In 3rd ICSE workshop on Co-operative and Human Aspects of Software Engineering (CHASE 2012), June 3rd 2012, Zurich, Switzerland.

Performance Assessment Metrics for Software Testers

Tanjila Kanij
Swinburne University of Technology
Hawthorn, Victoria, Australia
tkanij@swin.edu.au

Robert Merkel
Monash University
Clayton, Victoria, Australia
robert.merkel@monash.edu

John Grundy
Swinburne University of Technology
Hawthorn, Victoria, Australia
jgrundy@swin.edu.au

***Abstract*—The reliability of delivered software depends on the performance of the individuals responsible for testing it. However, there are currently no standard methods for assessing the performance of software testers, nor even agreement on important assessment criteria. A literature review reveals several different human-centric factors that have been suggested as important. However, the relative importance of these factors is unknown. This paper reports the results of a survey of professional testers indicating their views on the importance of a number of proposed factors. Their views, we believe, will be important in designing a more formal and widely acceptable performance assessment method for software testers.**

***Keywords*-performance, testing, survey, bug count**

I. Introduction

The majority of software testing research has been devoted to the enhancement and development of new techniques and tools for different types of testing [1]. Despite the immense research effort devoted to the development of techniques and tools to reduce manual involvement, adequate testing remains a labour-intensive task. Tester efforts are crucial to the success of a testing programme since the delivered reliability of software to a large extent depends on the performance of the human tester. Despite this, there is no widely accepted and well established performance assessment method. In fact, while there is anecdotal evidence suggesting various metrics, little empirical evidence exists. This absence of a formal method for performance assessment makes it difficult to distinguish among software testers exhibiting different levels of performance, both for commercial purposes and for research.

Perhaps the most obvious quantative metric to assess tester performance is the number of faults identified by a tester. Fenton and Pfleeger [2] suggest measuring program reliability, and thus the effectiveness of the software QA process using the number of bugs found per KLOC as a metric. Grady and Caswell [3] report industrial use of the metric "defects reported per working day" for the purpose of software process improvement.Kaner [4] is strongly critical of the use of simple bug counts for assessing tester skill. He notes that bug counts may be affected by unrelated factors, such as the reliability of the code they have tested, the difficulty of testing that code, and the type of testing they are performing. Furthermore, he argues that using bug counts as a metric has a number of undesirable side effects, as testers change their behaviour to maximize bug counts (for instance, by reporting large numbers of trivial bugs). Kaner [5] has proposed a multidimensional assessment method for software testers. His assessment method is primarily qualitative, and emphasizes the utility and clarity of bug reporting. In identifying and characterizing highly performing testers, Iivonen et al. [6] found that expertise in the domain helps testers to perform better. Although they didn't consider domain expertise in assessing tester performance, the relationship between domain expertise and performance indicate that it is a factor that might be used in performance assessment.

While the literature on tester performance assessment is limited, we believe it likely that a single factor is not sufficient to assess the performance of software testers. However, which factors are most informative, and how those factors might be combined to produce an overall rating for a tester is unclear. As a starting point to refine such a list of factors, we decided to ask a sample of testers what factors they considered most important in assessing their performance.

II. Research Methodology

We used a personal opinion survey for this research. This survey was designed according to the guidelines suggested by Kitchenham and Pfleeger[7] and was divided in two broad parts - "individual tester performance" and "testing team building". The first part explored how performance of a software tester can be assessed, what factors influence the performance and the importance of experience, training/certification in software testing. Since the context of this paper is performance assessment of software testers, only that portion is described here. Detailed results of the individual part of the survey have been reported separately [8]. The results on software testing team building are available in [9].

A. Objectives

The main objective of this research was to find out the respective importance of different factors in assessing the performance of software testers, from the view point of industry professionals. We listed factors identified from our literature review, experience, and feedback from a small pilot survey. We also asked participants to list more factors if they thought these were important.

