Software Testing Practices in Industry: The State of the Practice

Article *in* IEEE Software · May 2016

DOI: 10.1109/MS.2016.87

CITATIONS
5

READS
246

3 authors:

Mohamad Kassab
Pennsylvania State University
60 PUBLICATIONS **416** CITATIONS
SEE PROFILE

Joanna F. DeFranco
Pennsylvania State University
59 PUBLICATIONS **200** CITATIONS
SEE PROFILE

Phillip A. Laplante
Pennsylvania State University
273 PUBLICATIONS **2,105** CITATIONS
SEE PROFILE

Software Testing Practices in Industry: The State of the Practice

Mohamad Kassab
The Pennsylvania State University
Engineering Division
30 E Swedesford Rd, Malvern, PA 19355
610-648-3277
muk36@psu.edu

Joanna DeFranco
The Pennsylvania State University
Engineering Division
30 E Swedesford Rd, Malvern, PA 19355
610-648-3357
jfd104@psu.edu

Phillip Laplante
The Pennsylvania State University
Engineering Division
30 E Swedesford Rd, Malvern, PA 19355
610-648-3277
plaplante@psu.edu

Digital Object Identifier 10.1109/MS.2016.87 0740-7459/$26.00 2016 IEEE

Abstract— A Web-based survey of more than 167 software professionals to determine the use of various testing practices was conducted. This exploratory survey and its quantitative results reported herein offer opportunities for further interpretation and comparison to software testers, project managers and researchers. Data includes characteristics of projects, practices, organizations, and practitioners related to software testing.

Keywords— software testing; software quality; agile; industrial survey.

1. INTRODUCTION

The focus of recent research efforts on software testing has been largely on designing new techniques and validating their effectiveness in real development contexts [1]. Throughout the history of software development, however, testing techniques have struggled to keep up with the ever faster trends in software development paradigms [1]. In order to trigger any favorable change in this state of practice, a serious effort is required in predicting the trends, learning the stakeholder mindsets, and pinpointing the problem areas in software testing.

Software quality is likely to become an increasingly important factor in software marketing. As this situation evolves testing strategies become progressively more important. Unfortunately little contemporary data exists to document the actual practices used by software professionals for software testing and quality assurance (QA) activities. Therefore, a comprehensive survey of software professionals was conducted to attempt to discover these practices.

Surveys of software industry professionals are an effective way to determine the current trends in software engineering processes. Survey responses can also help others to understand the relationship between area such as software quality and testing [2]. A carefully constructed survey has the potential to: 1) remedy the deficiency of lack of data and 2) to identify the software testing best practices, which can then be disseminated. Based on these two objectives, a number of hypotheses were employed to design a survey study on the current software testing state of practice. The data from this survey is exhibited herein.

2. SURVEY DESIGN AND CONDUCT

A web-based survey instrument was created using the web-based QuestionPro survey tool (www.QuestionPro.com). The survey consisted of 40 questions. The survey questions were designed after a careful review to similar other conducted survey studies [1, 2, 3, 4, 5]. The effort for designing the survey was approximately at 120 person-hours and the design process lasted for about 1 month. To allow a valid comparison with the other conducted surveys, we included selected questions from these surveys into ours. A summary of our survey questions is available in https://goo.gl/kGBLhq

Respondents were asked to base their responses on only one software project that they were either currently involved with or had taken part in during the past five years. Using the conjectures in our hypotheses as means of constructing specific questions, the survey was arranged into six sections related to: project information, integrating software testing activities within the software development life cycle; software testing methodologies and techniques; testing metrics and defects management, organization's information and participant's professional information.

We drew our survey participants from multiple sources but primarily from a database of past graduate students in Software Engineering of the Penn State School of Graduate Professional Studies. The school caters primarily to working professionals. An email invitation (and subsequent reminder) was sent to these individuals. We also posted an invitation at related Linked-In professional testing and quality groups, to which one or more of the authors belonged.

We collected survey data through January and June 2015. Of the 199 who viewed the survey; there were 167 who started taking the survey. Of these survey takers; there were 67 who completed the survey all the way to the end. The completion rate was 40% and the average time taken to complete the survey was 17 minutes. We also included the results of the partially completed responses. All responses were treated anonymously and only aggregate data were used - not the individual responses.

