to consider.

Furthermore, aside from testing, there have been numerous efforts to shed light on general status of success, maturity and challenges of the software industry, e.g., the infamous CHAOS reports [98], which has been published since 1994, have been critical on the success rate of software projects and thus questioning whether the software industry, in general, is doing the "right" things w.r.t. SE and software testing practices. But there has been much criticism to the CHAOS report, e.g., [99-101]. With the high importance and penetration of software systems in the modern society, although we hear in the news about software defects once in a while, it is generally true that most systems have high quality. This honest view is in line with what Robert Glass reported in an IEEE Software paper in 1994 [102], in which he criticized the notion of "software crisis" and especially that it almost had not existed, or was not as major as many researchers had argued at the time. In his 1994 writing [102], he expressed the opinion that we had entered the "*Golden age of computing*" since that time, meaning that software industry was not crisis and was quite successful, in general, in developing high-quality software. The authors of this paper share this opinion, also for the current time. It is our belief that although complexity and size of software systems have increased tremendously since the 1990's, industry and, sometimes academia, have developed more sophisticated SE and software testing approaches.

With the above discussions on the realities of the modern software industry, we can argue that the software industry, in general, is doing reasonably "right" things in terms of testing. However, as discussed in Section 4.4, many companies still have the challenge (and goals) of increasing efficiency and effectiveness of their test practices.

4.5.3 Implications and Recommendations

In summary, we want to highlight that our article and the results presented in this article are intended to target both researchers and practitioners:

1. Researchers are the main audience of this article as they may get insights to the level of challenges of different test activities (Section 4.2). Furthermore, they may also review the set of reported concrete research challenges in testing (Section 4.3), and possibly derive research activities from them;
2. Practitioners could benefit two-fold:
    a. Practitioners may review the level of challenges in different test activities (Section 4.2) and also the set of concrete research challenges in testing (Section 4.3) reported in this article, which have been gathered/synthesized from a large number of other practitioners. Furthermore, they may "benchmark" their own challenges, i.e., they may or may not have the same challenges or on the same levels (magnitudes). That could provide a self-reflection on the issue and possibly encourages them to learn more about latest advances in different test activities.
    b. Practitioners may review our discussions on comparing industrial challenges versus the recent focus of the academic community (Section 4.4) and be motivated to reach out and study more papers in the academic literature, and possibly start new collaborations with researchers.

As discussed above, researchers are the main audience of our article as they may possibly plan research activities based on the set of concrete research challenges in testing (Section 4.3). Related to that long-term goal, we use our results in this

study and also our recent experience in industry-academia collaborations in SE, e.g., [19, 20, 66, 79], to provide a few additional recommendations for researchers:

- Academic researchers could try to (better) understand the industry problems [20] as such; not as a "normal" research problem known by the academia.
- Researchers could take into account that research in industry is expensive (from industry's standpoint) and, thus, it should return (immediate) useful/valuable outcomes to the industry [79]. It should be noted that every company's aim is making profit. Hence, researchers should also try including research areas with the highest benefit into consideration. In this regard, we consider it important applying ideas from value-based software engineering research [103].
- Researchers should understand that the timespan for industry is much shorter [79]. To be useful for practitioners, achieving many small short-term goals is better than targeting long-lasting investigations, that would not yield direct results to the industry.
- Researchers could try describing their research in laymen's terms, making it available in more forms than conference articles and journal papers [79]. Knowledge should be made available through various media, e.g., by publishing slides, recorded videos, and blogs.
- Researchers in academia could make sure that tools used and created, are available to practitioners after the researchers have left, and that tools contain sufficient documentation to be sustained and maintained by a company [79].
- Researchers could pay more attention to the deployment and dissemination of the results found; and not only collect data, write a paper, and "then leave" [79].
- Researchers should be more open to the actual problem to be solved. It is easy for an academic to approach a company with only one technique. Then research is looking for one specific problem, which might not be even close-to important to practitioners. If there is an openness towards industry problems, many solutions and possibilities for research exist that would be a "win-win", where many skills and techniques would be used, other than the "pet" area of the researcher. This requires much more know-how and experience of the researcher, including willingness to pursue other areas and to be able to contact other researchers in collaboration networks [79].

