Performance Appraisal of Software Testers

Tanjila Kanij[1] and John Grundy

Swinburne University of Technology, Melbourne, Australia

Robert Merkel

Monash University, Melbourne, Australia

Abstract

Context: To determine the effectiveness of software testers a suitable performance appraisal approach is necessary, both for research and practice purposes. However, review of relevant literature reveals little information of how software testers are appraised in practice.

Objective: (i) To enhance our knowledge of industry practice of performance appraisal of software testers; and (ii)to collect feedback from project managers on a proposed performance appraisal form for software testers.

Method: A web-based survey with questionnaire was used to collect responses. Participants were recruited using cluster and snowball sampling. 18 software development project managers participated.

Results: We found two broad trends in performance appraisal of software testers - same employee appraisal process for all employees and a specialized performance appraisal method for software testers. Detailed opinions were collected and analysed on how performance of software testers should be appraised. Our proposed appraisal approach was generally well-received.

Conclusion: Factors such as number of bugs found after delivery and efficiency of executing test cases were considered important in appraising software testers' performance. Our proposed approach was refined based on the feedback received.

Keywords: Performance appraisal, Software testers, Project management

1. Introduction

The reliability of delivered software, to a large extent, depends on the performance of software testers. An accurate performance appraisal of soft-

[1]Corresponding author, Tel: +61392148840, Email: tkanij@swin.edu.au

ware testers is thus very important for their recruitment, monitoring and development, and for testing team performance management. Furthermore, from a research perspective, to conduct studies of factors that potentially affect software testers' performance, a validated, reliable instrument to assess software testers' performance is an essential prerequisite. For example, in a research study [1] investigating the influence of personality on the effectiveness of software testing the authors needed to distinguish different levels of performance. To accomplish this, a method to assess high performing software testers was necessary.

However, from an extensive search of relevant literature we did not find any widely accepted and well established performance appraisal method for software testers. Therefore, as an operational need for our research, we have proposed a new Performance Appraisal Form (PAF) for software testers. However, any such instrument should be validated for use.

In this study, we sought a broader insight into industrial practice in the area of tester performance appraisal by surveying nearly 20 software development project managers to describe the practices in their own organizations, and their own views on tester performance appraisal. We then attempted to validate the approach taken in our PAF by collecting feedback from software development project managers on the proposed PAF, obtaining detailed feedback from 10 of them. With this two-pronged approach, we not only sought direct comment on our proposed PAF, we hoped to find out whether industrial practice could further inform our PAF design, and also whether the PAF proposed could be of industrial as well as research interest.

The rest of the article is arranged as follows- Section 2 summarises our review of relevant literature, Section 3 details our research questions, Section 4 describes the proposed Performance Appraisal Form (PAF), Section 5 describes the method of this research study, Section 6 presents our results, Section 7 lists the threats to validity of the research, Section 8 presents our discussion on the findings and finally Section 9 concludes the article.

2. Related Work

2.1. Performance evaluation of software testers

As reported in our earlier research [2], there is no widely accepted instrument or approach to evaluating the performance of software testers in the academic literature. However, we found some suggestions about criteria that may be important for evaluating software testers' performance. Fenton

and Pfleeger [3] suggest measuring efficiency of software testing using the number of bugs found per KLOC. Grady and Caswell [4] suggest looking for average reported bugs per working day. However, Kaner [5] has discouraged considering only bug counts to measure software testers' efficiency as bug counts are influenced by reliability of code being tested, difficulty of testing the code, and the testing techniques being used (for example, exploratory and regression testing will produce different bugs).

Kaner [6] proposed a multidimensional assessment method for software testers, emphasizing qualitative assessment of testers' plan of testing, execution of tests and bug reports. He suggests the reviewer conducts short, regular discussions with testers regarding their test progress to obtain information. Kaner's proposed approach is a plausible way to evaluate testers; however, it is not supported by any research results. In addition, the evaluation approach is time consuming and is dependent on the perception of the reviewer. Appropriate manager training and experience is needed to successfully carry out this type of evaluation.

In our earlier survey [2] we found that bug report quality was considered important in assessing software tester's performance.

2.2. Performance evaluation of other software practitioners

Killingsworth et al. [7] described a model to motivate and evaluate information systems staff using five factors: product quality, customer outreach, staff development, administrative efficiency and fiscal responsibility. A senior project manager and team leader assess each employee on each of the five factors with varying weights for the reviewer. For example product quality accounts for 40% of the review for a team leader and 20% of the review for a senior manager.

Mayer and Stalnaker [8] describe a number of methods that are useful for selection and evaluation of computer personnel. While most of the methods presented in their paper are useful for selection of programmers, very few of those can be used for evaluation as well. These include: Dickmann's [9] Programmer Appraisal Instrument (PAI) with four performance areas- professional preparation and activity, programmer competence, dealing with people and adapting to the job; Bairdain's approach [10] considering the following factors- programming knowledge/capability, working style, temperament traits and personal professional items and Berger and Wilson's [11] Basic Programmer Knowledge Test (BPKT) evaluating programmer's knowledge on six areas- logic estimation and analysis, flow diagramming, programming

constraints, coding operations, program testing and checking, and documentation.

Powell [12] presented 13 categories to be used to rate programmers and analysts. Some of the categories were tact and diplomacy, project selection, project planning, self expression- written and oral, ability to complete the job, and supervision. He defined each of the categories and proposed a distribution of performance according to his method for a group of 20 programmers and analysts.

The proposals for programmer assessment indicates similar instruments can be developed for software testers as well.

3. Research Questions

Current methods utilized to appraise software testers in the software industry have not been reported or evaluated in any detail in the open literature [2]. We therefore do not know what are the relative advantages and disadvantages of current performance appraisal practices for software testers and how we can improve on these.