Digital Object Identifier 10.1109/MS.2016.87 0740-7459/$26.00 2016 IEEE

3. Statistics Regarding the Survey Participants and their Organizations

In order to make well-informed statements about the practice of software testing, it was essential to attract as many experienced participants and industries as possible. Both objectives were achieved for this study. Eighteen different industries were represented. The reported professional experience represented in the survey was impressive with an average of 7.8 years of related IT/Software experience. The reported academic qualification was also impressive with 68% of survey participants stated that they have successfully completed a bachelor's or equivalent, and 32% even hold a master's degree or doctorate.

The survey responses captured a diverse mix of positions within the chosen projects. To view the complete survey results on participant and project characteristics in a graphical format, we suggest to the reader to view these charts at: https://goo.gl/xWHEhO

As far as the size of the participating companies is concerned, a representative sample can be determined. Almost 44% of the participants work in small companies (with 1 – 100 full time employees). But also very large companies (with more than 1000 full time employees) are well represented at 30%. The question enquiring about an independent QA department in the organization was affirmed by 65% of all participants; which is mapped to 79% in very large companies and only 44% in small companies. The sample population presents companies located in a different geographic regions (9 countries were represented in this survey).

4. Statistics Regarding the Project's Characteristics

Several questions were asked in an effort to classify and categorize the software projects respondents' referred to. In general, projects were distributed across a broad range of application domains with a mild bias towards applications in the Information Technology sector (15% of the projects). The projects were also distributed across different categories with bias towards database projects (22% of the projects).

The majority of the projects were classified as new development (at 65%), while 15% were legacy system evolution projects and 14% were classified as enhancement projects.

We also aimed at classifying the projects based on their safety-criticality. The respondents were asked to specify the maximum loss or damage if the software being developed for the project failed (that is, the delivered service no longer complies with the specifications). Only about 16% reported a highly critical system where serious failure could involve loss of one or more lives. The majority of respondents reported the loss would be limited to essential funds, discretionary funds or comfort.

As for the projects' duration from inception to delivery, 48% took one year or less to complete, 21% took between 12 months to 24 months to complete and 31% of the reported projects took more than 24 months to complete. The average number of full time staff (IT) involved in the project was 20. The survey also looked at project size in terms of lines of code. The projects experienced were predominantly of medium size in terms of lines of code. 36% of projects contained 50,000 lines of code or less.

5. How Testing Activities are Integrated within the Development Life Cycle?

Since the process of developing a software has evolved into an engineering discipline, often involving largely distributed teams, the techniques and frameworks used have evolved accordingly. The need for cost-effective software production has also reached the software testing. But how is software testing methodically implemented in organizations? To answer this, we started by asking the participants on the software development life-cycles (SDLCs) that best describe the ones used in the project. Agile development methodologies (e.g. SCRUM, Extreme Programming, Feature Driven Development) were selected 50% of the time making them the most popular (Figure 1).