Researchers should apply the principles of the Action-Research approach [104] in their work, especially when collaborating with industry, to ensure that the research problems are chosen from the actual needs of the industry, and that the research outputs will actually be used and provide benefit to their industrial collaborators. We should note that our article's goal is not to analyze the root causes of separation between industry and academia in SE in general and in software testing in particular. Many papers in the SE literature have focused on that issue and similar topics, e.g., the reader is referred to recent papers such as [105-107] and the pool of 33 studies reviewed in a 2016 systematic literature review (SLR) [79] on that topic.

4.6 Limitations and threats to validity

In this section, we critically review our work regarding the potential threats to the validity of our study in terms of internal, construct, conclusion, and external validity [53], and the measures that we applied to mitigate or minimize them.

Internal validity: Internal validity is a property of scientific studies which reflects the extent to which a causal conclusion based on a study and the extracted data is warranted [53]. A potential threat to internal validity of this survey could be whether subjects did not have a common understanding of the survey questions. As discussed in Section 3.2.1, we took three steps to minimize this potential threat: (1) we relied on the terminology and concepts used in popular textbooks on software testing, e.g., [39], and also the guidelines of industry certification bodies on testing, e.g., ISTQB [40]; (2) we piloted the questionnaire with five test engineers in the industry; and (3) we provided a brief explanation of the concepts for each survey question.

Another issue, which could impact the survey's "reliability", and its internal validity [36] was a design issue (trade-off) in terms of the survey's length versus having multiple questions for the same construct. As we discussed in Section 3.2.1, since this survey addressed practitioners, we kept the number of questions to a minimum to ensure getting as many responses as possible [35]. Thus, we designed the survey instrument to be short, which is an advantage in obtaining compliance and gathering more data points, but we understood the drawback that it lacked multiple questions for the same construct. If we had multiple questions for the same construct, we could apply the Cronbach's alpha [36]. We understood that the inability

to apply this test metric limits the internal validity, but in the context of this trade-off, we decided to keep the number of questions to a minimum to ensure getting as many responses from practitioners as possible.

Another issue which could impact our study' internal validity was the classification of the raw reported challenges (reported in Section 4.3.1). As discussed, we used the qualitative coding [52], and more precisely "open" and "axial coding" [52], which is the established widely-used approach to process qualitative data. To ensure highest rigour in the analysis, we conducted the qualitative coding as systematic as possible (see the examples in Section 4.3.1), and to prevent bias of one researcher, all involved authors peer reviewed the qualitative coding process and artifacts.

Construct validity: Construct validities are concerned with the extent to which the objects of study truly represents theory behind the study [53]. In another words, construct validity is the extent to which a survey measures what it is intended to measure. Construct validity in this study relates to whether we really gathered practitioners' opinions about what industry wants from academia in software testing. Since all our data sources include direct opinions of practitioners, this threat was mitigated. Another threat to construct validity is the simplicity of the survey. This could partially have caused confusion especially in the ratings given. This is a direct consequence of using surveys as data collection tool. We have addressed this threat by discussing the instrument and results with selected practitioners to better understand the particularities of individual interpretations. One of the easiest ways to assess construct validity is to give the measure to two groups, one of which is known to have higher knowledge on nutrition than the other group [108]. However, since our survey was anonymous, this was not possible for us to do.

Conclusion validity: Conclusion validity deals with whether correct conclusions were drawn through rigorous and repeatable treatment [53]. We analyzed and discussed the trends in an aggregated form, and we also discussed open topics and challenges raised by practitioners. For RQ 1, we reduced the bias by seeking support from the 5-point Likert scale data in the survey. Thus, all the conclusions we have drawn in this study are strictly traceable to data. For RQ2 and RQ3, multiple authors, all of them with industrial and academic experience, were involved in the coding and analysis to mitigate the risk of invalid conclusions.

External validity: External validity is concerned with the extent to which the results of the study can be generalized [53]. As for external validity, although our sample size is limited, it provides a limited snapshot of practitioners' opinions. Thus, although some general trends are indicated, we cannot generalize the results of this study to all practitioners. Another possible threat to external validity in this study is the selection bias, i.e., randomness and representativeness of the data points. Since we collected data from three sources and from different contexts and countries, we believe that the selection bias is low.

