An Improvised Algorithm for Improving Software Reliability

Taraq Hussain Sheakh
Research Scholar JJTU Jhunjunu, Rajasthan

ABSTRACT
In order to improve the reliability of Software, we need to implement better testing methods, but an attempt is made to the selection of the test is near to impossible task. The reliability of the software can be achieved by analyzing the test which is direly reliant upon the system. This research paper, tend to emphasize the selection of testing method and an algorithm which generate the reliability and an emphasis is made to generate the results reliability on the basis of faults and errors.

Keywords
Software Testing, Static Testing, and Dynamic Testing.

1. INTRODUCTION
Research on software reliability engineering has been conducted for more than 35 years and more than 70 statistical models have been proposed for measuring, estimating and improving software reliability. Most of the existing models predicting software reliability are purely based on observation of software failures[24]. However relevant information for the software development and improvement, the method of failure detection, environmental factors etc, are ignored. The demand for composite software systems has augmented more rapidly than the ability to design, implement, test, and maintain them, and the reliability of software systems has become a major concern for our modern society.IEEE (1991) defines software reliability as the probability of failure-free software operations for a specified period of time in a specified environment[8].

Mathematically, reliability R(T) is the probability that a system will be successful in the interval from time 0 to time t

$$R(T) = P(T>0),\ t \geq 0$$ [25]

It is one of the attributes of software quality, and is generally accepted as the key one since it quantifies software failures

2. SOFTWARE RELIABILITY
Software Reliability is defined as the probability that software will provide failure-free operation in a fixed environment for a fixed interval of time In fact software reliability is the foremost thing for the software engineering which propagates not only the functionality but also the operations of software quality. Software reliability also makes certain the prevention of the errors and failures that is the cause and concern of the barriers of reliability[23].

Software Reliability Engineering, consequently, is the field that quantifies the operational behavior of software-based systems with respect to user requirements concerning reliability[25]. It includes: 1. Software reliability measurement, which includes estimation and prediction, with the help of software reliability models established in the literature[24]; 2. The attributes and metrics of product design, development process, system architecture, software operational environment, and their implications on reliability[24]; and 3. The application of this knowledge in specifying and guiding system software architecture, development, testing, acquisition, use, and maintenance[24].

3. SOFTWARE TESTING
Software Testing defines the building of a program which emphasis the whether the inputs of the particular program ensures the expected and desired results. Software testing is an important component of software quality assurance, and many software organizations are spending up to 40% of their resources on testing. For risky software like flight control testing is very important and expensive as a consequence of much research on the risk analysis has been made and need to resolve out also.

Thus the possibility that a software project will experience undesirable events, such as schedule delays, cost overruns, or outright cancellation and other related operations should be controlled and measured[26]. Software testing is a process of executing a program with the goal of finding errors' and consequently to reduce it as far as possible so that the software will become approached to the accuracy of reliability. So, testing means that one inspects behavior of a program on a finite set of test cases (a set of inputs, implementation preconditions, and expected outcomes developed for a particular objective, such as to exercise a particular program path or to validate conformity with a specific obligation, for which appreciated inputs always exist[24].In practical the complete set of test is considered as infinitely large and theoretically there are many test cases for even simple or effortless programs. So how to select the most appropriate test cases? For the risk analysis need depth study and the research for the engineering expertise.

3.1. Static Testing techniques
Static testing/ Non-implementation based techniques focus on the range of ways that are used to prove the program irrespective of real execution of the program.

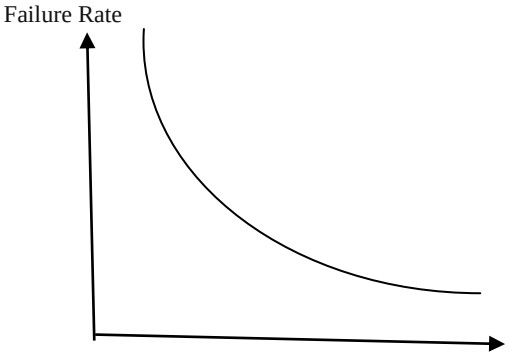

Fig.1 Basic Ideas of Software Reliability

It is not generally comprehensive testing, but verify the code, or document, it is mainly concerned with the analysis and scrutiny of system representation such requirements documents, design diagrams and the program source code,

either manually or automatically, without actually executing the code[26]. Techniques in this area include code inspection, program analysis, symbolic analysis, and model checking[26] etc.Testing is an activity performed for evaluating software quality and for improving it. Hence, the goal of testing is

