Reliable Software Development with Proposed Quality Oriented Software Testing Metrics

Ms. Latika Kharb*, Dr. Vijay Singh Rathore**
Research Scholar*, Professor**
Department of Computer Science, NIMS University, Jaipur*
Department of Computer Science, S.K. College, Jaipur**

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. 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 research objective in this paper is to construct and define a set of easy-to-measure software metrics for testing to be used as early indicators of external measures of quality. So, we’ve emphasized on the fact that reliable software development with respect to quality could be well achieved by using our set of testing metrics, and for that we’ve given the practical results of evaluation.

1. Introduction: Software Quality Measurement through Testing
In order to ensure that the software under development has been implemented correctly, software testing is deployed as an integral part of software development in every phase of the software development cycle specifically, to understand the requirements, to produce better-quality code and for the development of efficient testing measurement techniques that could assist in the creation of high-quality software within limited time and resources. In order to achieve this, entire user satisfaction is measured in terms of quality of compliant product and its delivery within scheduled time, cost and budget, i.e.

User Satisfaction = Compliant Product + Good Quality + Delivery within Schedule

With fundamental research that addresses the challenging problems like development of methods, tools and empirical studies, we could not expect significant improvement in the way we test software [1].

“Figure 1. The Purposes of Quality”
Software testing provides visibility into the product quality as well as process quality and as the goal of software testing is to discover errors in the software for building confidence in the proper operation of the software; therefore, solving the software-testing problems is not an easier task as one could never be sure that the specifications are correct. The key to effective measurement lies in the ability to clearly identify the goals to be accomplished and the issues to be tackled [2] because an effective software measurement technique for testing is the first step to make software development as well as software engineering a true engineering discipline as it helps in the evaluation of not only the quality of the requirements document for quality indicators, identifies volatility, but also tracks testing as a way for ensuring that all requirements have been satisfied. Measurement plays a critical role in effective software development [3]. Therefore, software-testing measurements must be planned carefully because it requires significant efforts to implement, and returns are realized only over a period of time. Software metrics for testing are
important because of the benefits associated with early detection and correction of problems while testing. Although many metrics have been proposed by researchers, but most of them are either ignored or are left in isolation. Testing metrics have been one of the most sophisticated processes for use in measurement and demonstration of the correctness and quality of a program. 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. Quality of a software product is directly linked with the faults present in the software module. Quality is an indicator of high performance, whether that performance is measured in terms of individuals, teams, products, or the entire organization. ISO 8402 defines quality as the totality of characteristics of an entity that bear on its ability to satisfy stated and implied needs [4]. Despite the range of definitions, the goals underlying the pursuit of quality are the same: achieving conformity, reducing variation, eliminating waste and rework, eliminating non-value-adding activity, preventing human error, increasing efficiency and effectiveness, improving productivity, and preventing defects. Metrics are important because of the benefits associated with early detection and correction of problems with requirements [5]. Actually, many factors determine the quality of the software like,

- Clear and documented functionality of the software.
- Status of the development of software.
- Test cases cover the complete functionality. Process to determine the severity and priority of the defect.
- Collecting and analyzing the testing metrics.

Using quality management techniques, as the basis for software development in testing is a promising avenue for improvements in a resource constrained environment where challenges are ever increasing. Software testing is one of the tools used to ascertain the quality of software as software quality is achieved by testing to verify its correctness, performance and scalability according to its specification. There are several test metrics identified as part of the overall testing activity in order to track and measure the entire testing process. These test metrics are collected at each phase of the testing life cycle /SDLC and analyzed and appropriate process improvements are determined and implemented as a result of these test metrics that are constantly collected and evaluated as a parallel activity together with testing both for manual and automated testing irrespective of the type of application. The test metrics can be broadly classified into the following three categories such as: Project Related Metrics, Process Related Metrics, Customer related Metrics.

2. Metrics for Quality Testing

Test metrics are an important indicator of the effectiveness of a software testing process. In current years, there have been many discussions about the role of software metrics in helping software organizations to improve productivity and software quality. Some researchers have defined new metrics for specific measurement purposes and contexts software engineering management. Researchers have put much effort into learning how to use metrics for software process improvement (SPI) [6]. Our future work includes using real industry level data to evaluate these new metrics we recommended to measure performance of individual test phases, giving suggestions to test teams and support teams for necessary changes in the test process, and implementing the whole set of metrics in a production test environment. In this section, four new quality metrics for achieving operational excellence in software testing have been proposed namely, quality of code metric (QCM), quality of product metric (QPM), total test improvement metric (TTIM), and test effectiveness metric (TEM). 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 phase that may help to achieve the organizational goal of quality.

2.1. Quality of code Metric (QCM)

The following metric is used to measure the quality of software code and is measured by taking the average of the sum of the number of weighted defects found in a product before and after release & thousand source lines of code with the total lines of code of developed product. It is given by the equation:

$$QCM = \frac{(W_{BR} + W_{AR})/KSLOC}{LOC_{DP}}$$

Where $W_{BR}$ is the number of weighted defects found in a product under test (before release), $W_{AR}$ is the number of weighted defects found in a product under test (after release). The weight
for each defect is defined on the basis of defect severity and removal cost. A severity is assigned to each defect by experienced software testers based on how important or serious the defect is. The more important or serious the defect, it’s more expensive to remove it, and the higher the weight which is assigned to the defect. KSLOC denotes the thousand source lines of code and LOC\textsubscript{DP} is the number of total lines of code of developed product. This metric (QCM) captures the relation between the number of weighted defects and the size of the product release. The lower this number, indicating fewer defects or less serious defects found, the higher is the quality of the code delivered by the development teams.