5. Conclusions and future work

Software testing is the dominating method for quality assurance of industrial software. "*Despite its importance and the vast amount of resources invested, few research results are adopted by industry*" [65]. "*At the same time, the software industry is in dire need of better support for testing its software within the limited time available*" [65].

Many researchers, e.g., [79, 109-111], including the authors of this article, believe that characterizing industry's challenges in SE applied research enable more applied research in SE. The article characterized what industry would like to see academia putting more effort on in software testing research. For that purpose, an opinion survey has been conducted, which resulted in 72 practitioner opinions from three different data sources. The study draws a first big picture of industry expectations on academia and its software testing research efforts.

Findings of this study confirm findings of a recently published study [5], highlighting that industry and academic focus areas in software testing are disjoint. While researchers tend to be more interested in theoretically challenging issues, e.g., search-based test-case design, test engineers in practice look for options to improve effectiveness and efficiency of testing. Our study resulted in 104 research topic proposals of which, notably, test management and test automation were among the highest ranked topics. Our collection of topic proposals thus contributes to shaping future research directions in (practically relevant) software testing research.

This is the pre-print of a paper that has been published in the Journal of Software: Evolution and Process
DOI: onlinelibrary.wiley.com/doi/full/10.1002/smr.2251

Our future work directions include the following: (1) to conduct joint industry-academia collaboration in testing in the areas identified in this study; (2) to perform further empirical analyses, for instance by correlating challenges with technological factors; (3) to continue collecting practitioners' perceptions about practically relevant research directions, and (4) to investigate why available research results are not accessible or not applicable for practitioners.

Acknowledgements

The authors would like to sincerely thank the anonymous reviewers for their constructive feedbacks during the revision process of this article, which have led to several improvements of the article. Michael Felderer was partly funded by the Knowledge Foundation (KKS) of Sweden through the project 20130085: Testing of Critical System Characteristics (TOCSYC). Sigrid Eldh was partly funded by Eureka ITEA 3 TESTOMAT Project #16032 (Vinnova) and by Ericsson. Furthermore, the authors owe thanks to the practitioners who participated in this survey.

References

[1] M. J. Harrold, "Testing: a roadmap," in *Proceedings of the Conference on The Future of Software Engineering*, 2000, pp. 61-72, 336532.

[2] A. Bertolino and E. Marchetti, "A brief essay on software testing," *Software Engineering-Development process 1,* pp. 393-411, 2005.

[3] P. Bourque and R. E. Fairley, "Guide to the Software Engineering Body of Knowledge (SWEBOK), version 3.0," *IEEE Press,* 2014.

[4] V. Garousi, M. Felderer, M. Kuhrmann, and K. Herkiloğlu, "What industry wants from academia in software testing? Hearing practitioners' opinions," in *International Conference on Evaluation and Assessment in Software Engineering*, Karlskrona, Sweden, 2017, pp. 65-69.

[5] V. Garousi and M. Felderer, "Worlds apart: industrial and academic focus areas in software testing," *IEEE Software,* vol. 34, no. 5, pp. 38-45, 2017.

[6] ICST Conference, "Archive of Live streams," *http://www.es.mdh.se/icst2018/live/,* Last accessed: May 2019.

[7] P. Mohagheghi, M. A. Fernandez, J. A. Martell, M. Fritzsche, and W. Gilani, "MDE Adoption in Industry: Challenges and Success Criteria," in *International Conference on Model Driven Engineering Languages and Systems*, 2009, pp. 54-59.

[8] D. Damian, F. Lanubile, and H. L. Oppenheimer, "Addressing the challenges of software industry globalization: the workshop on Global Software Development," in *Proceedings of International Conference on Software Engineering*, 2003, pp. 793-794.

[9] R. Feldt, R. Torkar, E. Ahmad, and B. Raza, "Challenges with software verification and validation activities in the space industry," in *International Conference on Software Testing, Verification and Validation*, 2010, pp. 225-234: IEEE.

[10] V. Garousi, A. Coşkunçay, A. B. Can, and O. Demirörs, "A survey of software engineering practices in Turkey," *Journal of Systems and Software,* vol. 108, pp. 148-177, 2015.