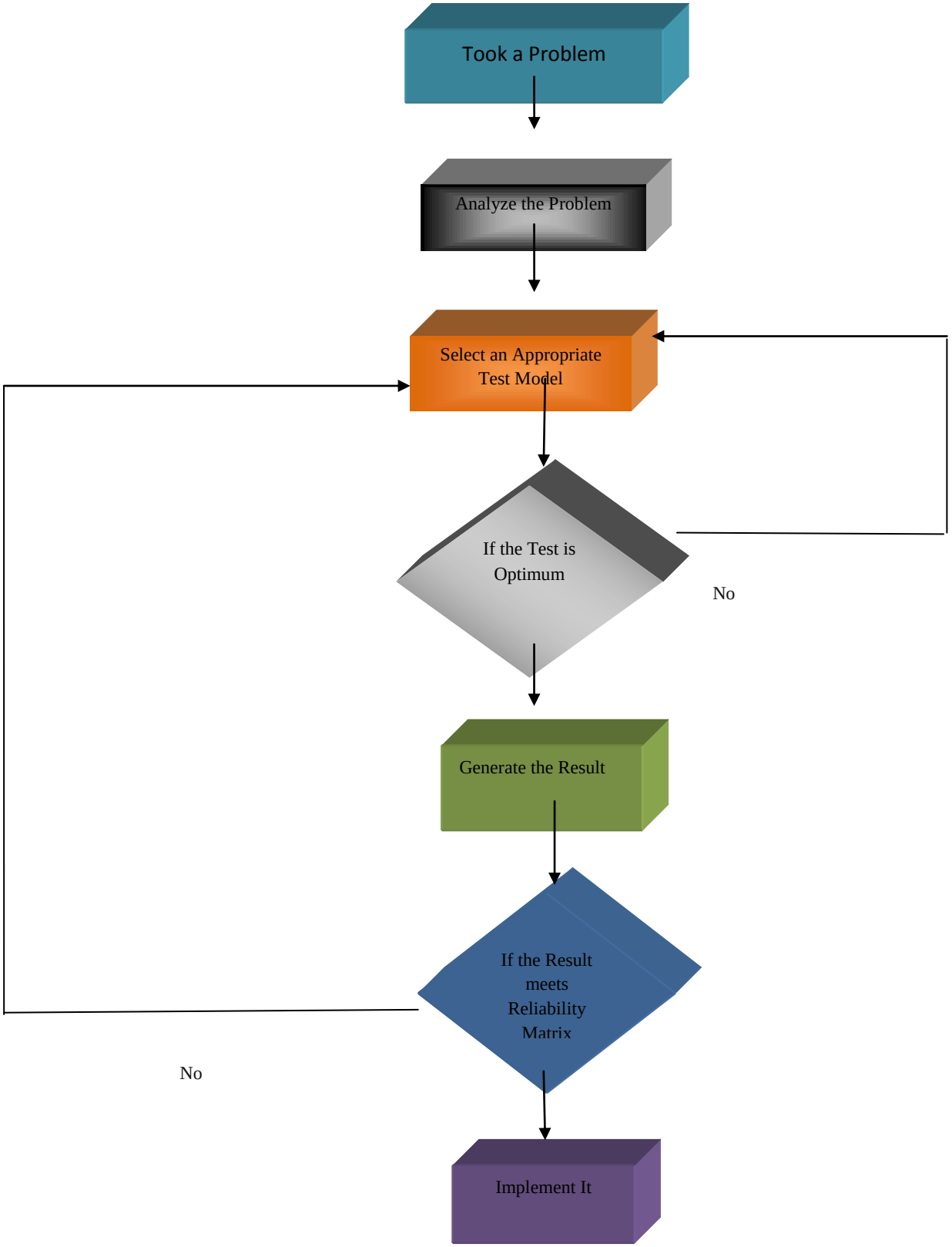

Fig.2 An Improvised Algorithm

systematical detection of different classes of errors (error can be defined as a human action that produces an incorrect result, in a minimum amount of time and with a minimum amount of effort

3.2. Dynamic Testing

Whenever we choose techniques that are used to discover software quality and authenticate the software through actual executions of the software under test[26]. We test the software with real or stipulated inputs, both normal and abnormal, under controlled and predictable conditions to check how a complex, non deterministic system might respond with different behaviors to same input, depending on the system state only by studying levels set for the dynamic aspects evaluated are met.

