Proposing a Comprehensive Software Metrics for Process Efficiency

Dr. Latika Kharb

Abstract— For an effective test measurement, a software tester requires a testing metrics that could measure the quality and productivity of software development process along with increasing its reusability, correctness and maintainability. In order to ensure that we produce better quality code; what we need is development of efficient testing measurement techniques that could assist in the creation of high quality software within limited time and resources along with user satisfaction. Efficient test process measurement is essential for managing and evaluating efficiency of final software product. In addition, there is increasing demand for enhancement of operational capabilities of a software product so as to be able to get more and more output with a reduced investment plan. Effective usage of our proposed metrics can serve as an important landmark for identifying efficiency and effectiveness of a software testing process and then on the basis of its analysis, it would help us to identify the areas of improvement and guide us to devise subsequent plans. Keeping this demand in mind, in this paper, we've proposed three metrics for software process quality testing; which could be used as one of the approaches for further guidance in software process quality testing. The intention behind our proposed set of metrics is to inculcate a complete improvement in overall software testing life cycle.

Index Terms— Efficiency, Effectiveness, Maintainability, Process, Productivity, Quality, Reusability, Test Measurement

1 INTRODUCTION

OVER the recent years, the enhanced discipline of software quality engineering has got the flavor of a scientific discipline with introduction of various aspects of testing metrics. Delivering a high quality reliable product is the main focus in any software development. The basic quality measure is the defects in the product. Defects found in the later phases of the product development are mainly because of faulty design and code and poor reviewing capability [1]. Software metrics serve valuable information to enable important decisions over the entire software development lifecycle and also helps in effective risk assessment and reduction of errors. Software quality based testing metrics are developer-oriented and developers could use them to estimate quality at a very early stage in the software development process [2].

Over time, due to increased product functionalities, software projects have become more and more complex and along with increasing work completion pressures, the software projects are required to be accomplished in lesser amount of time but with fewer people. So, the increasing complexity included with time and budget constraints, has declined the product quality. Suppose if we are able to measure something, it can be analyzed it in a better way as soon as we know more about it, and finally fetch a way to improvise it.

Metric is the cornerstone in assessment and foundation for any business improvement. It is a Measurement Based Technique which is applied to processes, products and services to supply engineering and management information and working on the information supplied to improve processes, products and services, if required [3].

Metrics help gauging not only the progress and quality but also highlights future of a software testing effort. Metrics can also be leveraged to evaluate past performance, present status and foresee future trends. Effective and efficient metrics are not only simple and objective but are also measureable, meaningful and have easily accessible underlying data. A promising software metrics suite is one that encompasses relevant, effective and efficient and simpler metrics that covers different aspects of software development; thereby enabling managers in making judgments during the software life-cycle. This paper illustrates the importance of metrics in the whole software testing process and provides an insight in to some new proposed process metrics contributing towards effective risk assessment and reduction of errors through their relevancy, effectiveness and efficiency.

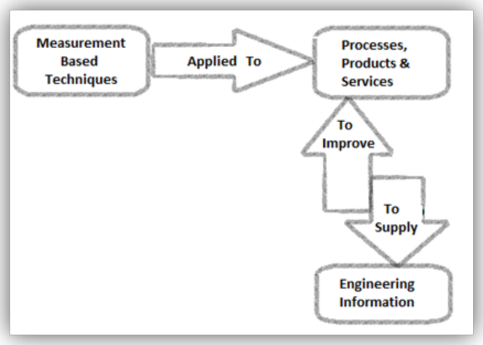

Fig. 1. *Software Metrics*.

The benefits of having good metrics:
- Helps to predict the future scope for an organization
- Provides a basis for estimation for closure of the performance gap.
- Identifies risk areas that require more testing.
- Quickly identifies and helps to resolve potential problems

• *Dr. Latika Kharb is currently working as Associate Professor in Department of IT at JIMS, Sector-05, Rohini, New Delhi, INDIA. PH-+91-9991738097. E-mail: latika.kharb@jimsindia.org, latika.kharb@gmail.com*

and identifies areas of improvement.

• Test metrics provide an objective measure of the effectiveness and efficiency of testing.