Digital Object Identifier 10.1109/MS.2016.87 0740-7459/$26.00 2016 IEEE

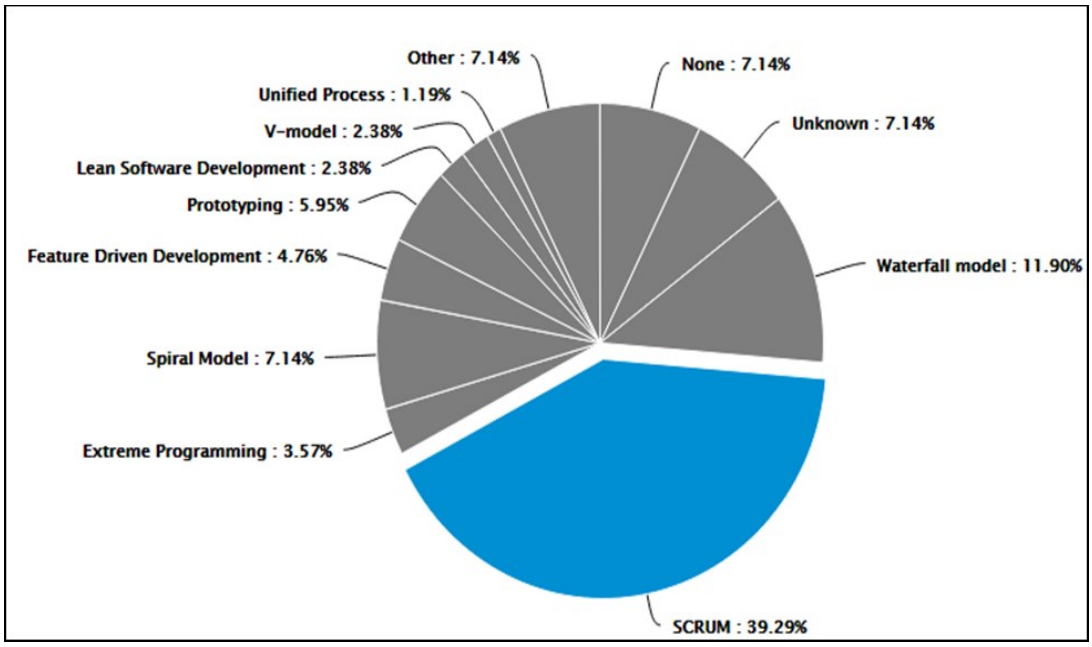

Figure 1: Which of the following development life-cycles best describes the one you are using / did use in the project?

The question on whether there is an independent QA team / department in the organization was answered negatively by 41% of agile participants, but by the users of the Waterfall model at only 13%. This response, however, must also be seen in relation to the company size, since agile processes are mainly popular in small-sized companies that usually do not have their own separate QA team / department.

The results from the question "In which phases are / were Quality Assurance measures applicable within this project?" have shown that performing QA measures is concentrated on the late phases of software development. This result is consistent with the one reported in [1]. Only 19% of the participants agreed that they use QA measures in the "Study & Concept" phase. While the shares of those that use quality assurance in the "Requirements Specifications" and "System Design" phases are 47% and 49% respectively. From the implementation phase onwards, quality assurance practices increase significantly (Figure 2). When we broke the responses against the SDLC, we observed that the waterfall process model outperformed agile ones in considering QA measures earlier in the development cycle; while agile outperformed waterfall in considering QA measures at the later stages of the development.

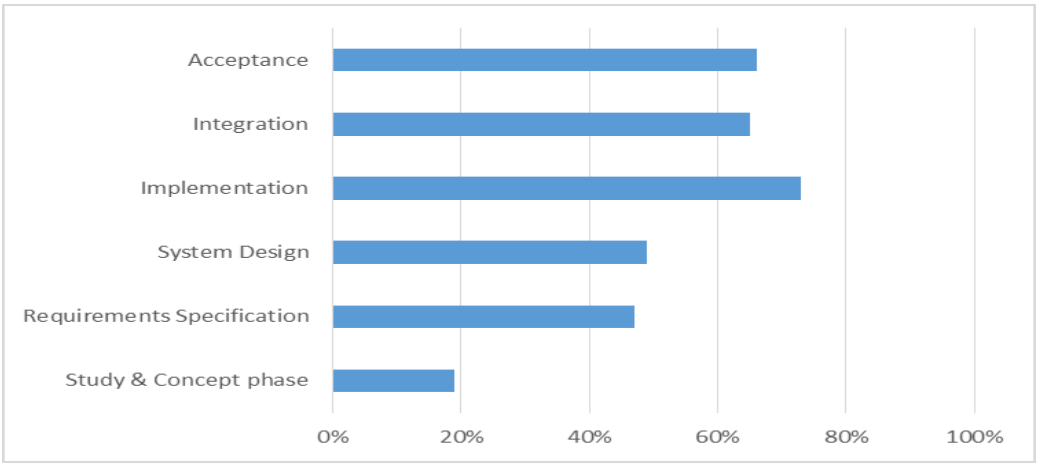

Figure 2: In which phases are/were Quality Assurance measures applicable within this project?

Digital Object Identifier 10.1109/MS.2016.87 0740-7459/$26.00 2016 IEEE

When asked on the "Testing as defined phase in the project development"; 71% of participants reported a level of agreement (strongly agree or agree) that this was the actual practice in the project. Eighty four percent also reported an agreement that they personally prefer such a practice to have testing as a defined phase on its own. When asked on whether "Testing and code development should not be distinct phases in a project"; 44% reported a level of agreement that this was the actual practice in the project (47% for agile comparing to 30% for Waterfall) and 50% reported an agreement that they personally prefer not to have testing and code as distinct phases. If we considered a dissatisfaction in a question as the difference between the preferred and the current practice, then these numbers indicate a good level of satisfaction on how the software testing activities are distinct from the rest of activities within a project.