S.No	No. of Errors in %	Reliability in %
1.	E >70	Nil
2.	E > 50 and E< =70	20 %
3.	E >30 and E <=50	40%
4.	E > 10 and E <=30	60%
5.	E <=10	Maximum

Table 1. Reliability Matrix

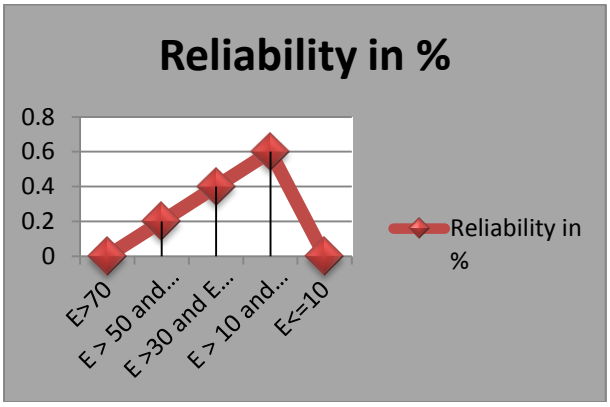

Fig.3 Line Chart Showing Reliability Matrix

Fig.1 shows that on removing the faults entirely from the software lot of time is needed i.e. failure rate decreases with the consumption of more time for test.

Fig.2 shows An Improvised Algorithm which will be used ensures the reliability of software. The prime facie of the Algorithm is to choose an appropriate test model which further generates the quality.As mention in 3.2 dynamic testin -g ensures the quality as well as the reliability of software if the inputs are taken in controlled environment.

Table 1. generates the reliability matrix on the accord of error In percentage. But it is almost practically impossible to have Errors lesser than 10% and reliability maximum which further offer the research problem.

4. FACTORS FOR SELECTING SOFTWARE TECHNIQUES

For the introduction of the software in the organization it not only meet the requirements but also solve their needs effectively and efficiently. On the other hand organization should also ready to adopt the new changes with the controlled environment. The tool should help in building the strengths of the organization and should also address its weaknesses. The organization needs to be ready for the changes that will come along with the new tool if the latest testing techniques do not full fill the obligation for the end user, and then it is necessary that the organization must choose steps to improvise the testing practices.

Certainly, the processes can be improvised by introducing parallel tool to support those practices and it can always pick up some good ideas for improvement from the ways that the tools work. However, it do not depend on the tool for everything, but it should provide support to your organization as expected.

The following factors are important during tool selection:

- Accessing the organization's maturity.
- Pick out the techniques in the problem domain of organization that will help to improve the testing process.
- Estimation and Evaluation of techniques of problem domain for requirement and objective area.
- Evidence to show that whether the product works for the desired objectives and requirement.
- Evaluation of the vendor (training, support and other commercial aspects) or open-source network of support;
- Identifying and preparation internal execution.

5. EXPERIMENTATION PROBLEM[26]

1) A comparison of testing techniques is to filter out errors and faults so as to produce effectiveness and efficiency. A criterion for comparative analysis of testing techniques is usually not well defined.
2) As most of the studies do not focus on the comparative analogy, thus creating ambiguity in test technique selection.
3) Previously studies show the difference according to the number and parameter chosen. A common standard is missing for their comparison.
4) As much of studies also do not emphasize the factors of experimental control.
5) Many experimental techniques generate faults and errors which result to less random variation in fault rate and more statistical power.
6) We have another drastic problem that only small sample are chosen for experimental evaluation which demonstrate the better outcome than other.
7) Experimental techniques biased either towards academic or industrial evaluation on the basis of academic and industrial system.

6. DISCUSSIONS AND CONCLUSIONS

Software reliability modeling has drawn a focus to and various research Scholars attempt to leads to software quality. Which thereafter boost the various industries seriousness of software quality measurement? In order to provide reliability to the software, it demands effective software testing techniques. Although any efficient software testing procedures demand time and cost. As static testing focus on the documentation, Tester needs to emphasize and depend upon the dynamic testing for improving the reliability. There lies the authentic problem. Research scholars have to consider

the dynamic testing techniques that will used to debug the errors or faults in minimum time and cost.

7. FUTURE SCOPE

There are many techniques and method which have been used in order to provide the reliability of the Software, but some of them have been proved the reliability, but it is also true that not any methods have the ability to realized 100% percent reliability of software in practical. In this regard if this algorithm is used, it will provide the reliability, which is the theme of the problem. This Paper further defines the problem domain and the area for Research.

8. REFERENCES

[1] Crow, L.H., 1974. "Reliability Analysis for Complex Repairable Systems," Reliability and Biometry, F. Proshan and R.J. Serfling *(eds.) SIAM, Philadelphia, 379-410.

[2] Dalal, S.R. and C.L. Mallows. 1988. When should one stop software testing? J. Amer. Statist. Assoc. 83:872-879.