Based on our review of the literature and analysis of different requirements listed in job advertisements for testers, we designed a new Performance Appraisal Form (PAF) for software testers. We then wanted to collect feedback on the appropriateness of our proposed PAF and to make suggested improvements. A brief description of the proposed PAF is given in the following subsection. The PAF itself is available at:

http://www.testingsurveys.org/PAF_static/initialPaf.html

In this study, we attempt to answer four research questions via a two-part practitioner survey:

- How is performance of software testers currently appraised in industry?
- What are the advantages and disadvantages of current performance appraisal methods used for software testers?
- How can the currently used performance appraisal methods for software testers be improved?
- What do software project managers think of our proposed PAF for software testers?

4. Proposed Performance Appraisal Form (PAF)

The objective of our proposed PAF is to provide a standard assessment instrument to assess overall performance of software testers from different performance dimensions. Some performance appraisal instruments use multiple forms to assess different aspects of employee performance. However, for simplicity we chose to design an integrated form. The performance dimensions of our proposed PAF were based on different approaches [13] to performance appraisal: *Performer focused appraisal :* This approach attempts to discern whether some qualities are exhibited by the performer or not. *Work behaviour based appraisal -* This approach judges the performance on the work behaviour of the performer. *Result focused appraisal:* This approach includes assessment of performance based on predefined goals and objectives.

For software testers, how effectively testing has been carried out and how efficiently the testing contributed to the reliability of the software, are important. We also believe there are some general skills that are important to be high performing software testers. The appraisal form, therefore include different rating dimension on work behaviour, work outcome and personal attributes, with seven dimensions in total.

In order for the better the understand-ability of the readers, before describing the dimensions of our proposed PAF, we discuss some of the scale types used to evaluate on those dimensions. We have used three types of rating scale: behaviour frequency scales, compare against standard scales and evaluation concept scales [13]. A behaviour frequency scale considers the occurrence of defined behaviour; labels such as "always", "seldom", and "never" and used. Standard scale type compares the performance against a standard with labels such as "exceeds standard", "below standard" and so on. The evaluation concept scale judges the quality of performance and the associated labels are typically "outstanding", "marginal", "unsatisfactory" and so on.

Dimensions related to work outcome: Two work outcomes considered to be included in the dimensions related to work outcome of a software tester are bug report and number of bugs reported (bug count).

Bug report: Kaner [6] emphasized on the qualitative assessment of bug report based on- ease of understanding, sufficient information to replicate the bug, short and precise description, absence of unnecessary information and using polite tone for communication. We designed two performance dimensions on the quality of bug report according to the suggestions of Kaner.

Dimension 1- Bug report (ease of understanding): This dimension helps to assess the qualities of the bug reports that are important to make those understandable. The dimension uses evaluation concept scale labels [13] with five choices since this scale is designed to evaluate the quality of the attribute.

PAF: SECTION ONE - TEST OUTCOME

DIMENSION 1- BUG REPORT (EASE OF UNDERSTANDING):

DESCRIPTION: The extent to which bug reports are easy to understand and precise. Qualities of the bug report to be considered include: clear description of faulty behaviour; short and to-the-point description of fault; and use of a polite tone. Please consider all the bug reports produced by the software tester during the period of evaluation and select on the basis of the *average* report quality.

INSTRUCTION: Please consider bug reports (both- written and verbal) produced by the software tester being appraised using this dimension and indicate the rating you will give to those reports.

Label	Score	Definition	Rating
Perfect	5	Recipient gains a complete understanding of the issue identified in the report after reading it once. Reports are always clear, precise, polite, and attractive to the reader.	○
High	4	It is easy to understand the issues in the reports. Reports are usually clear, precise and polite. Usually contains only relevant information and rarely includes distractions.	○
Satisfactory	3	Reports can be understood. However, may not always be clear, precise and short. Sometimes distracts the reader with extraneous information. Sometimes the reader needs to read the report multiple times to completely understand it.	○
Difficult	2	Difficult to understand what the software tester wanted to describe, sometimes imprecise, and/or use of impolite tone.	○
Impossible	1	It is not possible to understand what the software tester attempts to describe in bug reports.	○

Figure 1: Dimension 1 of our proposed PAF

Dimension 2- Bug report (ease of replication): This dimension evaluates the presence of sufficient information to replicate the bugs addressed in the reports. This dimension also uses evaluation concept scale labels for the same reason as the previous dimension.

Number of bugs found: We have designed two performance dimensions related to the number of bugs found. However, taking into account the drawbacks as outlined by Kaner [5] of considering only raw bug count, we have incorporated two mitigating criteria - the severity of found bugs and the difficulty of finding the bugs - that produce extra context about the bug detection performance.

Dimension 3- Bug count (compared to the ease of finding the bug): The number of bugs found by a tester is directly dependent on the ease of testing the code. We have defined three levels of difficulty from our experience and

DIMENSION 2- BUG REPORT (EASE OF REPLICATION):

DESCRIPTION: The extent to which reported bugs can be easily replicated from the bug reports. Qualities of the bug report to be considered include: the presence of all necessary information; clear description of steps to reproduce the bugs; and the presence of necessary details regarding input and environment. Please consider all the bug reports produced by the software tester during the period of evaluation and select a rating based on the average quality.

INSTRUCTION: Please consider bug reports (both- written and verbal) produced by the software tester being appraised using this dimension and indicate the rating you will give to those reports.

