Journal of Engineering & Technological Advances ISSN: 2550-1437
Vol. 1, No 1, 2016

ENHANCING SOFTWARE RELIABILITY WITH THE EFFECTIVENESS OF SOFTWARE TESTING TECHNIQUES

Leelavathi Rajamanickam[1*], Kate Lam Woon Yee[1] ,Chen Woon Choong[1]
[1]School of Information Technology, SEGi University, Malaysia
[*]Email: leelavathiraj@segi.edu.my

ABSTRACT

There are a great number of software testing techniques at our disposal for testing a software product and although the utilization of these techniques is growing, we do not have an adequate knowledge of their relative quantitative and qualitative statistics. The choice of a software testing technique in the evaluation of software influences both the process of evaluation and the product quality. So it is imperative for us to find a testing technique which is effective as well as efficient. However, it is not sufficient if testing techniques are only compared on the basis of their fault detecting ability. They should also be evaluated to discover which among them enhances reliability the most. As such, to establish a useful theory for testing, we need to evaluate existing and novel testing techniques not only for effectiveness and efficiency but also for their ability for enhancing software reliability.

Keywords: *Software Testing Techniques; Software Testing; Effectiveness; Software Reliability.*

1.0 INTRODUCTION

According to Leelavathi, R. (2014) studies on software testing have described the process of testing as a feature with varying test data to obtain a result and then compares the actual result with the expected result. It is not merely a tool to find defects or bugs in the software, instead it is a completely dedicated discipline of evaluating the quality of the software. Software testing constitutes a major part of the software development lifecycle. As such, a lack of testing has resulted in many software related problems in the past, and have actually brought about social problems and financial losses. Despite all the efforts put in to ensure the quality of the software, the effectiveness of testing still has not met with the general expectations. One is often faced with a lack of time and resources, which can limit one's ability to effectively complete testing efforts, thus ruling out exhaustive testing. Therefore, it is imperative to select proper and effective testing techniques which will increase test effectiveness to a maximum extent, yet efficiently keeping in view the limited resources available for testing. It is difficult to select a testing technique as there is inadequate information about the relative effectiveness and cost of testing techniques. To obtain such information, it is not easy, given the variability of their operation, which depends on the subject that applies it, the programming language, type of software being tested, the type of faults etc. In any case, some advances have been made in evaluating the effectiveness of the testing techniques. Additional effort is required to further evaluate and compare software testing techniques in a way that would yield fruitful results. However, one important thing to be noted is that most experiments conducted thus far focused only on evaluating software testing technique's fault finding effectiveness and efficiency. Little attention has been given on evaluating software testing techniques in terms of software reliability improvement criterion as producing dependable software is also one of the main objectives of software testing. Therefore, there is an urgent need to evaluate software testing techniques not only for effectiveness and efficiency of finding faults but also for their ability in enhancing software reliability.

2.0 SOFTWARE TESTING TECHNIQUES

Quadri, S. M. K., *et. al.* (2010) has stated that software testing is a process of verifying and validating a software application or program to see if the software meets the business and technical requirements that guide its design and development and works as expected and also identifies important errors or flaws categorized as per the severity level in the application that must be fixed. Software is tested by selecting appropriate testing technique and applying them systematically. Software testing techniques refer to the different methods of testing a particular feature of a computer program, system or product. Farooq, S. U., *et. al.* (2010) states that one needs to ensure that the selected technique(s) help to assure the most efficient and effective testing of the system. Test techniques should be able to find the greatest possible number of errors with manageable amount of efforts applied over a realistic time span with a finite number of test cases. Some techniques are easy while others require a little experience to really be employed effectively. However, the fundamental question is which technique(s) that should be adopted for effective testing.

3.0 SOFTWARE TESTING AND RELIABILITY

Software testing and software reliability have traditionally belonged to two separate communities. However, at present there is a strong bond between software testing and software reliability. An important aspect of testing is to make quality and its characteristics visible which would also include the reliability of the software. The reliability attribute is not directly measurable and must therefore be derived from other measurements such as data failure collected during the testing. Software testing is an effective method for estimating the present reliability and predicting future reliability and also ways to improve it. The difficulty of the reliability attribute is that it only has meaning if it is related to a specific user of the system. Different users experience different reliability issues, because they use the system in different ways. If one is to estimate, predict or certify reliability, it has to be related this to the usage of the system. Wohlin, C., *et. al.* (2001) states that one way of relating the reliability to the usage is to apply usage-based testing. On the other hand, Dalal, S., *et. al.* (1988) states that software reliability can be used to measure how much progress has been made in system-level testing. According to Naik, K., *et. al.* (2008) studied that the size of a maintenance work can be determined by the amount of system reliability that can be sacrificed for a while. The concept of reliability also allows us to quantify the failure-related quality aspect of a software system. Quantification of the quality aspect of software systems gives developers and managers a better insight into the process of software development. In future, it is important to bring these two groups closer, so that on one hand, software testing can be effectively conducted, while at the same time, software reliability can be improved.