2 Various Dimensions of Test Metrics

The core of software test engineering is to examine the relationships among various in-process metrics, project characteristics and product quality. Further, based on the findings, it aims to instill improvements in both the testing process and product quality. Developers could save both time and money and get a better product if they make the right decision [4]. Software testing metrics are a subset of overall software metrics umbrella that focus primarily on the quality facets of process and the product. Software testing metrics can be further classified into Process Quality Metrics and Product Quality Metrics. In this research paper, our stress will be on Process Quality Metrics only. In this paper, we discuss three metrics falling under Process quality metrics and for each metric; we've discussed its purpose with the method of interpretation.

3 Proposed Metrics for Process Efficiency

Organizations today are striving to deliver more and more with less. Achieving this objective is highly dependent upon, how efficient the various processes running in these organizations are.
Software development activities require more time for best software quality [5]. Software developers are the one who deal with practical implementation of data and so they have to check each and every metric very minutely as even 1% error rate could be critical if it belongs to real life software development *viz.* aeronautical systems [6]. Until now, the understanding of measuring software quality is not yet sophisticated enough and is still far away from being standardized and in order to assess the software quality, an appropriate set of software metrics needs to be identified that could express these quality attributes. Our proposed process efficiency metrics depict the ability of a process to produce desired outcome with optimum utilization of resources. As software testing is the key process behind a high-quality deliverable / end product; it has become the prime focus for efficiency improvement efforts. And for this, metrics serve as a useful guide to ensure that all the improvement efforts are applied in the right direction and enables one to measure various efficiency aspects of software testing process.

In this section, a set of three process quality metrics have been proposed that could help the developers and/or researchers to measure a process from various dimensions and the main focus is to make improvement while development of a quality outcome within least time and cost and that is possible only if the process itself is effective as well as efficient in working. As a single metric could only measure one aspect of software test quality; therefore, it must not be applied or used in isolation. So, we recommended metrics to measure test process that may help to achieve the organizational goal of quality.

3.1. Test Progress over Time Metrics (TPOTM)

The main purpose behind our proposed Test Progress over Time Metrics is to track the progress of testing. The proposed metric depicts planned testing progress over time in terms of number of test cases completed successfully compared with the number of test cases attempted.

$$TPOTM = \frac{TTCA-TTCC}{TTCA} * 100 \qquad (1)$$

where
TPOTM is the Test Progress over Time Metrics
TTCA is the total test cases attempted
TTCC is the total test cases completed

The objective of this metric is to track and compare testing progress with the scheduled plan, hence enabling teams to take early actions upon any indications that the testing activity is lacking behind. Testing activity generally gets impacted the most under pressurizing schedules. So, with a formal testing progress metric deployed, it is better for the team to solve the problem.

3.2. Test Execution Productivity Metrics (TEPM)

In order to measure test team's efficiency is to check the productivity at which the team carries out test execution. In our metrics: Test Execution Productivity Metrics, the test execution productivity may be defined as the average no. of test cases executed by the team per unit of time (where the unit may be hour/ day/ week; but is generally taken as an hour or a day). The objective of this metric is to help identify the problematic areas that have negative impact on team's productivity and thus guide the tester to take remedial actions, where ever feasible.

$$TEP_t = \frac{TEP_{t1}\ TEP_{t2} \text{----} TEP_{tn}}{2} \qquad (2)$$

where
TEP_t = denotes Test Execution Productivity over Time

TEP_{t1} TEP_{t2} - ..... - TEP_{tn} denotes Test Execution Productivity over Time i.e from time t_1 till time t_n

3.3. Defect Response Time Metrics (DRTM)

One of the key activities carried out by test teams is verification of defects, which turns back to them after getting fixed by development team. This activity generally needs close attention by test teams especially for defects having higher priorities. As an indicator of test team's operational efficiency, the proposed Defect Response Time Metrics depicts the average time taken by the team for verification of these defects. The proposed metric depicts the average verification/ turnaround time taken by the team to verify defects of various levels of priorities.

$$DRTM=DRT+TAT \tag{3}$$

Where

DRTM is the Defect Response Time Metrics
DRT is the Defect Removal Time.
TAT is the Turnaround Time taken.