Label	Score	Definition	Rating
Perfect	5	Bugs can be replicated based on the bug reports. Sufficient information about the required sequence of actions, input and environment is provided. No unnecessary information is provided.	○
High	4	Bugs can be replicated based on the bug reports. Necessary information is clear in the report however may not always be to-the-point.	○
Satisfactory	3	Bugs can be replicated based on the bug reports, however not all information is present and clear in the report. Sometimes unnecessary information is provided, or steps to replicate are confused or "mixed up".	○
Difficult	2	It is difficult to replicate the bugs based on the bug reports only. Necessary information is missing or poorly presented, and/or too much unnecessary information is provided.	○
Impossible	1	Completely impossible to replicate the bugs from the information in the bug reports. Necessary information is unavailable.	○

Figure 2: Dimension 2 of our proposed PAF

assigned varying weights. The standard scale labels [13] is associated with this dimension. The standard is considered as the average number of bugs which is highly dependent on the project, so no range or number is specified and should be decided by the appraiser.

Dimension 4- Bug count (compared to the severity of found bugs): This dimension considers the severity of the found bugs in regard to the frequency of finding those. Four levels of severity are adopted from [14]. The weights of the different levels of severity are assigned from our experience.

Dimensions related to work behaviour: We have collected and analysed the job descriptions of software testers in the popular recruitment web site [15] over a period of five days. We found the responsibilities of testers can be classified in two broad classes - test planning and execution of tests.

Dimension 5- Assessment of performance in test planning: This dimension uses frequency scale labels [13] with five choices. Since this dimension is related to work behaviour and frequency scale helps to assess how often certain behaviour is displayed, frequency scale was considered most appropriate for this dimension.

Dimension 6 - Assessment of performance in executing tests: This di-

DIMENSION 3- BUG COUNT (COMPARED TO EASE OF FINDING):

DESCRIPTION: Number of bugs found in comparison to difficulty of finding them.

INSTRUCTIONS: In the following table five labels of frequency of finding bugs are defined in the vertical direction (columns) and three levels of bug "difficulty" (how difficult the bug in question is to find - please note that this is a subjective judgement) are defined in the horizontal direction (rows). Please consider the bugs reported by the software tester being appraised and, for each level of difficulty, select the box indicating the relative number of bugs found by this tester. Since the average number of bugs, and difficulty levels, are dependent on the project, this should be decided by the appraiser.

Difficult to find bugs are weighted more heavily in calculating the tester's rating in this dimension.

Label	Bug Count		Very high	High	Average	Low	Very low
Difficulty of finding	Definitions		Found well above average number of bugs	Found above average number of bugs	Found average number of bugs	Found below average number of bugs	Found well below average number of bugs
		score weight	5	4	3	2	1
Difficult	Very difficult to find bugs (Not easily found by all)	0.5	○	○	○	○	○
Normal	Bugs with average difficulty of finding (Needs careful attention)	0.3	○	○	○	○	○
Easy	Obvious bugs	0.2	○	○	○	○	○

Figure 3: Dimension 3 of our proposed PAF

DIMENSION 4- BUG COUNT (COMPARED TO SEVERITY):

DESCRIPTION: Number of bugs found in comparison to severity of the bugs.

INSTRUCTION: In the following table, there are four rows listing different bug severity levels. For each severity level, you should indicate the frequency with which the tester finds bugs of that severity. The five columns represent five frequency categories, from "much higher than average" to "much lower than average". Since the "average" bug count is project-specific, the appraiser should use their judgement to decide what an "average" level is and how the tester being appraised compares.

Label	Bug Count		Very high	High	Average	Low	Very low
Severity	Definitions		Found well above average number of bugs	Found above average number of bugs	Found average number of bugs	Found below average number of bugs	Found well below average number of bugs
		score weight	5	4	3	2	1
Critical	Program ceases meaningful operation	0.4	○	○	○	○	○
High	Severe function error but application can continue	0.3	○	○	○	○	○
Medium	Unexpected result or inconsistent operation	0.2	○	○	○	○	○
Low	A cosmetic issue	0.1	○	○	○	○	○

Figure 4: Dimension 4 of our proposed PAF

PAF: SECTION TWO - TEST BEHAVOUR

DIMENSION 5- TEST PLANNING:

DESCRIPTION: Frequency of preparing good quality test plans. Quality attributes of a test plan include: efficiency in finding bugs; ability of assessing high risk area; and selection of efficient test strategy.

INSTRUCTIONS: Please consider the test plans made by the software tester being appraised using this dimension and indicate the rating you will give to those plans.

Label	Score	Definition	Rating
Always	5	Consistently prepares satisfactory test plan.	○
Mostly	4	Almost always prepares satisfactory test plan.	○
Usually	3	Sometimes prepares satisfactory test plan.	○
Rarely	2	Test plan is most often not satisfactory.	○
Never	1	Test plan is never of a satisfactory standard.	○

Figure 5: Dimension 5 of our proposed PAF

mension also uses frequency scale labels with five choices for the same reason as stated above.

DIMENSION 6- TEST EXECUTION:

DESCRIPTION: Frequency of optimizing execution of tests in allotted time frame. Attributes of test execution to be considered include: errorless execution of tests; and good assessment of test outcome.

INSTRUCTION: Please consider the tests executed by the software tester being appraised using this dimension and indicate the rating you will assign to those execution of tests.

Label	Score	Definition	Rating
Always	5	Test execution is consistently completed to a satisfactory standard within allotted time.	○
Mostly	4	Test execution is almost always completed to a satisfactory standard within allotted time.	○
Usually	3	Sometimes test execution remains incomplete.	○
Rarely	2	Test execution remains incomplete very often.	○
Never	1	Test execution is never completed to a satisfactory standard within allotted time.	○

Figure 6: Dimension 6 of our proposed PAF

Dimensions related to personal attributes: We have listed the soft-skills or qualities mentioned in job advertisements of software testers in recruitment web site[15]. We have also reviewed related literature and found skills like good domain knowledge [16; 17] are important. We have designed the seventh dimension with these personal attributes of a software tester.