4.0 SOFTWARE TESTING VS. RELIABILITY TESTING

Software testing is a generic term which includes attempts to verify and improve all aspects of software quality. Software testing techniques serve multiple purposes in the life cycle of software testing. Figure 1 shows the test information flow reflecting the different purposes of software testing. Correctness is the minimum requirement of a software; therefore, determination of correctness of the final program with respect to its requirements is the essential purpose of testing. The most common approaches of testing focuses on finding faults as much as possible. Thus, the main objective is fault detection i.e. correctness testing. Lyu, M. R., (2007) found that software testers test software without referring to how the software will operate in the field, as often the environment cannot be fully represented in the laboratory. As a result, they spend more time trying to break the software rather than conducting normal operations and mostly design test cases for exceptional and boundary conditions only. Fault detection does not necessarily inspire confidence. Farooq, S. U., et,. al. (2010) states that ultimately one important goal of software testing should be to measure the dependability of the tested software and also to increase it. If failures are more important than faults, the goal pursued during the test phase may also change. Alternatively, another main objective of testing could be to increase the confidence in failure-free operations of the software, i.e. reliability testing. In such a case, the pursuit is not about the discovery of as many faults as possible but it is about striving for higher reliability. In order to obtain confidence in the daily operations of a software system, the exact situation needs to be replicated. The software should be tested according to its operational profile (representing the typical usage scenarios) in order to allow accurate reliability estimation and prediction. In reliability testing, one is not so much interested in the faults themselves, but rather in their manifestations. A fault which frequently shows itself will in general cause more damage than a fault which seldom shows up. Reliability testing will naturally tend to uncover early those failures that are most likely to take place in actual operations, thus directing efforts at fixing the most important faults. So the aim should be to focus the testing efforts on both types of testing so as to develop a software which will be effective as well as being reliable. The fault-finding effectiveness of a correctness testing method hinges on whether the tester's assumptions about faults represent reality; for reliability testing to deliver on its promise of better use of resources, it is necessary for the testing profile to be truly representative of the operational use.