[11] V. Garousi and J. Zhi, "A survey of software testing practices in Canada," *Journal of Systems and Software,* vol. 86, no. 5, pp. 1354-1376, 2013.

[12] A. Page, K. Johnston, and B. Rollison, *How we test software at Microsoft*. Microsoft Press, 2008.

[13] J. A. Whittaker, J. Arbon, and J. Carollo, *How Google tests software*. Addison-Wesley, 2012.

[14] A. Rainer, "Using argumentation theory to analyse software practitioners' defeasible evidence, inference and belief," *Information and Software Technology,* vol. 87, pp. 62-80, 2017.

[15] A. Williams and A. Rainer, "Toward the use of blog articles as a source of evidence for software engineering research," in *Proceedings of the International Conference on Evaluation and Assessment in Software Engineering*, 2017, pp. 280-285, 3084268.

[16] V. Garousi, M. Felderer, and M. V. Mäntylä, "The need for multivocal literature reviews in software engineering: complementing systematic literature reviews with grey literature," in *International Conference on Evaluation and Assessment in Software Engineering*, Limmerick, Ireland, 2016, pp. 171-176.

[17] C. Kaner, "Fundamental challenges in software testing," in *Technical report, Butler University, http://kaner.com/pdfs/FundamentalChallenges.pdf*, 2003.

[18] 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.

[19] V. Garousi, M. M. Eskandar, and K. Herkiloğlu, "Industry-academia collaborations in software testing: experience and success stories from Canada and Turkey," *Software Quality Journal,* vol. 25, no. 4, pp. 1091–1143, 2017.

[20] V. Garousi and K. Herkiloğlu, "Selecting the right topics for industry-academia collaborations in software testing: an experience report," in *IEEE International Conference on Software Testing, Verification, and Validation*, 2016, pp. 213-222.

This is the pre-print of a paper that has been published in the Journal of Software: Evolution and Process
DOI: onlinelibrary.wiley.com/doi/full/10.1002/smr.2251

[21] V. Garousi, M. Felderer, J. M. Fernandes, D. Pfahl, and M. V. Mantyla, "Industry-academia collaborations in software engineering: An empirical analysis of challenges, patterns and anti-patterns in research projects," in *Proceedings of International Conference on Evaluation and Assessment in Software Engineering*, Karlskrona, Sweden, 2017, pp. 224-229.

[22] V. Garousi *et al.*, "Characterizing industry-academia collaborations in software engineering: evidence from 101 projects," *Empirical Software Engineering,* journal article April 23 2019.

[23] A. Bertolino, "Software testing research: Achievements, challenges, dreams," in *2007 Future of Software Engineering*, 2007, pp. 85-103: IEEE Computer Society.

[24] P. Helle, W. Schamai, and C. Strobel, "Testing of Autonomous Systems–Challenges and Current State-of-the-Art," in *International Symposium of the International Council on Systems Engineering (INCOSE)*, 2016, vol. 26, no. 1, pp. 571-584.

[25] A. Santos and I. Correia, "Mobile testing in software industry using agile: challenges and opportunities," in *IEEE International Conference on Software Testing, Verification and Validation*, 2015, pp. 1-2: IEEE.

[26] A. Deak, T. Stålhane, and G. Sindre, "Challenges and strategies for motivating software testing personnel," *Information and software Technology,* vol. 73, pp. 1-15, 2016.

[27] J. Peleska, "Industrial-strength model-based testing-state of the art and current challenges," *arXiv preprint arXiv:1303.1006,* 2013.

[28] A. Gonzalez, E. Piel, H.-G. Gross, and M. Glandrup, "Testing challenges of maritime safety and security systems-of-systems," in *Testing: Academic & Industrial Conference-Practice and Research Techniques*, 2008, pp. 35-39: IEEE.

[29] M. Felderer and R. Ramler, "Experiences and challenges of introducing risk-based testing in an industrial project," in *International Conference on Software Quality*, 2013, pp. 10-29: Springer.

[30] M. Felderer and R. Ramler, "Integrating risk-based testing in industrial test processes," *Software Quality Journal,* vol. 22, no. 3, pp. 543-575, 2014.

[31] G. Canfora and M. Di Penta, "Testing services and service-centric systems: Challenges and opportunities," *IT Professional,* vol. 8, no. 2, pp. 10-17, 2006.