Dimension 7- Personal attributes of a tester: This dimension uses compare against a standard scale labels [13] to assess whether the software tester possesses the following personal attributes: domain knowledge, adaptability to new tools and techniques for testing, communication skill, attention to details and ability to handle complex technical aspects. Different weights are assigned to those attributes from our experience.

Overall score: this is calculated by summing individual scores and dividing by 7. Modifications could include weighting different sections more or less heavily depending on manger and organisational needs and perceived inter-relationships between scores. It should be noted that scores in one dimension may (or may not) influence another e..g it could be the case that a good rating in the test planning dimension implies a good performance in the bug count dimensions. This may be an appraisers interpretation and scoring result though we do not explicitly make these links. Some appraisers may

PAF: SECTION THREE - PERSONAL ATTRIBUTES

DIMENSION 7- PERSONAL ATTRIBUTES:

DESCRIPTION: The extent to which the software tester being appraised possesses certain personal attributes.

INSTRUCTION: In the following table some personal attributes have been listed and different weights have been assigned to those attributes. Please consider the software tester being appraised using this dimension and indicate the label of possession of the listed attributes.

Attributes	Exceptional	Exceeds expectations	Meets expectations	Below expectation	Needs Improvement	
	5	4	3	2	1	Score Weight
Domain knowledge	○	○	○	○	○	0.3
Adaptability to new tools and techniques of testing	○	○	○	○	○	0.2
Communication skill	○	○	○	○	○	0.1
Attention to detail	○	○	○	○	○	0.3
Ability to handle complex technical aspects	○	○	○	○	○	0.1

Figure 7: Dimension 7 of our proposed PAF

similarly treat the dimensions as completely unrelated in their own usage and scoring.

Similarly, the PAF and overall scoring could be augmented by other dimensions. For example, scoring the tester's ability to work well in a testing team. These can be incorporated by an organisation or specific testing team manager.

Our PAF is intended to be used to appraise any and all testers carrying our any types of tester tasks. Again, specific focused questions or subdimensions could be added asking about performance relating to very specific testing tasks e.g. unit testing, inspections, performance testing and so on. Scoring could be used to influence the score for a particular dimension or the overall score, depending on manager and organisational needs.

Using the PAF: Testing managers complete the PAF and discuss results with the tester. The overall score, while numeric, is qualitative as it is based on the qualitative scores entered previously. The idea is to identify areas of strength and weakness that can be addressed in the tester under appraisal. Over time, results from each individual question and overall scores can be charted to help inform further tester development. We envisage time vs

PAF: SECTION FOUR - OVER ALL PERFORMANCE ASSESSMENT

DESCRIPTION: The over all performance assessment score is obtained by dividing the summation of individual scores in seven dimensions by 7.

INTERPRETATION OF OVERALL SCORE: The over all performance assessment score is interpretated according to the following table.

Over all score	Interpretation
0-0.99	Poor
1-1.99	Marginal
2-2.99	Satisfactory
3-3.99	Good
4-5	Outstanding

COMMENT ABOUT OVER ALL PERFORMANCE: Please provide your comment about the overall performance

Figure 8: Overall score of our proposed PAF

rating line charts. The results from a whole testing team could be graphed with bar charts of number vs rating.

5. Methodology

A personal opinion survey [18] was used to conduct our survey. Compared to other available research methods, a survey enabled us to collect opinions of higher number of participants in limited available time [19]. The design of the survey was by following the steps as suggested by Kitchenham and Pfleeger [18], and these are presented in the following subsections.

5.1. Setting the Objectives

The two main objectives of the research were to (i) collect information about the state of practice of performance appraisal of software testers; and (ii) collect feedback on a proposed Performance Appraisal Form (PAF) for software testers. Accordingly, the research was divided in two sections.

5.2. Survey Design

We used a web-based survey and prepared a data collection tool containing a self-administered questionnaire. The benefit of using a web-based, self-administered questionnaire is that the participants can respond at their own convenience. A potential disadvantage of web-based survey is the generalization of the sample.

5.3. Development of Survey Instrument

According to the suggestion of Kitchenham and Pfleeger [18], we searched for relevant research studies in the literature before developing our own survey instrument. However, we did not find any research study we could adopt some questionnaire or designing tips from. As such we developed our own instrument. Since our aim was to collect opinion and feedback, most of the data collected was in the form of open-ended free-form text.

5.4. Questionnaire design

Opinion on state of practice - The section asked whether there is a formal performance appraisal process for employees, is there any specialized appraisal process for software testers or not, whether the performance appraisal of software tester in practice was considered sufficient and appropriate or not and in their opinion how software testers performance should be appraised.

Feedback on proposed PAF - In our initial design we planned to evaluate the proposed PAF using the following two steps: *Step 1:* We requested participants to consider a software tester worked under their supervision. We asked them to rate the software tester's overall performance using a scale of five with rating labels of poor, marginal, satisfactory, good and outstanding. *Step 2:* The managers were then asked to rate the same software tester's performance using our proposed appraisal form. The managers were requested to rate more than one tester working in their supervision by repeating the steps above. The overall score obtained from using our form and the overall score the manager had assigned before completing our form would be compared to check the validity of our proposed form in obtaining the right appraisal score. Once managers rate their testers, they would be given a survey form where the managers could give their feedback about our proposed appraisal form. The attempt to validate our proposed PAF by collecting feedback from managers was chosen since managers usually perform performance appraisal in practice and as such they could comment whether performance appraisal conducted with proposed PAF actually helps them to systematically assess testers' performance and reflects their judgement.