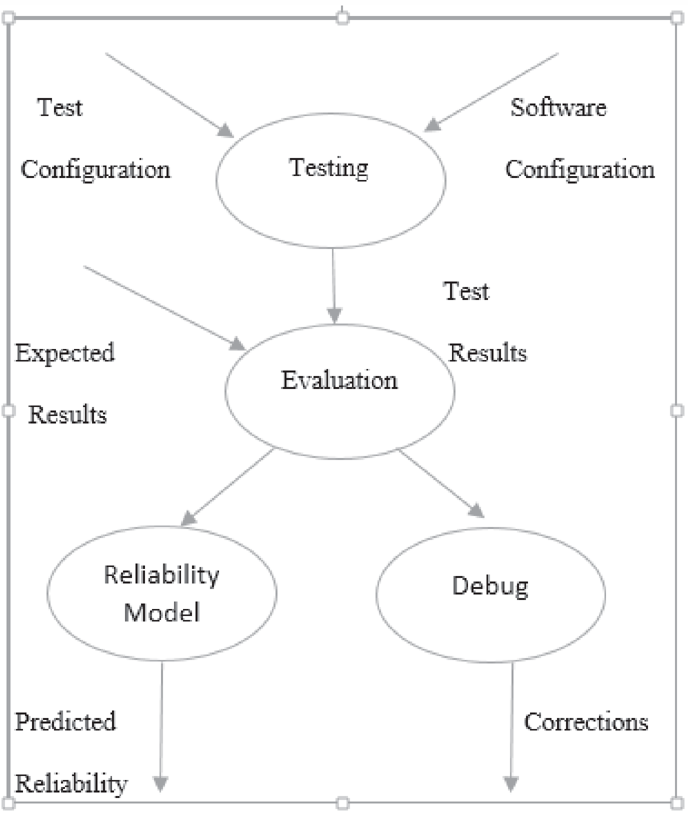

Figure 1. Test Information Flow

5.0 RELIABILITY

Most previous evaluations have considered all failures to be equivalent to one another; in reality, however, this is not the case. Fault finding ability is useful for evaluating testing techniques when the goal of testing is to gain confidence so that the program is free from faults. A software testing method that can find more faults than other testing methods need not be able to have a higher reliability at the same time can get higher reliability. Testing to find faults may be more effective (provided the intuitions that drive it are realistic), but if it uncovers failures that appear very insignificant during actual operations, test efforts will end up in insignificantly improving the software which in turn means wastage of test efforts and resources. If the goal of testing is to improve the reliability of the program, then the measure of test effectiveness must be able to distinguish between those faults that are likely to cause failures and those that are unlikely to do so. As testing techniques will be used for this purpose also, it will be interesting to evaluate effectiveness of different testing techniques for reliability enhancement criterion. In other words, testing techniques should be checked for their effectiveness in exploring failure types, and thus yielding improvement in reliability. Software reliability can be used as a criterion for evaluating techniques in terms of its effectiveness to produce reliable software. For example, consider two testing techniques T1 and T2 which are applied on a software S separately. Then by monitoring the reliability level of the software, we can observe which technique is more effective in producing software systems of higher reliability. We would like to be able to compare testing strategies in a way that allows us to say that if a system has been tested using technique T1, it is likely to have less risk (more reliable) associated with its use than if it has been tested using technique T2. Many initiatives have been taken in the past to compare testing techniques. Delivered reliability describes the idea that for a given program, requirement, and operational distribution, test cases produced according to a particular testing technique may or may not detect various faults and that correction of those faults that are detected by that testing technique will increase the reliability of the program by an amount governed by the operational distribution. Since different test cases produced according to the particular testing technique will detect different sets of faults, different improvements in reliability can be achieved by applying different testing techniques. Frankl, P. G., et. al. (2000) found a probabilistic notion of delivered reliability while Vegas, S., (2004) found that some testing techniques are never considered for use at all and others are used over again in different software projects. However, this is done without even examining, after use, whether or not they were really suited used to account for this, instead of basing the comparison on the probability of finding one or more faults or failures, or the number of faults detected, their goal is to base their assessment of a testing criterion on the reliability of the program under test after it has been tested using a given strategy. Although the intuition on which this assessment is based on is interesting, there is still a long way to go ahead, as a lot of work is needed in this direction.

6.0 RELATIVE EFFECTIVENESS

Two types of studies can be carried out to evaluate the relative effectiveness of software testing techniques and provide information for selecting one among them; *analytical* or *empirical*. An analytical solution would describe conditions under which one technique is guaranteed to be more effective than another, or describe in statistical terms the relative effectiveness of software testing techniques. On the other hand, analytical studies can produce more generalized results, i.e., results which are not tied to a particular experimental perspective. However, analytical studies remain so far quite theoretical in nature. They are highly useful to expand our knowledge behind testing techniques but provide little practical guidance in selecting an effective test technique. Bertolino, A. (2004) observed that the conclusions provided by such analytical comparisons are based on assumptions that are far beyond what one can reasonably expect to know or even hypothesize about a program under test. Whereas, empirical solutions are closer to a practitioner's mindset in which measures are obtained from observed experimental outcomes. Young, M., (2008) obtained an empirical solution based on extensive studies of the effectiveness of different testing techniques in industrial practice, including controlled studies to determine whether the relative effectiveness of different testing methods depends on the software type, subjects who test it, the type of faults in the software, the kind of organization in which the software is tested, and a myriad of other potential confounding factors. However, the empirical evidence available fell short of providing any clear-cut answers. Empirical approaches to measuring and comparing effectiveness of testing techniques are still at its infancy. A major problem is to determine when, and to what extent, the results of an empirical assessment can be expected to generalize beyond the particular programs and test suites used in the investigation. Easier said than done, empirical comparisons of test techniques are very difficult and expensive for a series of obvious reasons.

Empirical studies are quite challenging as several uncertainties against both the generality of the experimental results, and the over simplification of the experimental settings can be raised after an experiment is concluded. However, at present, the trend in research is towards empirical, rather than theoretical, in comparing the effectiveness of testing techniques. Directly observed results can sound more realistic and convincing than mathematical formulas built on top of theoretical assumptions. While empirical studies have, to a large extent displaced theoretical investigation of test effectiveness, in the longer run useful empirical investigation may require its own theoretical framework. Both analytical and empirical research are required to close the circle; analytical studies give us hints on which conditions make a test technique more convenient than another, and by means of empirical studies, one then can check when it is that such conditions are met for real programs. Such kind of empirical studies could also be used to make conclusions from analytical comparisons within a practical context.