[32] B. Shneiderman, *The New ABCs of Research: Achieving Breakthrough Collaborations*. Oxford University Press, 2016.

[33] V. Garousi and M. V. Mäntylä, "When and what to automate in software testing? A multivocal literature review," *Information and Software Technology,* vol. 76, pp. 92-117, 2016.

[34] Z. Sahaf, V. Garousi, D. Pfahl, R. Irving, and Y. Amannejad, "When to automate software testing? decision support based on system dynamics – an industrial case study," in *Proc. of International Conference on Software and Systems Process*, 2014, pp. 149-158.

[35] J. S. Molleri, K. Petersen, and E. Mendes, "Survey Guidelines in Software Engineering: An Annotated Review," presented at the Proceedings of International Symposium on Empirical Software Engineering and Measurement, 2016.

[36] A. N. Ghazi, K. Petersen, S. S. V. R. Reddy, and H. Nekkanti, "Survey research in software engineering: problems and strategies," *arXiv preprint arXiv:1704.01090,* 2017.

[37] R. M. d. Mello, P. C. d. Silva, and G. H. Travassos, "Sampling improvement in software engineering surveys," presented at the Proceedings of ACM/IEEE International Symposium on Empirical Software Engineering and Measurement, 2014.

[38] R. M. d. Mello, P. C. d. Silva, P. Runeson, and G. H. Travassos, "Towards a framework to support large scale sampling in software engineering surveys," in *Proceedings of ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*, 2014, pp. 1-4.

[39] P. Ammann and J. Offutt, *Introduction to Software Testing*, 2nd ed. Cambridge University Press, 2016.

[40] D. Graham, E. V. Veenendaal, and I. Evans, *Foundations of Software Testing: ISTQB Certification*. Cengage Learning, 2008.

[41] V. Garousi, A. Mesbah, A. Betin-Can, and S. Mirshokraie, "A systematic mapping study of web application testing," *Elsevier Journal of Information and Software Technology,* vol. 55, no. 8, pp. 1374–1396, 2013.

[42] F. Strack, "“Order Effects” in Survey Research: Activation and Information Functions of Preceding Questions," in *Context Effects in Social and Psychological Research*, N. Schwarz and S. Sudman, Eds.: Springer, 1992, pp. 23-34.

[43] Anonymous author(s), "Learning patterns/Framing survey questions," *https://meta.wikimedia.org/wiki/Learning_patterns/Framing_survey_questions*, Last accessed: May 2019.

[44] H. Schuman, S. Presser, and J. Ludwig, "Context effects on survey responses to questions about abortion," *Public Opinion Quarterly,* vol. 45, no. 2, pp. 216-223, 1981.

[45] D. I. K. Sjoeberg *et al.*, "A survey of controlled experiments in software engineering," *Software Engineering, IEEE Transactions on,* vol. 31, no. 9, pp. 733-753, 2005.

[46] T. Punter, M. Ciolkowski, B. Freimut, and I. John, "Conducting on-line surveys in software engineering," in *Proceedings of International Symposium on Empirical Software Engineering*, 2003, pp. 80-88.

This is the pre-print of a paper that has been published in the Journal of Software: Evolution and Process
DOI: onlinelibrary.wiley.com/doi/full/10.1002/smr.2251

[47] V. Garousi and T. Varma, "A replicated survey of software testing practices in the Canadian province of Alberta: what has changed from 2004 to 2009?," *Journal of Systems and Software,* vol. 83, no. 11, pp. 2251-2262, 2010.

[48] V. Garousi and J. Zhi, "A Survey of Software Testing Practices in Canada," *Journal of Systems and Software,* vol. 86, no. 5, pp. 1354–1376, May 2013.

[49] L. Wallace, M. Keil, and A. Rai, "Understanding software project risk: a cluster analysis," *Inf. Manage.,* vol. 42, no. 1, pp. 115-125, 2004.

[50] T. R. Lunsford and B. R. Lunsford, "The Research Sample, Part I: Sampling," *J. Prosthetics and Orthotics,* vol. 7, pp. 105-112, 1995.

[51] R. Ferber, "Editorial: Research by Convenience," *J. Consumer Research,* vol. 4, pp. 57-58, 1977.