2.2. Quality of Product Metric (QPM) 
(measured after official delivery to customer)

The following metric is used to measure the quality of software product and is measured by taking the average of the sum of the number of weighted defects found in a product before and after release with the total lines of code of developed product. The weight for defects is again defined based on defect severity and removal cost. It is given by the equation:

\[ QPM = \frac{W_{BR} + W_{AR}}{LOC_{DP}} \]

This metric (QPM) shows the relation between the number of weighted defects shipped to customers and the size of the product release. A low number here indicates fewer defects, or less serious defects, implying a higher quality of the code delivered by the development teams.

2.3. Total Test Improvement Metric (TTIM)

The following metric is used to measure the total test improvement and is measured by taking the average of the sum of number of weighted defects found in a product under test through white-box testing, black box testing and grey-box testing respectively with the total lines of code of developed product. It is given by the equation:

\[ TTIM = \frac{W_{WBT} + W_{BBT} + W_{GBT}}{LOC_{DP}} \]

Where \( W_{WBT} \) is the number of weighted defects found in a product under test through white-box testing, \( W_{BBT} \) is the number of weighted defects found in a product under test through black box testing, and \( W_{GBT} \) is the number of weighted defects found in a product under test through grey-box testing. This metric (TTIM) shows the relation between the number of weighted defects detected by the test team during testing and the size of the product release. The higher this number, indicating more defects or more important defects were detected, the higher the improvement to the quality of the product that can be attributed to the test teams.

2.4. Test Effectiveness Metric (TEM) (to drive out defects after official delivery to customer)

The following metric is used to measure the total test effectiveness and is measured by calculating the ratio of the average of the Total Test Improvement Metric with the sum of the number of weighted defects found in a product before and after release. It is given by the equation:

\[ TEM = \frac{TTIM}{(W_{BR} + W_{AR})} * 100\% \]

Where TTIM is the test Improvement Metric. This metric (TE) shows the relation between the number of weighted defects detected during testing and the total number of weighted defects in the product. The higher the number, indicating a higher ratio of defects or important defects were detected before release, the higher is the effectiveness of the test organization to drive out defects.

3. Practical Work: The Project Process

The basic idea behind these quality metrics is that it is possible to set a number of basic requirements that shall be fulfilled so that a software system becomes be well designed. By formalizing the requirements and gathering metrics for them it is possible to analyze code, and becomes possible to grade developed systems in terms of the design quality. In next section, the practical usage of software metrics will be discussed by presenting practical strategies. In this research work, there were three major phases in our research work:
- Data collection phase,
- Metrics calculation phase, and
- Result analysis phase.
3.1. Data collection Phase

Data for calculating test process metrics have to be collected from the test process. From our experience, a small project is useful and necessary before adopting any metric into practice. In doing theoretical work, such as defining the metric, we may not truly understand the availability of data. Even when we have tried to make data collection relatively easy, some minor changes may still be required in the test process to get correct and accurate data. In this research work, we used two of our working projects of Hotel Management and Payroll System in languages of Visual Basic and Oracle. Based on the definition of the metrics, we derived a list of variables for which we needed to collect data in the data collection phase. These variables were organized into four categories as follows.

- Product size data: KSLOC, LOC_Dp
  
  Where KSLOC denotes the thousand source lines of code and LOC_Dp is the number of total lines of code of developed product.

- Weighted defects data: W_BR, W_AR, W_WBT, W_BBT, WGBT

3.2. Metrics calculation Phase

In the data collection phase, we’ve collected information for all the variables for whom we need to calculate the metrics. The metrics calculation phase is straightforward: by applying the value of each variable to the metrics, we can calculate the results easily. After all the metrics have been calculated, table similar to the ones given in next section could be created.

We’ve taken values of W_BR, W_AR, KSLOC and LOC_Dp and placed the respective values in the equation of QCM, QPM, TTIM and TEM metrics. Placing values in metrics equations, we’ve the results:

1. QCM for Project 1
   
   \[ QCM = \frac{(220+4)}{8000} = 0.0025 \]

2. QCM for Project 2
   
   \[ QCM = \frac{(150+5)}{10000} = 0.0015 \]

3. TTIM for Project 1
   
   \[ TTIM = \frac{(100+120+4)}{8000} = 0.0074 \]

4. TTIM for Project 2
   
   \[ TTIM = \frac{(100+50+5)}{10000} = 0.0051 \]

4. Effectiveness of the Metrics

Researchers have put much effort into learning how to use metrics for software process improvement (SPI) 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 research spans a wide variety of topics. Some researchers have defined new metrics for specific measurement purposes and contexts software engineering management and metrics are not purely technical disciplines. 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. An effective measurement activity should be able to evaluate the current process and provide suggestion to the manager for future improvement. The metrics we used in our project could be able to provide information that is helpful for justifying the current test process. The metric results clearly show the improvement that the test teams had made in the test process in terms of quality. In section 4.3, we discussed some of the results found in our two projects and our work includes real data for the evaluation of proposed metrics and implementing the whole set of metrics in a project environment.

5. Conclusion
The goal of the testing activity is to find as many errors as possible before the user of the software finds them. 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 software testing metrics in a consistent manner, software developers will see improvement in the software and on the use of the metrics. 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 process of software development. The metrics covered in this paper are the following: quality of code metric (QCM), quality of product metric (QPM), total test improvement metric (TTIM), and test effectiveness metric (TEM). This paper presents a study and implementation of different software metrics. We apply these metrics to our sample projects, and evaluated the results. We find that there are specific metrics for different software quality assessments. 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 software-testing metrics for the software quality assessment.

6. References