7.0 COMPARISION OF SOFTWARE TESTING

Studies carried out so far do not examine the conditions of applicability of a technique at length or assess the relevant attributes for each technique. There is no study which can tell us about the relative effectiveness of testing techniques because of the difference between parameters that they have taken into consideration and the results also, do not reveal much information and instead show a lot of contradiction. In addition, comparison criterion for testing techniques is usually not well defined. Most of the studies do not take into account all the parameters necessary for comparison, as a result, one technique does not supersede other techniques on all fronts; thereby creating ambiguity in test technique selection. The big question that still remains is; which is the technique that is most effective and efficient. At present, the answer is still unclear. Presently, selecting testing techniques is neither systematically done, nor does it follow well-defined guidelines. Current studies suggest that it is just not sufficient to rely on a single method for identifying all errors or problems in a program. Perhaps the single most important thing to understand is that the best testing technique is to avoid the use of one single testing technique. Using a combination of software testing techniques will help us to ensure that diverse faults are found, resulting in more effective testing. So it is argued that more experimental efforts are required to find out testing techniques which will make software testing both effective and efficient. So there is a need to review and evaluate software testing techniques to compare their effectiveness but experiment should be carried out in such a way so that results can be realistic and with very less contradictions.

8.0 CONCLUSION

Juristo, N., et. al., (2004) who has conducted surveys on comparisons of various software testing techniques concludes that further empirical research in software testing is needed, and that much more replications have to be conducted before general results can be stated. Experimentation on a large scale needs to be carried out for a comprehensive comparative analysis and hopefully without any contradictions. In order to that, one needs to establish common and standard parameters so that there are little variations in experimentation goals. However, the actual research settings of creating reasonable comparative models have not been totally explored. There needs to be a common schema according to which experiments should be carried out. Specific dimensions are needed on the basis of which agreements can be made on which testing method is more effective than another method. Rather than evaluating testing techniques for fault finding ability only, other factors need to be taken into consideration such as fault severity, cost, efficiency etc. Techniques should also be evaluated for reliability dimension as one cannot expect one testing technique to supersede all other techniques. There is often a specific interest or purpose of evaluating a particular test technique, based on the assumption that the technique will be more effective. Regardless of the technique, it would be wise to try to understand what types of failures and faults a particular technique can be expected to find and at what cost. One has to check whether testing technique effectiveness and efficiency depends on the program to which it is applied for, the subject who applies it, the number of faults in the program or the type of faults in the program. Evaluation of software testing techniques should be done in order to acquire the basic knowledge about the relative effectiveness of software testing techniques for both fault finding and reliability criterion. Software testing technique should be validated in terms its effectiveness and efficiency for real world software, and only then, can these testing techniques be put into practice.

9.0 REFERENCES.

- Leelavathi, R., (2014). “Software Testing, Analysis and Objectives” International Journal of Advanced Trends in Computer Science and Engineering, Vol 3, No.5, Pages: 01-04.
- Quadri, S. M. K., & Farooq, S. U. (2010). “Software Testing – Goals, Principles, and Limitations”, *International Journal of Computer Applications* (0975 –8887) Volume 6– No.9.
- Farooq, S. U., & Quadri, S. M. K. (2010). Identifying some problems with selection of software testing techniques. *Oriental Journal of Computer Science & Technology* Vol. 3(2), 266-269.
- Wohlin, C. Höst, M. Runeson, P., & Wesslén, A. (2001). Software Reliability, in Encyclopedia of Physical Sciences and Technology (third edition), Vol. 15, *Academic Press*.
- Dalal, S., & Mallows, C. (1988). “When Should One Stop Testing Software”, *Journal of the American Statistical Associations*, Vol. 81, 1988, pp. 872–879.
- Naik, K., & Tripathy, P. (2008). “Software testing and quality assurance: theory and practice”, *John Wiley & Sons.*
- Lyu, M. R. (2007). Software Reliability Engineering: A Roadmap, *International Conference on Software Engineering (FOSE '07)*, pp. 153-170.
- Farooq, S. U., & Quadri, S. M. K. (2010). ”Effectiveness of Software Testing Techniques on a Measurement Scale”, *Oriental Journal of Computer Science & Technology,* Vol. 3(1), 109-113
- Bertolino, A. (2004). The (Im) maturity Level of Software Testing. *SIGSOFT Software Engineering Notes* 29(5), 1–4.
- Young, M. (2008). Software testing and analysis: process, principles, and techniques. John Wiley & Sons.
- Juristo, N., Moreno, A. M., & Vegas, S. (2004). Reviewing 25 years of testing technique experiments. Empirical Software Engineering, 9(1-2), 7-44.

- P.G. Frankl and Y. Deng, “Comparison of Delivered Reliability of Branch, Data Flow and Operational Testing: A Case Study,” Proc. ACM Int’l Symp. Software Testing and Analysis, ACM Press, 2000, pp. 124–131.
- Vegas, S. (2004, August). Identifying the relevant information for software testing technique selection. In Empirical Software Engineering, 2004. ISESE'04. Proceedings. 2004 International Symposium on (pp. 39-48). IEEE.