B. Development of Survey Instrument

From our literature review, we listed two factors "Number of bugs found" and "Quality of bug report" that can

be important for assessing the performance of software testers. We added "Severity of bugs" to this list from our personal experience.. We also collected job descriptions of software testers from jobs advertised in the popular web site www.monster.com over a period of five days to assess employer views of software tester responsibilities. We analysed different responsibilities and found that the unique responsibilities can be classified in two broad classes - test planning and execution of tests. To assess performance on these responsibilities we added "Rigorousness of test planning and execution" to our list. From the feedback of the pilot survey, we added "Ability of bug advocacy". Finally, we listed five factors ("Number of bugs found", "Severity of bugs", "Quality of bug report", "Ability of bug advocacy" and "Rigorousness of test planning and execution") that we believed might be important for measuring the performance of testers. The question used a Likert scale with five level of agreements ("Completely disagree", "Somewhat Disagree", "Neither disagree nor agree", "Somewhat agree" and "Completely agree"). Participants indicated their level of agreement that these were important.

We also provided an open-ended question where participants could list additional factors if they chose.

The survey website was piloted with a sample of seven software engineers. Based on their feedback, we added "ability of bug advocacy" as a factor to the survey.

C. Sample Selection

We conducted a keyword-based search for software testing-related Yahoo! and LinkedIn groups, and from this initial search selected 21 Yahoo! and 29 LinkedIn groups, using purposive sampling [10] to select active groups related to professional testing. An email requesting permission to send an invitation email to the group was sent to the selected group moderators. Moderators of 12 LinkedIn and 12 Yahoo! groups approved our request. The group response rate was 41.4% for the selected LinkedIn and 57.1% for the selected Yahoo! groups. Quantifying the individual response rate was impossible, as the overlap of group membership is unknown; nor is it known what proportion of group members actually read our invitation.

III. Results

A. Demographic Information

There were a total of 104 responses. The majority of the respondents (71.8%) were male. Around 75% of the respondents were between 18-40 years of age. 28.8% of respondents reported their "Country" as India, with the second largest group of respondents (24%) coming from the United States. The balance of responses came from a wide variety of other nations. Nearly 60% of respondents were employed by large[1] IT companies; with a little under 20% of the sample employed by smaller IT companies, and a similar proportion (17.3%) employed by larger non-IT companies . Almost half of the respondents (49%) had more than five years of job experience. Table I indicates the respondents' main job responsibilities[2].

Table I
Main Job Responsibilities

Developing software module/program, and testing self developed modules/programs	7.7%
Developing software module/program, and testing modules/programs developed by others	11.5%
Testing modules/programs developed by others	76.9%
Manage Software Testers within a project	46.2%
Others	19.2%
No Response	2.9%

B. Assessment of Performance

The responses to the listed factors that can be important for assessing the performance of software testers are shown in Figure 1. We can see that, except for "Number of bugs", the distribution of responses to the different factors were quite similar. The level of agreement is highest for "Bug report quality" and "Rigorousness of testing". In order to test whether the responses to different choices for a question differ significantly from each other, we converted the Likert response categories to integers. A Kruskal-Wallis test showed that the perceived importance of the factors differed significantly (p ¡ 0.05). Post-hoc Tukey's HSD tests showed that "Number of bugs found" was considered significantly (p ¡ 0.05) less important than any other factor, and that that "Quality of bug report" was considered significantly (p ¡ 0.05) more important than "Severity of bugs found". There were no other significant pairwise differences.

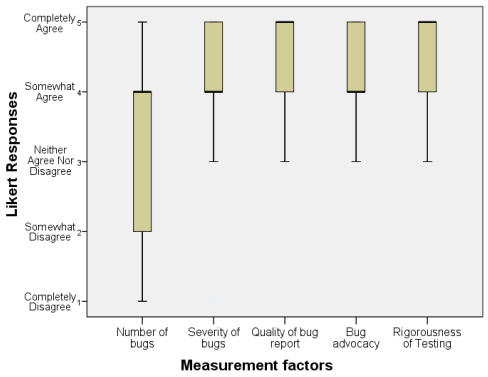

Figure 1. Responses on "Factors important in measuring performance of software testers"

We analysed the responses to the open question, identifying and categorizing common words and phrases. The frequency of responses of the identified categories is reported. 31.7% of the respondents noted other factors as important, for instance, the quality of the communication with developers (8.7%), domain knowledge (6.7%) and understanding of

[1]Large defined as having more than 50 employees

[2]Respondents were able to select more than one option

requirements (4.8%). Other responses to the open question included (in order of frequency of occurrence): analytical ability, implementation of plans, creativity, level of testing automation, and preventative teaching to the developers. Interestingly, one respondent suggested that the performance of a software tester can be measured by the number of bugs reported in the "live" environment (after deployment).