[52] M. B. Miles, A. M. Huberman, and J. Saldana, *Qualitative Data Analysis: A Methods Sourcebook*, Third Edition ed. SAGE Publications Inc., 2014.

[53] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén, *Experimentation in Software Engineering: An Introduction*. Kluwer Academic Publishers, 2000.

[54] V. Garousi, M. Felderer, M. Kuhrmann, K. Herkiloğlu, and S. Eldh, "Online survey questions: What industry wants from academia in software testing (phase 2)," *https://doi.org/10.5281/zenodo.893279*, Last accessed: May 2019.

[55] V. Garousi, M. Felderer, M. Kuhrmann, and K. Herkiloğlu, "Online survey questions: What industry wants from academia in software testing (phase 1)," *https://doi.org/10.5281/zenodo.837330*, Last accessed: May 2019.

[56] V. Garousi, M. Felderer, M. Kuhrmann, and K. Herkiloğlu, "Survey raw data: What industry wants from academia in software testing (phase 1)," *https://doi.org/10.5281/zenodo.322628*, Last accessed: May 2019.

[57] S. P. Ng, T. Murnane, K. Reed, D. Grant, and T. Y. Chen, "A preliminary survey on software testing practices in Australia," in *Proceedings of Australian Software Engineering Conference*, 2004, pp. 116-125.

[58] M. Felderer and F. Auer, "Software Quality Assurance During Implementation: Results of a Survey in Software Houses from Germany, Austria and Switzerland," in *Proceedings of International Conference on Software Quality. Complexity and Challenges of Software Engineering in Emerging Technologies*, 2017, pp. 87-102.

[59] G. v. Belle, *Statistical Rules of Thumb*. John Wiley & Sons, 2002.

[60] H. Weisberg, *Central tendency and variability*. Sage, 1992.

[61] Experimentus Corp., "State of Testing Resport 2017: Experimentus anual TMMi Industry Survey Results," *https://www.tmmi.org/tm6/wp-content/uploads/2016/08/Experimentus-State-of-Testing-2017-Infographic.pdf*, Last accessed: May 2019.

[62] V. Garousi, M. Felderer, and T. Hacaloğlu, "Software test maturity assessment and test process improvement: A multivocal literature review," *Information and Software Technology,* vol. 85, pp. 16–42, 2017.

[63] TMMI Foundation, "TMMI specification (reference model), release 1.0," *http://www.tmmi.org/pdf/TMMi.Framework.pdf*, Last accessed: Oct. 2015.

[64] V. Garousi, Y. Amannejad, and A. Betin-Can, "Software test-code engineering: a systematic mapping," *Journal of Information and Software Technology,* vol. 58, pp. 123–147, 2015.

[65] S. Eldh, "On Test Design," *Doctoral dissertation, Mälardalen University, Sweden,* 2011.

[66] V. Garousi, D. C. Shepherd, and K. Herkiloğlu, "Successful engagement of practitioners and software engineering researchers: Evidence from 26 international industry-academia collaborative projects," *IEEE Software, In press,* 2019.

[67] A. Arcuri, "An experience report on applying software testing academic results in industry: we need usable automated test generation," *Empirical Software Engineering,* vol. 23, no. 4, pp. 1959-1981, 2018.

[68] M. Kuhrmann *et al.*, "Hybrid software development approaches in practice: a European perspective," *IEEE Software, in press,* 2018.

[69] M. Kuhrmann *et al.*, "Hybrid software and system development in practice: waterfall, scrum, and beyond," in *Proceedings of the 2017 International Conference on Software and System Process*, 2017, pp. 30-39.

[70] M. v. Genuchten, "Why is Software Late? An Empirical Study of Reasons for Delay in Software Development," *IEEE Transactions on Software Engineering,* vol. 17, no. 6, pp. 582-590, 1991.

[71] S. McConnell, "What Does 10x Mean? Measuring Variations in Programmer Productivity," in *Making Software: What Really Works, and Why We Believe It*, A. Oram and G. Wilson, Eds.: O'Reilly Media, 2010.

[72] A. Bednarz, "How IBM started grading its developers' productivity," *http://www.networkworld.com/article/2182958/software/how-ibm-started-grading-its-developers--productivity.html*, 2011, Last accessed: May 2017.