Unfortunately, due to a poor response rate (14 participants, though the actual response rate cannot be determined as invitations were sent to the groups) to our initial PAF appraisal survey we had to revise this process and plan a more lightweight survey based on feedback from participants and potential participants that the original was too detailed and too time-consuming. We modified our survey design and participants were no longer

requested to use our proposed performance appraisal form. The form was presented to the participants to review followed by a simplified feedback questionnaire asking them whether they thought the performance dimensions considered in the proposed PAF were sufficient and appropriate or not, how the proposed PAF could be improved, whether the weight assignment to different attributes were appropriate or not and so on. Most of the feedback questions used closed Likert scale responses.

5.5. Evaluation of Survey Instrument

To evaluate the survey instrument 11 participants including professional software engineers, academics teaching software engineering and PhD students with prior industry experience were requested to participate in a pilot survey. Based on the feedback of six participants (54.5%) who responded, definition of a severity class was modified.

5.6. Sampling to Obtain Valid Data

We used cluster and snowball sampling in this survey [19]. In cluster sampling, instead of selecting individuals from the population randomly, clusters of individuals are selected and within one cluster all individuals are included in the sample. In applying cluster sampling, permission to send invitation email was requested from the 12 LinkedIn and 12 Yahoo! groups that had approved us in our preliminary survey [16; 20; 2]. Three LinkedIn and four Yahoo! groups permitted us to post to the group this time, making the group response rate 25% and 33.3%, respectively. It is, however, impossible to calculate an accurate individual response rate, since most of the invitations were sent to the groups and the number of group members who actively read emails cannot be obtained.

Snowball sampling, on the other hand, is a process where samples are selected through references. The authors invited participants from their personal contacts and requested the invited participants to nominate more participants. Unfortunately we did not find any participant in the initial survey, however, we obtained 3 participants in lightweight survey with snowball sampling.

5.7. Data Analysis

We used grounded theory [19] to analyse open-ended responses. In this analysis process the researchers read the data multiple times and assign codes to the data according to the interpretation made by the researchers. Similar

codes are then grouped together to form categories and categories are analysed to develop a hierarchy. The key concepts are found from the hierarchy.

6. Results

Our survey was divided into two main sections. We noticed that we obtained a different rate of participation for the two sections. Unfortunately, 22.2% of participants dropped out after completing the first section. Thus we describe the results obtained from each section separately.

A total of 18 participants (8 in the initial survey and 10 in the new lightweight survey) participated in this section.

6.1. Demographic information

The majority of our participants in our sample were male (83.3%). This is not surprising since the majority of practitioners in the IT field are male [21]. We assume the gender ratio is similar for software testers since female participants were in the minority in all our previous studies [16; 20; 2]. Participants were distributed over a number of countries, with the most coming from (in order of frequency) Bangladesh (27.8%), Australia (22.2%), Canada (11.1%) and United Kingdom (11.1%). There were also participants from China (5.6%), Egypt (5.6%), United States of America (5.6%), Hungary (5.6%) and Romania (5.6%). Most participants (72.2%) worked in IT organizations with little (22.2%) working in non-IT organization and 5.6% being self employed. Since the intended participants of this survey were software development project managers, we did not explicitly ask the role of the participants. Around 40% (38.9%) participants had between 1 to 3 years experience, 22.2% had between 3 to 5 years of experience, 22.2% had more than 5 yesrs of experience in managing testers.

Participants came from small to large organizations. The majority of participants (44.4%) reported the organizations they worked in had 11-50 employees, 16.7% reported having 251-1000 and 1000+ employees each, 11.1% reported having 50-250 and 10000+ employees each and 5.6% reported having 1-10 employees.

6.2. State of practice

6.2.1. Current performance appraisal approaches for software testers

We asked the participants whether the organization they work in practiced a formal process of employee appraisal and whether there is any spe-

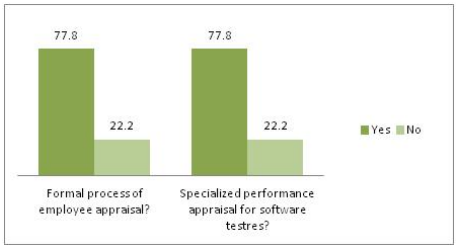

Figure 9: Responses on performance appraisal practice

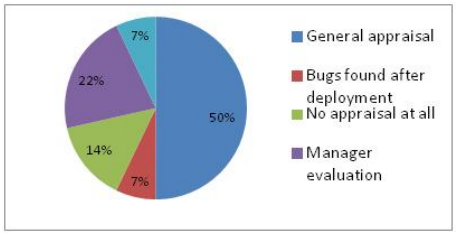

Figure 10: Responses on "How performance of software testers is appraised (not using specialized performance appraisal method or process)"

cialized performance appraisal method/form used for software testers. The responses to these questions are summarized in Figure 9.

From Figure 9 we can see that 78% of the organizations our participants came from, conduct a formal process of employee appraisal. A similar number of organizations (77.8%) have specialised processes or methods for performance appraisal of software testers. For these we asked whether they thought the specialized process or method was adequate and sufficient. We also asked whether the specialized method or process were designed for custom use by the organization. 50% of those participants (77.8% of total) indicated that their specialized performance appraisal process or method was designed specifically for their organization. All were satisfied with the customized method. However, responses were divided on the sufficiency of performance components considered in the specialized process or method. Only 50% mentioned the performance components were sufficient.

On the other hand, those who reported that they did not have a specialized appraisal method or process for software testers were asked how performance of software testers is appraised in those organizations. Different views were obtained in response to this question. The responses are shown in Figure 10.