IV. Threats To Validity

Misinterpretation of survey questions by respondents threatens our study's internal validity. However, we see no evidence of this occurring. Another potential threat is random or less than candid survey responses, which is a common issue in this kind of study. We see no evidence that it occurred in our data, and no particular motivation for participants to do so. One possible threat to the external validity is the representativeness of the respondents. As a voluntary survey with an unknown response rate, the survey does not represent any kind of random sample. Additionally, this study seeks only the thoughts and views - the "common wisdom" of expert testers. That common wisdom may be wrong.

V. Discussion

From our results, it is clear that, contrary to currently perceived wisdom in many circles in software engineering and in line with Kaner [4], our respondents do not think "Number of bugs" is a good measure of tester performance, preferring more human-centric measures such as bug report quality. However, our survey does not provide data on the reasons for this belief.

The distribution of responses on the other factors listed in the closed question was similar. However, the post hoc tests show that our respondents regard "Quality of bug report" is more important than "Severity of bugs found". The importance of quality of bug report is supported by Kaner's [5] proposal of emphasizing on the quality and employing qualitative assessment of the bug report. However, we remain unconvinced that this result is applicable to all software projects - for instance, would it apply to safety critical software?

Responses to the open questions were also enlightening. For instance, it is notable that our respondents identified "the quality of the communication with developers" as an important factor. As "quality of bug report" was a factor explicitly mentioned in the survey, it seems that communication other than through bug reports is also considered important. "Domain knowledge", was found to be an attribute of highly performing testers by Iivonen [6]. However, it is not clear whether it is an appropriate criterion to assess the work of software testers. A tester acquires domain knowledge for the purpose of performing more effective testing; the domain knowledge is not the end product. Logically, where the end product of the testing can be assessed directly, it makes more sense to judge based on outputs rather than contributions to that net output. However, surprisingly, only one tester in our survey mentioned "number of bugs in the live environment" as a useful metric. Further investigation is required to determine whether testers and other stakeholders indeed do not consider this to be an appropriate metric, and if not, why they do not.

VI. Conclusion

Performance assessment of software testers appears to be an area where little empirical research has been conducted, but a number of different factors for performance assessment of testers have been proposed. We attempted to rate the comparative importance of these different factors. The results indicate that of all the factors we enumerated in our survey, "Number of bugs found" was perceived as the least important for performance assessment of testers. There was also some indication that "Bug report quality" was considered particularly important.

We believe that this area is worthy of further research, and based on our work to date, we are designing a new Performance Appraisal Form (PAF) to permit structured, formal performance appraisal of testers. Such a PAF will, in itself, need to be empirically evaluated.

References

[1] Bertolino, Antonia, "Software testing research: Achievements, challenges, dreams," in *FOSE '07: 2007 Future of Software Engineering*. Washington, DC, USA: IEEE Computer Society, 2007, pp. 85–103.

[2] N. Fenton and S. L. Pfleeger, *Software metrics (2nd ed.): a rigorous and practical approach*. Boston, MA, USA: PWS Publishing Co., 1997.

[3] R. B. Grady and D. L. Caswell, *Software metrics: establishing a company-wide program*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1987.

[4] C. Kaner, "Don't use bug counts to measure testers," *Software Testing & Quality Engineering*, p. 80, May/June 1999.

[5] ——, "Measuring the effectiveness of software testers," *Software Testing Analysis & Review Conference (STAR) East*, May 2003.

[6] J. Iivonen, M. V. Mäntylä, and J. Itkonen, "Characteristics of high performing testers: a case study," in *Proceedings of the 2010 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement*, ser. ESEM '10. New York, NY, USA: ACM, 2010, pp. 60:1–60:1.

[7] F. Shull, J. Singer, and D. I. Sjøberg, *Guide to Advanced Empirical Software Engineering*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2007.

[8] R. Merkel and T. Kanij, "Does the individual matter in software testing?" http://www.swinburne.edu.au/ict/research/sat/-technicalReports/TC2010-001.pdf.

[9] T. Kanij, R. Merkel, and J. Grundy, "A preliminary study on factors affecting software testing team performance," in *ESEM*, 2011, pp. 359–362.

[10] M. Denscombe, *The good research guide for small-scale social research projects*. Milton Keynes, UK: Open University Press, 2003.