[73] V. Garousi, "Incorporating Real-World Industrial Testing Projects in Software Testing Courses: Opportunities, Challenges, and Lessons Learned," in *Proceedings of the IEEE Conference on Software Engineering Education and Training (CSEE&T)*, 2011, pp. 396-400.

This is the pre-print of a paper that has been published in the Journal of Software: Evolution and Process
DOI: onlinelibrary.wiley.com/doi/full/10.1002/smr.2251

[74] V. Garousi and A. Mathur, "Current State of the Software Testing Education in North American Academia and Some Recommendations for the New Educators " in *Proceedings of IEEE Conference on Software Engineering Education and Training*, 2010, pp. 89-96.

[75] V. Garousi, "An Open Modern Software Testing Laboratory Courseware: An Experience Report " in *Proceedings of the IEEE Conference on Software Engineering Education and Training*, 2010, pp. 177-184.

[76] G. Mathew, A. Agrawal, and T. Menzies, "Finding Trends in Software Research," *IEEE Transactions on Software Engineering, in press,* 2019.

[77] V. Garousi and M. V. Mäntylä, "Citations, research topics and active countries in software engineering: A bibliometrics study " *Elsevier Computer Science Review,* vol. 19, pp. 56-77, 2016.

[78] V. Garousi, M. Felderer, Ç. M. Karapıçak, and U. Yılmaz, "What we know about testing embedded software," *IEEE Software, In press,* 2017.

[79] V. Garousi, K. Petersen, and B. Özkan, "Challenges and best practices in industry-academia collaborations in software engineering: a systematic literature review," *Information and Software Technology,* vol. 79, pp. 106–127, 2016.

[80] V. Pekar, M. Felderer, R. Breu, F. Nickl, C. Roßik, and F. Schwarcz, "Integrating a lightweight risk assessment approach into an industrial development process," in *International Conference on Software Quality*, 2016, pp. 186-198: Springer.

[81] S. Mohacsi, M. Felderer, and A. Beer, "Estimating the cost and benefit of model-based testing: a decision support procedure for the application of model-based testing in industry," in *2015 Euromicro Conference on Software Engineering and Advanced Applications*, 2015, pp. 382-389: IEEE.

[82] C. Wohlin *et al.*, "The Success Factors Powering Industry-Academia Collaboration," *IEEE Software,* vol. 29, no. 2, pp. 67-73, 2012.

[83] D. Lo, N. Nagappan, and T. Zimmermann, "How practitioners perceive the relevance of software engineering research," presented at the Proceedings of Joint Meeting on Foundations of Software Engineering, 2015.

[84] I. Sommerville, "The (ir)relevance of academic software engineering research," *http://iansommerville.com/systems-software-and-technology/the-irrelevance-of-academic-software-engineering-research/*, Last accessed: Feb. 1, 2019.

[85] A. K. Biswas and J. Kirchherr, "Prof, no one is reading you," *Singapore Press Holdings, https://www.straitstimes.com/opinion/prof-no-one-is-reading-you*, 2015, Last accessed: Feb. 1, 2019.

[86] L. Briand, D. Bianculli, S. Nejati, F. Pastore, and M. Sabetzadeh, "The Case for Context-Driven Software Engineering Research," *IEEE Software,* vol. 34, no. 5, pp. 72-75, 2017.

[87] S. Mohacsi, M. Felderer, and A. Beer, "Estimating the Cost and Benefit of Model-Based Testing: A Decision Support Procedure for the Application of Model-Based Testing in Industry," in *2015 41st Euromicro Conference on Software Engineering and Advanced Applications*, 2015, pp. 382-389.

[88] S. A. Jolly, V. Garousi, and M. M. Eskandar, "Automated Unit Testing of a SCADA Control Software: An Industrial Case Study based on Action Research," in *IEEE International Conference on Software Testing, Verification and Validation (ICST)*, 2012, pp. 400-409.

[89] V. Garousi and E. Yıldırım, "Introducing automated GUI testing and observing its benefits: an industrial case study in the context of law-practice management software," in *Proceedings of IEEE Workshop on NEXt level of Test Automation (NEXTA)*, 2018, pp. 138-145.