From the responses, we see that about half of the organizations (50%) that do not have any specialized performance appraisal process or method for their software testers use a "general appraisal method". Sometimes the same appraisal method was used for software testers and programmers, sometimes a common HR appraisal policy was followed for all employees. These are grouped under "general appraisal method". The second most common practice was "manager evaluation". Different methods of manager evalu-

Table 1: Performance criteria with respective frequency of occurrence

Performance criteria	Number of time mentioned
Bugs found after delivery, Number of test cases run with regards to time	4 each
Agreed upon KPIs	3
Number of bugs found, Agreed upon personal goals, In depth code inspection	2 each
Bug report quality, Severity of bugs found, Teamwork, Skill on applying testing methodology, Technical knowledge, Knowledge of the product, Use of formal QC/QA methodology, Innovativeness of testing requirement analysis, Depth of testing, Thought process behind discovery of bug, Time spent on type of bugs (cosmetic vs crucial), Number of buggy tests, Number of automated test cases, Behavioural or functional or requirement coverage, Efficiency of test case design, Communication skill	1 each

ation were described by our participants. These include managers setting goals and evaluating the performance based on these goals; managers sitting closely by the software testers and evaluating them based on their activity; and software testers evaluating themselves and manager evaluation taking place afterwards. Surprisingly, in 14% of those organizations, no appraisal method at all is practiced for software testers. Only one participant mentioned that in their organisation software testers' performance was evaluated based on the bugs found in the live environment.

6.2.2. Suggestions on how software testers' performance can be appraised

In total 17 participants (94.4%) gave their views on how the performance of software testers can be appraised. Their responses were broad and detailed. From the detailed responses we see two major themes. One group of participants (29.4%) thought that software testers' performance should be appraised using the same process as is practiced for others. The other group of participants (70.6%) thought the opposite and advocated for a specialized performance appraisal method for software testers. Some participants (44.4%) of later group precisely described what criteria should be considered for performance appraisal of software testers. Some of the criteria included comparison against testing specific KPIs, delivered system performance, efficiency of designed tests and so on. Table 1 lists the proposed criteria along with respective frequency of occurrences.

6.3. Feedback on Proposed PAF

Unfortunately not all of the participants who responded to the first section of the survey participated in the second section. We noticed a 25% and 20% drop out for the initial survey and the new lightweight survey, respectively. As a result we obtained a total of only 14 participants who completed feedback on the proposed PAF.

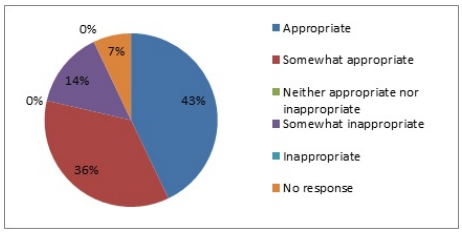

Figure 11: Responses on "Do you think the form is appropriate for performance appraisal of software testers?"

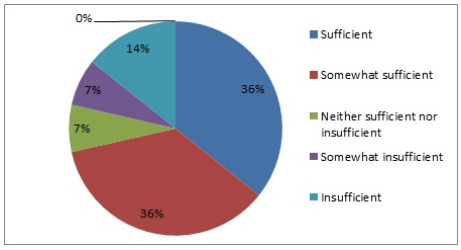

Figure 12: Responses on "Do you think the dimensions considered in this form are sufficient to assess the performance of a software tester?"

Responses to the first three questions from the initial and the new lightweight survey are analysed together. However, possible responses to the other questions in this section were different, so those are presented separately.

The main feedback on our new proposed PAF was whether participants considered the proposed form is appropriate for performance appraisal of software testers. The responses to this are shown in Figure 11. The majority of participants thought the proposed PAF was at least somewhat appropriate. Three (21.4%) participants gave their comments on the appropriateness of the PAF in the accompanying open ended question. According to the responses, the form is considered appropriate for general performance appraisal of software testers. In order to increase appropriateness, participants also suggested considering the number of bugs that pass through testing without noticing instead the number and severity of found bugs, including some evaluation about automated testing, being more specific about what a satisfactory test plan is and ignoring the number of bugs found.

The responses to whether the dimensions considered in the proposed PAF are sufficient or not, are shown in Figure 12. The participants suggested to add more dimension on setting own goals in terms of capabilities and weaknesses, taking the innovation skill of the process into account, considering the time taken to reproduce a bug often reported by customers and requested to be fixed urgently and ability to work in a team.

We requested participants to indicate whether they thought the proposed interpretation of the overall score is appropriate. As shown in Figure 13 none of the participants considered the proposed interpretation inappropriate.

We asked participants to indicate whether they clearly understood the performance labels attached to each performance dimension from respective definitions. In the new light weight survey they could indicate their overall

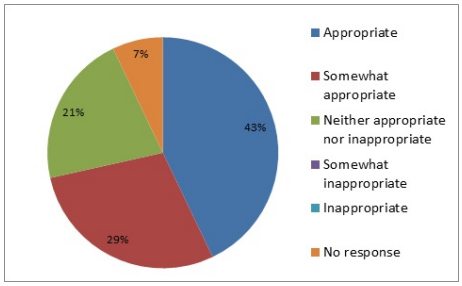

Figure 13: Responses on "Do you think the interpretation of overall score is appropriate?"

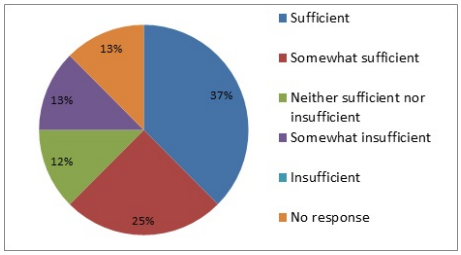

Figure 14: Responses on "Do you think the personal attributes considered in dimension7 are sufficient?"

response to this. However, in the initial detailed survey they could indicate the ease of understanding for each dimension.