On testing effort and budget estimation, around 17% of survey participants reported that they did not plan any explicit estimation for QA / testing activities (see Figure 3) -- comparing to 7.3% that was reported on a similar question in [1]. When the testing effort and budget were estimated, it was mostly done together with the other software development activities in a total package (37% of all respondents). While only 29% reported that the team is not doing a good job for estimating the size / effort of software testing activities; it was surprising to see that 43% then reported that they didn't have enough time to test the software before it was deployed.

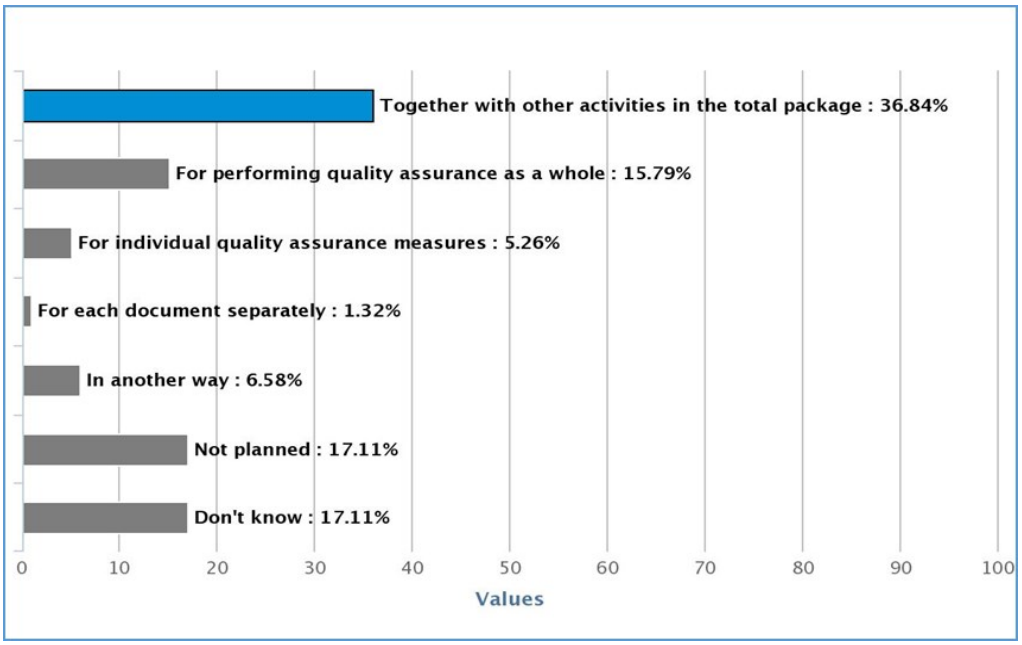

Figure 3: The quality assurance effort and budget in your projects are planned...

6. Testing Methods and Practices

Respondents, who reported that they conducted testing activities were additionally asked a series of questions regarding techniques and tools in use within their projects.

It is very common for organizations to have defined levels of testing [3]. Those levels include unit, integration, system, acceptance and regression testing. While the results in Figure 4-a clearly show that system-level testing is the most common level of testing (81% of participants reported applying system test for their projects), around 90% of participants reported that system testing was applied in order to test more than just one characteristic of the system with a clear focus on testing the functionality (Figure 4-b). We presented a list of characteristics that was derived from the taxonomy of system testing proposed in [13]; participants could select all that applied. Performance was the most tested quality attribute in the surveyed projects. The survey showed that in all of their testing activities, participants use principally Black-Box testing techniques (78% used this technique comparing to only 42% using structured-based (white-box) test design techniques).

Regression Testing is a level of software testing that seeks to uncover new software bugs, or regressions, in existing functional and non-functional areas of a system after changes such as enhancements, patches or configuration changes,

Digital Object Identifier 10.1109/MS.2016.87 0740-7459/$26.00 2016 IEEE

have been made to them (Wikipedia – Regression Testing [6]). Only 12 % reported that regression testing was outsourced for their projects; and this number is very close to the only 11% of participants who personally preferred to outsource regression testing. This indicates a level of satisfaction on the current outsourcing practices of the regression testing activities. The actual practice of outsourcing was reported at its highest level in the telecommunication and gaming industries (at 50%) comparing to 33.33 % for the IT industry.