[90] E.-B. M. W. Group, "Evidence-based medicine. A new approach to teaching the practice of medicine," *Jama,* vol. 268, no. 17, p. 2420, 1992.

[91] W. Aryitey, "Efficacy vs Effectiveness," *https://thetranslationalscientist.com/disease-area/efficacy-vs-effectiveness*, Last accessed: April 2019.

[92] V. Garousi and F. Elberzhager, "Test automation: not just for test execution," *IEEE Software,* vol. 34, no. 2, pp. 90-96, 2017.

[93] B. Boehm, "Applying the Incremental Commitment Model to Brownfield Systems Development," in *Annual Conference on Systems Engineering Research (CSER)* 2009.

[94] R. Hopkins and K. Jenkins, *Eating the IT elephant: moving from greenfield development to brownfield*. Addison-Wesley Professional, 2008.

[95] Various contributors, "How to answer "it depends" questions," *https://softwareengineering.meta.stackexchange.com/questions/766/how-to-answer-it-depends-questions*, Last accessed: May 2019.

[96] T. Pfeiffer, "Are comments a code smell? yes / no. It depends.," *https://pragtob.wordpress.com/2017/11/14/are-comments-a-code-smell-yes-no-it-depends/*, Last accessed: May 2019.

[97] I. Burnstein, A. Homyen, R. Grom, and C. R. Carlson, "A Model to Assess Testing Process Maturity," *Crosstalk: The Journal of Defense Software Engineering,* vol. 11, 1998.

[98] The Standish Group, "CHAOS Report 2016," *https://www.standishgroup.com/outline*, Last accessed: May 2019.

[99] J. L. Eveleens and C. Verhoef, "The rise and fall of the CHAOS report figures," *IEEE software,* no. 1, pp. 30-36, 2009.

[100] D. Rubinstein, "Standish group report: There's less development chaos today," *Software Development Times,* vol. 1, 2007.

[101] R. L. Glass, "The Standish report: does it really describe a software crisis?," *Communications of the ACM,* vol. 49, no. 8, pp. 15-16, 2006.

This is the pre-print of a paper that has been published in the Journal of Software: Evolution and Process
DOI: onlinelibrary.wiley.com/doi/full/10.1002/smr.2251

[102] R. L. Glass, "The software-research crisis," *IEEE Software,* no. 6, pp. 42-47, 1994.

[103] S. Biffl, A. Aurum, B. Boehm, H. Erdogmus, and P. Grünbacher, *Value-Based Software Engineering*. Springer, 2006.

[104] D. E. Avison, F. Lau, M. D. Myers, and P. A. Nielsen, "Action research," *Communications of the ACM,* vol. 42, no. 1, pp. 94-97, 1999.

[105] C. L. Goues, C. Jaspan, I. Ozkaya, M. Shaw, and K. T. Stolee, "Bridging the Gap: From Research to Practical Advice," *IEEE Software,* vol. 35, no. 5, pp. 50-57, 2018.

[106] J. C. Carver and R. Prikladnicki, "Industry–Academia Collaboration in Software Engineering," *IEEE Software,* vol. 35, no. 5, pp. 120-124, 2018.

[107] V. Basili, L. Briand, D. Bianculli, S. Nejati, F. Pastore, and M. Sabetzadeh, "Software engineering research and industry: a symbiotic relationship to foster impact," *IEEE Software,* vol. 35, no. 5, pp. 44-49, 2018.

[108] K. Parmenter and J. Wardle, "Evaluation and design of nutrition knowledge measures," *Journal of Nutrition Education,* vol. 32, no. 5, pp. 269-277, 2000.

[109] K. Petersen, C. Gencel, N. Asghari, D. Baca, and S. Betz, "Action research as a model for industry-academia collaboration in the software engineering context," in *Proceedings of the ACM International Workshop on Long-Term Industrial Collaboration on Software Engineering*, 2014, pp. 55-62.

[110] A. Sandberg, L. Pareto, and T. Arts, "Agile collaborative research: Action principles for industry-academia collaboration," *IEEE Software,* vol. 28, no. 4, pp. 74-83, 2011.

[111] S. J. Lamprecht and G. J. Van Rooyen, "Models for technology research collaboration between industry and academia in South Africa," in *Proceedings of the Software Engineering Colloquium*, 2012, pp. 11-17.