We asked participants to indicate whether they clearly understood the performance labels attached to each performance dimension from respective definitions. From the responses we see that the majority of participants could understand the labels from definitions well (25% indicated the labels were "very clear", 50% indicated the labels were "clear"), although a few (12% indicated those were "neither clear nor unclear") were not sure about this. However, two participants thought the definitions were unclear for dimension 3 and 4.

We asked participants to comment on the relative weight assignment to three of the performance dimensions in the proposed PAF. The majority of the participants were happy with the weight assignments with one participant indicating the severity of bugs found should not limit the number of bugs found.

We asked participants to indicate whether they thought that the personal attributes we considered in dimension 7 were sufficient with possible response options "yes" and "no" in the initial survey and five point Likert scale responses in light-weight survey. In the initial survey 50% participants thought the attributes were sufficient. The responses to this obtained in the new light weight survey are presented in Figure 14. Participants who thought the attributes considered were insufficient, suggested several additional attributes (in order of frequency of occurrence): ability to cooperate in a team, ability to deal with clients and colleagues, patience, ability to raise important issues to management at the right time, self organization, presentation skill, and being pedantic.

More detailed results are available at:

http://www.testingsurveys.org/PAF_results/partialResults.pdf

7. Threats to validity

One of the threats that can limit the external validity of our reported results is over-generalization of the findings. We believe that the nature of participation required for this study put off many potential participants and as such we obtained a limited number of participants. Due to the small number of participants we had to modify the survey to require less time and yet still the participation rate was not satisfactory. We found that many participants were less interested in the second section and as a result we noticed a number of drop outs. Although some of our findings are interesting, due to the low number of responses we cannot strongly conclude that our finding applies in general. Another limitation of this study was modifying our proposed PAF based on the feedback we received from the participants. Compared to the first section of the survey, the responses to this section were not as extensive as we would have preferred. This limits the strength any conclusions we draw about our proposed PAF. However, useful feedback was obtained on various aspects which have helped us to refine it further.

Misinterpretation of survey questionnaire can be a threat to the internal validity of our research. However, the broad and informative responses to the open questions indicate that participants understood the questions.

8. Discussion

From the responses to the survey questionnaire of the state of practice of performance appraisal, it is evident that a formal process of employee appraisal is practiced in the majority of organizations. A specialized appraisal process is also common for software testers. However, when there is no such specialized process, performance of software testers is most commonly appraised using a more general employee appraisal process or a manager's evaluation. In only a small number of organizations are bugs reported after deployment also considered a measure of the performance of software testers. It is the software testers responsibility to ensure certain level of reliability of the software. Bugs encountered in a live environment is an ultimate measure of the reliability of the software. We believe that this might be the reason that this criterion is used in small instances.

In response to the request to propose a performance appraisal method for software testers, participants gave detailed responses. The majority of participants advocated for specialized performance appraisal for software testers. Participants indicated different criteria for testers on which performance can be measured. Among those, counting the number of bugs encountered after delivery and the number of test cases run during unit testing were most popular. Participants also stated that KPIs should be agreed upon from the beginning and that evaluation should occur more frequently. The number of bugs found was also considered important by a few participants. As has already been discussed, this contradicts Kaner's [5] views on bug counts and our finding in the preliminary survey [2]. In the preliminary survey we also found that the severity of bugs found was a better measure than number of bugs found. However, number of bugs found was stated as a performance criterion far more frequently than bug severity. We are slightly puzzled by this result and are unclear as to whether it is simply lack of awareness of the potential problems with bug counts, or whether their practical experience suggest that the problems are less severe in practice than Kaner's [5] critque suggests.

We found that our proposed PAF was considered appropriate by the majority of the participants. The performance dimensions in the proposed PAF were also considered generally sufficient. Participants suggested that we consider some extra dimensions, such as time taken to reproduce a bug and agreed upon KPIs. The responses indicate the majority of participants were satisfied with the interpretation of the overall score, the labels attached to each dimension and relative weight assignment. We refined our proposed PAF according to the overall feedback as summarised in Table 2. The modified version of the proposed PAF is available at:

http://www.testingsurveys.org/PAF_static/refinedPaf.html

The feedback on industrial practice and the respondent's general thoughts on tester performance appraisal did produce a number of interesting ideas; however, in our view, they do not immediately suggest a clearly superior alternative to the approach taken in the PAF for our research purposes. While we would be very hesitant about using bugs *detected* in the testing process as a metric for our own purposes, the support for the idea of using bugs detected after delivery as a performance assessment tool is interesting. This may be a useful additional quantitative approach that could be applied in a research context to identify high-performing testers. It would be extremely difficult, however, to control for the many factors outside a tester's control