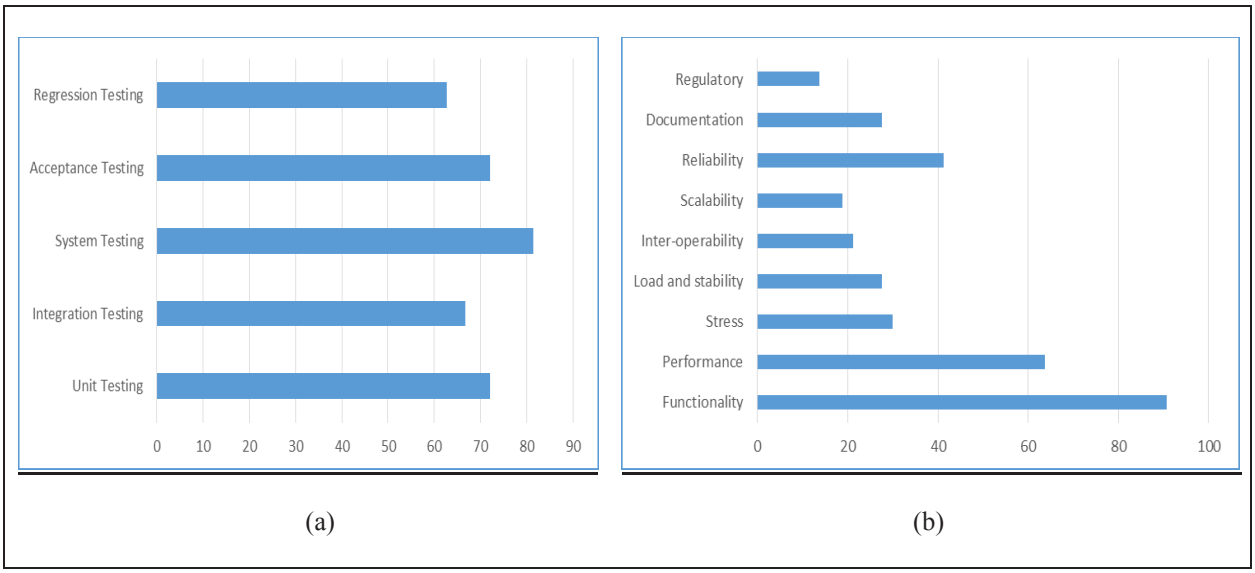

Figure 4: a-Which of the following levels of testing are/ were applied for this project? (Select all that apply) (n = 122); b- Which of the following types of system tests are / were executed for your project? (Select all that apply) (n = 122)

The results show that the systematic design of test cases and the definition of test data are widespread. Systematic test case design involves using a methodology for identifying test cases so that the amount of test cases needed is reduced without reduced test efficacy, for example, by using orthogonal tests or some form of data flow testing. Systematic test data definition involves identifying inputs to individual test cases via some process other than a purely random one. Even though 66% of the participants reported that they prefer to have the test cases written before writing the code; only 26% reported that this was the actual practice in their projects. It was surprising to see that for the non-safety-critical systems category of respondents, the current practice of writing test cases before writing code was more popular than for the safety-critical systems category. However, while the non-safety-critical respondents seem willing to even improve the situation more, the safety-critical respondents show no interest as a group in changing towards a more test-driven development. This is consistent with the results that were reported from another survey study in [1]. This is noteworthy considering the fact that empirical studies seem to ascribe test-driven developed code a high external code quality [1].

An interesting corollary to test case writing is the test case representation's degree of formality. About 38% of the respondents reported that test cases are expressed freely in verbal or text-based forms for their projects. Formal languages are used by 37% of survey respondents, showing that these approaches are becoming established in practice. With respect to tools used as a support for writing the test cases, respondent indicated that there is utilization of both proprietary and open source tools. However, while open source testing tools are in place mostly for unit testing (e.g. Junit), proprietary tools seem to be mostly used for higher level of testing (e.g. Jira, HP Quality Center). This is consistent with the results reported in [1].

The overall effectiveness of test cases is highly rated, since almost 47% reported that most or all of defects are found during test cases execution. In comparison, 37% rate test case effectiveness as medium (some defects are found) and only 11% indicated that effectiveness is low because the test cases find too few defects.