4 Effectiveness of Proposed Metrics

Educating project managers, test managers, and development managers as to what we are measuring, as well as what those numbers mean is very important. This should be done for two reasons. The first is to ensure that managers support and understand the value of the metrics. It is vital that they are interested in these metrics as much as we are in providing them. The second reason is to educate them on what they can do to affect each metric positively. This last reason is the most important, yet is also the most difficult to explain. Defects themselves pose an interesting problem when it comes to classification.

Software metrics can provide useful information to project managers and software developers by providing means of measuring the complexity of a software product [7]. Researchers have put much effort into learning how to use metrics for software process improvement and there have been many discussions in current years about the role of software metrics in helping software organizations to improve productivity and software quality. The essential step is establishing test metrics is to identify the key software testing processes that can be objectively measured [8].

Test metrics are key “facts” that project managers can use [9]:

- To understand their current position
- To prioritize their activities to reduce the risk of schedule over-runs on software releases.

An effective measurement activity should be able to evaluate the current process and provide guidelines to the manager for future improvement. The metrics we proposed in our paper could provide information that is helpful for improvising the current test process. We expect by gathering real data and analyzing it, some useful results can be produced by using these metrics [10].

5 Conclusion

Metrics is a measurement based technique which is applied to processes, products and services to improve processes, products and services, if required. It indicates level of customer satisfaction, whenever required and act as monitor when the process is going out-of-control. We can use testing to determine whether a program component meets its requirements. To accomplish its primary goal (finding errors) or any of its secondary purposes (meeting requirements), software testing must be applied in a systematic fashion. Testing involves operation of a system or application under controlled conditions and evaluating the results. By using our proposed process testing metrics in a consistent manner, software developers will see improvement in the software process development. However, no single metric works during all of the development phases; therefore, using several metrics for one system helps to have a handy solution that can be used during different aspects of the software process lifecycle. The metrics covered in this paper are the following: *test progress over time metrics (TPOTM), test execution productivity metrics (TEPM), and defect response time metrics (DRTM).* This paper presents a study and methods of implementation of different software process testing metrics. When used properly, i.e., when a company uses the best software testing metric during each development phase, the quality of the software will dramatically increase. Therefore, we highly recommend using our proposed software process testing metrics for the software quality assessment, process efficiency and effectiveness.

References

[1] Kiran Kumar Marri, “Models for evaluating review effectiveness”, 3rd Annual International Software Testing Conference, 2001.
[2] Latika Kharb et al, “Reliable Software Development with Proposed Quality Oriented Software Testing Metrics”, Int. J. Comp. Tech. Appl.(IJCTA), July-August 2011, Vol 2 (4), pg. 798-802.
[3] Premal B. Nirpal et al.”A Brief Overview Of Software Testing Metrics”, International Journal on Computer Science and Engineering (IJCSE), Jan 2011, Vol. 3 No. 1.
[4] Latika Kharb et al, “Software Testing vs. Code Inspection: What else for Verification &Validation?” Proceedings of the 4th National Conference; INDIACom-2010 Computing For Nation Development, February 25 – 26, 2010 Bharati Vidyapeeth’s Institute of Computer Applications and Management, New Delhi.
[5] Pradeep Tomar et al, “New Algorithm for Component Selection to Develop Component-Based Software with X Model”, Lecture Notes on Software Engineering, August 2013, Vol. 1, No. 3.
[6] Latika Kharb et al, “Complexity Metrics for Component-Oriented Software Systems”, ACM SIGSOFT Software Engineering Notes, March 2008, Volume 33 Number 2.
[7] Latika Kharb et al, “Software Component Complexity Measurement through Proposed Integration Metrics”, Journal of Global Research in Computer Science, June 2011, Volume 2, No. 6.
[8] Anitha.A, “A Brief Overview of Software Testing Techniques and Metrics”, International Journal of Advanced Research in Computer and Communication Engineering, December 2013,Vol. 2, Issue 12.
[9] Dr. Arvinder Kaur et al,” Software Testing Product Metrics – A Survey”, Proceedings of National Conference on Challenges & Opportunities in Information Technology (COIT-2007), March 23, 2007, RIMT-IET, Mandi Gobindgarh.
[10] Latika Kharb et al, “Assessment of Component Criticality with Proposed Metrics”, 2nd National Conference, INDIACom-2008: Computing for Nation Development, February 08 – 09, 2008, Bharati Vidyapeeth’s Institute of Computer Applications and Management, New Delhi.