Table 2: Feedback on proposed PAF and corresponding modifications

Feedback	Modification to be carried out	Comment
PAF should be designed based on business domain	No modification	The aim of the proposed PAF is to develop and refine a generalized performance appraisal form for software testers. Employers can deploy the PAF and add domain specific dimension(s) if necessary.
Consider number of bugs that pass testing without notice	No modification	Undiscovered bugs may indicate poor performance or especially hard-to-find bugs. However, testing cannot reveal all bugs. The importance of this criteria needs to be evaluated before considering this as a performance dimension.
Specific definition for satisfactory test plan	Added more text to dimension 5.	Dimension 5 - test planning: added text to give explanation. Modified text is-"DIMENSION 5- TEST PLANNING: Frequency of preparing efficient and good quality test plan. Quality and efficiency attributes of the test plan include: ability of assessing high risk area and selection of efficient test strategy. A satisfactory test plan should incorporate such test strategy that is able to test most important parts of the software in feasible time."
Ignore number of bugs	No modification.	We agree that number of bugs should not be directly used as a measure of testing performance. As such number of bugs are considered as related to severity of bugs and the difficulty of finding those.
Specific KPIs from testers themselves	Added a dimension.	A dimension is added where the manager can set some specific KPIs with the software tester at the beginning of the evaluation period. The score of this dimension and relative weight assignment (if necessary) is suggested done by the manager.
To consider innovation skill of process	No modification.	We believe the selection of appropriate and feasible test strategy covers the innovativeness of selected test process.
Time taken to reproduce a bug	No modification.	Suggested by one participant. In some organizations software testers may be requested to reproduce bugs that are reported by clients but not all, hence we didn't include directly
Unclear definition for "bug count vs ease of finding"	Simplified text	
Unclear definition for "bug count vs severity"	Simplified text	
Add more personal attributes	Added attribute.	"Team playing capability" was mentioned by multiple participants. Others, such as "ability to deal with clients and colleagues", "patience", "ability to raise important issues to management at the right time", "self organization", "presentation skill" and "being pedantic" were listed only once. If managers think some are particularly important, those can be added as specific KPIs.

which might affect this metric.

A number of issues would have to be considered by an organisation or test team manager in adopting our PAF. This includes issues raised previously: would they need to adapt some questions or add new ones for their specific needs e.g. to assess specific testing tasks? They would need to adapt their appraisal processes to incorporate use of the PAF, KPI setting, and provide targeted support and training to improve tester performance over time. However, this is necessary when using any appraisal process. Ultimately, management (and testers) would need to identify suitable improvements in tester performance over time by user of the PAF to justify its adoption.

9. Conclusion

This study aimed to obtain information about the state of practice of performance appraisal of software testers and to make suggestions on how the appraisal process can best be conducted. We found that there are two

trends: some organizations use the same performance appraisal process for all employees, whereas some use a specialized one for software testers. Our participants suggested a number of criteria that should be considered in appraising performance of software testers. Among those number of bugs found in live environment and efficiency of running test cases were most prominent.

We also aimed to obtain feedback on our new proposed PAF for software testers and refine this based on their feedback. In spite of some dropouts in participation, we obtained good feedback on our proposed PAF. We found that parts of the proposed PAF were unclear to our participants and so we modified the text to make those parts more understandable. A few more useful performance dimensions were proposed by our participants. We added two more dimensions and one more personal attribute to address these proposals on our PAF. We believe the new refined PAF can appraise the performance of software testers appropriately. However, further industry deployment and evaluation of the PAF by managers is required to verify this. The proposed PAF will be helpful for the researchers aiming to investigate the influence of different factors on the performance of software testers as well as for industry practitioners to assess the performance of software testers for improvement, promotion and renumeration purposes.

References

[1] T. Kanij, R. Merkel, J. Grundy, An empirical study of the effects of personality on software testing, in: Proceedings of the 2013 IEEE International Conference on Software Engineering Education and Training, CSEET2013, 2013.

[2] T. Kanij, R. Merkel, J. Grundy, Performance assessment metrics for software testers, in: Proceedings of the 2012 ICSE Workshop on Human Factors in Software Engineering, CHASE2012, 2012.

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

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

[5] C. Kaner, Don't use bug counts to measure testers, Software Testing & Quality Engineering (1999) 80.

[6] C. Kaner, Measuring the effectiveness of software testers, Software Testing Analysis & Review Conference (STAR) East.

[7] B. L. Killingsworth, M. B. Hayden, D. Crawford, R. Schellenberger, A model for motivating and measuring quality performance in information systems staff, Information Systems Management 18 (2) (2001) 1–7.

[8] D. B. Mayer, A. W. Stalnaker, Selection and evaluation of computer personnel- the research history of sig/cpr, in: Proceedings of the 1968 23rd ACM national conference, ACM '68, ACM, New York, NY, USA, 1968, pp. 657–670.

[9] R. A. Dickmann, A programmer appraisal instrument, in: Proceedings of the second SIGCPR conference on Computer personnel research, SIGCPR '64, ACM, New York, NY, USA, 1964, pp. 45–64.

[10] E. F. Bairdain, Research studies of programmers and programming.

[11] R. M. Berger, R. C. Wilson, Correlates of programmer proficiency, in: Proceedings of the fourth SIGCPR conference on Computer personnel research, SIGCPR '66, ACM, New York, NY, USA, 1966, pp. 83–95.

[12] B. Powell, Performance evaluation of programmers and analysts, in: Proceedings of the 3rd annual ACM SIGUCCS conference on User services, SIGUCCS '75, ACM, New York, NY, USA, 1975, pp. 19–21.

[13] D. Grote, The Complete Guide to Performance Appraisal, Amacom Books, 1996.
URL `http://books.google.com.au/books?id=u1n3tgAACAAJ`

[14] M. L. Hutcheson, Software Testing Fundamentals: Methods and Metrics, 1st Edition, John Wiley & Sons, Inc., New York, NY, USA, 2003.

[15] Recruitment website, http://www.monster.com/.

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

[17] J. Iivonen, M. V. Mäntylä, 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, ESEM '10, ACM, New York, NY, USA, 2010, pp. 60:1–60:1.

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

[19] M. Denscombe, The Good Research Guide for Small-Scale Social Research Projects, Open University Press, Milton Keynes, UK, 2003.

[20] Proceedings of the 5th International Symposium on Empirical Software Engineering and Measurement, ESEM 2011, Banff, AB, Canada, September 22-23, 2011, IEEE, 2011.

[21] L. F. Sanz, Personal skills for computing professionals., IEEE Computer (10) 110–112.