[3] Dalal, S.R. and C.L. Mallows. 1990. Some graphical aids for deciding when to stop testing software., IEEE J. Special Areas in Communications 8:169-175. (Special issue on Software Quality & Productivity.)

[4] Dalal, S.R. and C.L. Mallows. 1992. Buying with exact confidence. Ann. Appl. Prob. 2:752-765.

[5] Dalal, S.R. and A.M. McIntosh. 1994. When to stop testing for large software systems with changing code. IEEE Trans. Software Engineering, 20:318-323.

[6] Gaffney, J.D. and C.F. Davis, 1988. "An Approach to Estimating Software Errors and Availability," SPC-TR-88-007, version 1.0, March 1988, Proceedings of the 11th Minnowbrook Workshop on Software Reliability, July 1988.

[7] Goel, A.L. and K. Okumoto, 1979. "Time-Dependent Error-Detection Rate Model for Software and Other Performance Measures," IEEE Transactions on Reliability, R- 28(3):206-211.

[8] Institute of Electrical and Electronics Engineers, 1991. ANSI/IEEE Standard Glossary of Software Engineering Terminology, IEEE Std. 729-1991.ISO, 1991. "Quality Management and Quality Assurance Standards - Part 3: Guidelines for the Application of ISO 9001 to the Development, Supply and Maintenance of Software," ISO 9000-3, Switzerland.

[9] Keiller, P.A., B. Littlewood, D.R. Miller, and A. Sofer, 1983. Comparison of software reliability predictions, Proceedings of the 13th IEEE International Symposium on Fault-Tolerant Computing (FTCS-13), Milano, Italy, 128-134.

[10] Littlewood, B and V. Verrall, 1973. A Bayesian Reliability Model with a Stochastically Monotone Failure Rate, IEEE Transactions on Reliability, R-23(2):108-114.

[11] Lee, L., 1992. The Day the Phones Stopped: How people Get Hurt When Computers Go Wrong, Donald I. Fine, Inc.

[12] New York. Lyu, M.R., 1996. Handbook of Software Reliability Engineering, McGraw-Hill, New York.

[13] Moranda, P.B. and Z. Jelinski, 1972. Final report on Software Reliability Study, McDonnell Douglas Astronautics Company, MADC Report Number 63921.

[14] Moranda, P.B., 1975. Predictions of Software Reliability During Debugging, Proceedings of the Annual Reliability and Maintainability Symposium, Washington, D.C., 327-332.

[15] Musa, J.D. and K. Okumoto, 1984. A Logarithmic Poisson Execution Time Model for Software Reliability Measurement, Proceedings Seventh International Conference on Software Engineering, Orlando, Florida, 230-238.

[16] Musa, J.D., A. Iannino, and K. Okumoto, Software Reliability - Measurement, Prediction, Application, 1987. McGraw-Hill, New York.

[17] Musa, J.D., G. Fuoco, N. Irving, D. Kropfl, and B. Juhlin, 1996. "The Operational Profile," Chapter 5 in (Lyu 1996), 167-218.

[18] Rome Laboratory, 1987. Methodology for Software Reliability Prediction and Assessment, Technical Report RADC-TR-87-171 ; revised on Technical Report RL-TR-92-52, 1992.

[19] Schneidewind, N.F., 1975. "Analysis of Error Processes in Computer Software," Sigplan Note, 10(6):337-346.

[20]Singpurwalla, N.D. 1991. Determining an optimal time interval for testing and debugging software. IEEE Trans. Software Engineering 17(4):pp313-319 Schick, G.J., and R.W. Wolverton, 1973. "Assessment of Software Reliability,"

[21] Proceedings of the Operations Research, Physica-Verlag, Wurzburg-Wien, 395- 422. South West Thames Regional Health Authority, 1993. Report of the Inquiry into the London Ambulance Service.

[22] Yamada, S., M. Ohba, and S. Osaki, 1983. "S-Shaped reliability Growth Modeling for Software Error Detection," IEEE Transactions on Reliability, R-32(5):475-478.

[23] Software Reliability by S.R Dalal, M.R Lyu, C.L Mallows Bellore, Lucent Technologies, AT&T Research.

[24] Software Testing Methods and Techniques by Jovanovic, Irena.

[25] Software Reliability by Hoang Pham.

[26]"Evaluating Effectiveness of Software Testing Techniques with emphasis on enhancing Software Reliability" by Sheikh Umar Farooq and S.M.K Quadri on Journal of Emerging Trends in Computing and Information Sciences Vol 2. No.12, Dec. 2011.