Sixty Nine percent of the participants affirmed that the expected test results were available in the test cases prior to test execution. This survey result is a positive surprise. On the other hand, approximately 66% of participants compare

Digital Object Identifier 10.1109/MS.2016.87 0740-7459/$26.00 2016 IEEE

the expected results with the actual results manually. Only 34% stated that the comparison between the expected and the actual results is normally automated and performed with the help of tools.
While data protection is a matter that organizations must take very seriously these days, it was therefore quite disappointing that only 22% of survey respondents indicated that they still test with original data from production (figure 5-a). Twenty Eight percent use an (anonymized) copy of the production data, and only 22% of the respondents indicated that they actually comprehensively document the test data. In addition, majority (62%) of the survey participants stated that they do not explicitly distinguish between test case generation and the generation of associated test data.

While the majority of respondents reported on using more than just one testing environment; around 76% reported using the development system as a testing environment and 39% reported on using the production system even as a testing environment. Only 45% of respondents reported on using a separate test system (Figure 5-b).

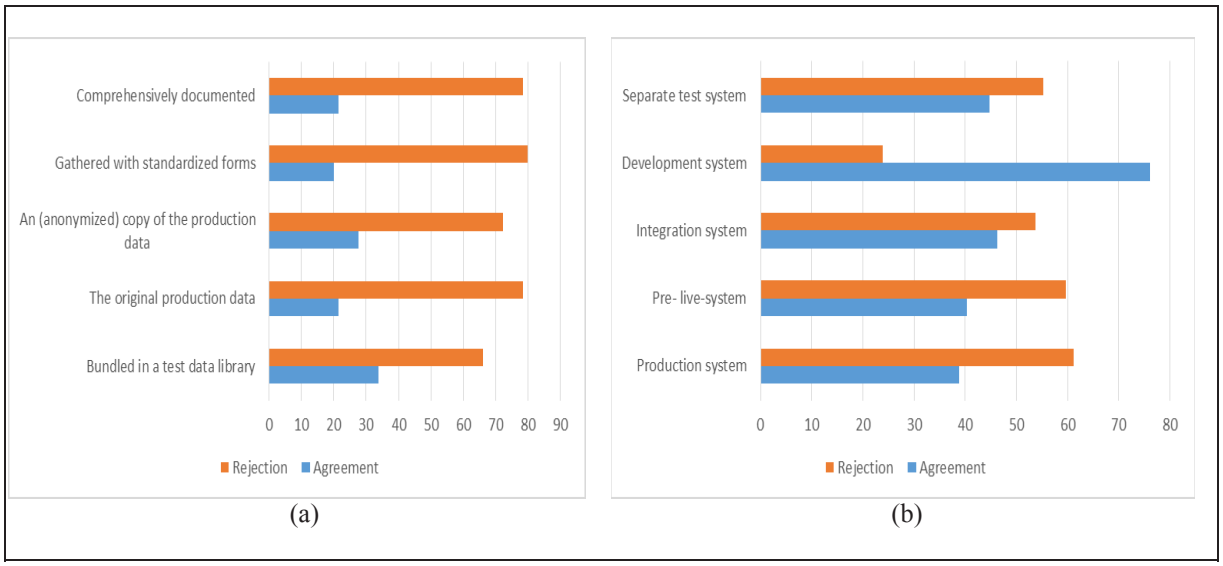

Figure 5: a- In this project, test data are / were ... (n = 113); b- Which environments are used for testing in your projects? (n = 112)

7. Test Metrics and Defects Management

The participants were asked to report on the metrics used in their projects for test control. The most used metric was the requirements coverage (66% used) followed by test case execution rate (60% used). Although a high-level test process maturity may be concluded from this, it is nonetheless 39% of the survey participants indicated that they end their test activities when the deadline for the delivery time has been reached, and 9% when the test budget is exhausted (figure 6).

The most common cause for the discovered defects was related to requirements problems (omissions, changes, and errors) -72% reported this cause (See figure 7). Design problems was the second most reported cause at 66% and Jira was the most popular tool used to report the defects (25% of all tools' usage).

Digital Object Identifier 10.1109/MS.2016.87 0740-7459/$26.00 2016 IEEE

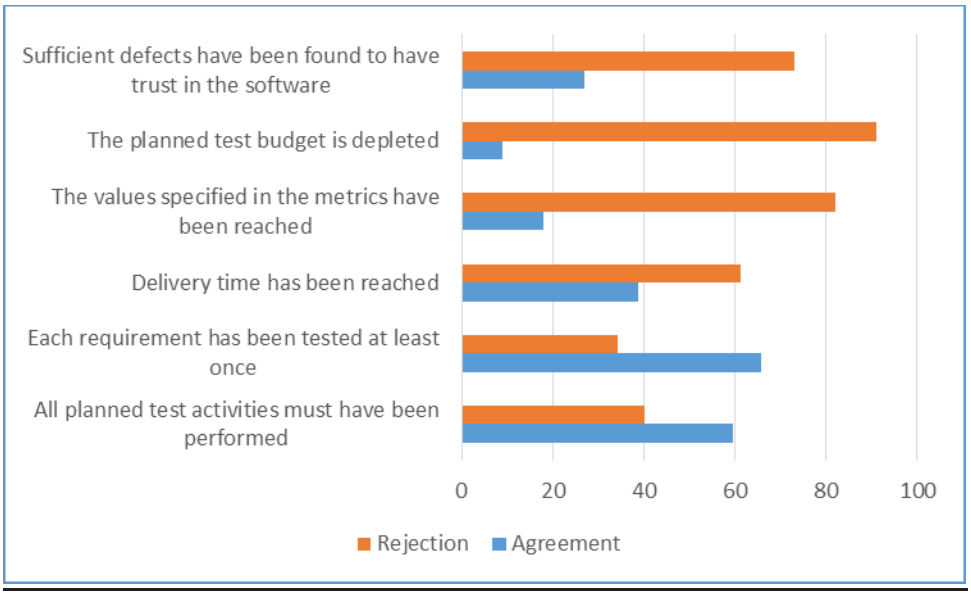

Figure 6: For test completion, the following exit criteria are used ... (n = 103)

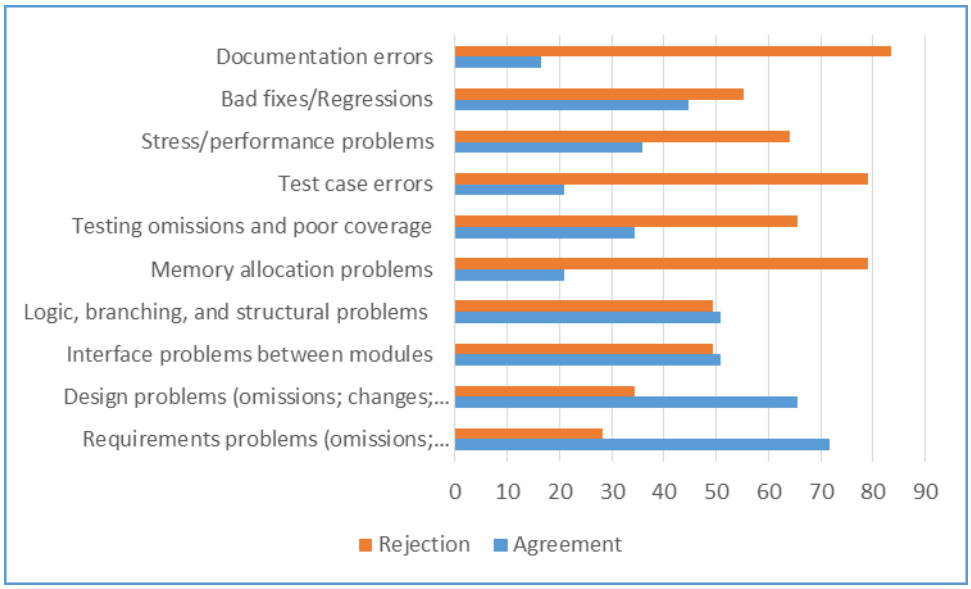

Figure 7: Which of the following are / were applicable as causes for the discovered defects in your project? (n = 98)

8. Related Work

There are few works available involving surveys of software professionals with respect to testing. For example Kasurinen et al. [7] examined the cost factor, finding that testing is often a much underestimated part of the project. These researchers found, however, that more effective testing process may reduce testing time which is often underestimated.

Others have discussed that automating testing activities can help improve test reusability, repeatability, and testing effort [8]. However, many survey results indicate that the percent of automated testing is low [3,8,9]. Supporting this fact are other studies that indicate tool adoption is also low [2, 10]. This information supports our finding that only 34% of our respondents state that the comparison of expected vs. actual results is automated and performed with the

Digital Object Identifier 10.1109/MS.2016.87 0740-7459/$26.00 2016 IEEE

help of tools. This may be due to the high cost of the tool, the training required, and the test case design effort [8]. Thus tool adoption can be a major barrier in testing [2, 10]. Clearly, organizations do not make effective use of testing tools [11]. These barriers may be the reason why Kasurinen's survey [7] indicated that the average effort spent on testing is only 25%. On the bright side, even with these barriers, Garousi et al. [10] found that automated testing has increased since 2004.

Causevic's survey results [3] indicated that the use of open source vs. proprietary testing tools depended on whether or not they were unit testing or performing higher level system testing. As mentioned earlier, our respondents reported that open source was utilized for unit testing (Junit) and higher level was performed with a proprietary tool such as Jira or HP Quality Center. Although Causevic [3] found that writing test cases before writing code is mostly not considered a current practice, our survey results showed that these approaches are becoming established in practice.

Another area of testing research is how to build an effective testing team. Kanij, Merkel and Grundy [12] conducted a survey of software practitioners, to determine the importance of factors in building testing teams. The results suggest that *experience in software testing* is more important than a team member's *interpersonal skills*. The results also suggest the desire for the testing team to be built with members having diverse *work experience* [12] as an important factor.

9. Recommendations and Conclusions

The survey results included a wide variety of raw information, but interpreting some of this information, we offer the following key findings:

- Performing quality assurance measures is concentrated on the late phases of software development.
- Practitioners prefer to have testing as a defined phase on its own.
- Systematic test case design and test data definition are not uniformly practiced.
- The overall effectiveness of test cases is perceived to be high (47%) or medium (37%).
- Only 22% of the respondents comprehensively document the test data.
- The most common cause for the discovered defects was related to requirements problems (72% reporting this observation).

We hope the survey and corresponding results stimulate research into prevailing software practices, but moreover, we intend these results to highlight the areas of software testing that need the attention of both the research community and the industry professional.

References

1. Haberl, P., Spillner, A., Vosseberg, K., Winter, M., "Software Test in Practice," released by German Testing Board, May 2011.
2. Ng, S.P., Murnane, T., Reed, K., Grant, D., Chen, T.Y., "A Preliminary Survey on Software Testing Practices in Australia," Australian Software Engineering Conference, 2004.
3. Causevic, A., Sundmark, D., Punnekkat, S., "An Industrial Survey on Contemporary Aspects of Software Testing," International Conference on Software Testing, April 6-10, 2010, pp. 393-401.
4. "Software Quality Report 2014-2015" released by Turkish Testing Board, 2014 : http://www.istqb.org/documents/TurkeySoftwareQualityReport_2014_2015.pdf
5. "ISTQB Effectiveness Survey 2013-14," 2014: http://www.istqb.org/documents/ISTQB_Effectiveness_Survey_2013_14.pdf
6. Wikipedia – Regression Testing: https://en.wikipedia.org/wiki/Regression_testing
7. Kasurinen, J., Taipale, O., Smolander, K., "Software Test Automation in Practice: Empirical Observations," Advances in Software Engineering, Volume 2010, pp 1-18.
8. Rafi, D., Moses, K., Petersen, K., "Benefits and Limitations of Automated Software Testing: Systematic Literature Review and Practitioner Survey," Automated Software Testing Conference, Zurich Sw., 2012, pp. 36-42.
9. Lee, J., Kang, S., Lee, D., "Survey on software testing practices," IET Software, Volume 6, Issue 3, 2012, pp. 275-282.

Digital Object Identifier 10.1109/MS.2016.87 0740-7459/$26.00 2016 IEEE

10. Garousi, V., Varma, T., "A Replicated Survey of Software Testing Practices in the Canadian Province of Alberta: What has Changed from 2004 to 2009?," The Journal of Systems and Software, Volume 83, 2010, pp. 2251-2262.
11. Grindal, M., Offutt, J., Mellin, J., "On the Testing Maturity of Software Producing Organizations," Proceedings of the Testing: Academic & Industrial conference, 2006.
12. Kanij, T., Merkel, R., Grundy, J., "A Preliminary Study on Factors Affecting Software Testing Team Perforamnce," International Symposium on Empirical Software Engineering and Measurement, 2011, pp. 359-362.
13. Naik, K., Tripathy, P., "Software Testing and Quality Assurance: Theory and Practice," Wiley-Spektrum; 1 edition (August 18, 2008).

Digital Object Identifier 10.1109/MS.2016.87 0740-7459/$26.00 2016 IEEE