International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 06 Issue: 10 | Oct 2019 www.irjet.net p-ISSN: 2395-0072

A Study on Software Reliability Models

B. Vasundhara Devi[1]

[1]*Assistant Professor, Sreenidhi Institute of Science and Technology, Ghatkesar, Hyderabad, India*

Abstract - *In the present generation, software is of no use unless it is reliable. Organizations tend to focus on reliability along with efficient and error-free software. The only way to ensure the software's lesser crash rate is to keep it more reliable. The key idea to present this paper is to give a systematic study on the works carried out so far and give the scope of future research. The paper first presents the present scenario of software reliability and continues to discuss on various existing models along with its demerits, then we discuss the metrics used for evaluating such models and we interpret the results achieved by the existing models. Finally, we end up with the scope to carry out future work.*

***Key Words*: Software Reliability, Models, Metrics, Interpretations**.

1. INTRODUCTION

Reliability in general refers to believing something or something performing well. Software Reliability is the key attribute in measuring the overall quality of software, together with its easy to use, ease to maintain and performance of the software. It has been a major concern for the present society to have reliable software systems. Software reliability is basically measured by the number of failures occurred during development phase along with other information which includes the failure occurrence time, type of failure found, in which part of the software, current state of the software at that time, the nature of failure, etc. Quality improvement costs a lot if the software is not reliable. Software Reliability is not an easy task to achieve as the complexity involved in making software is very high. Any software tends to change from time to time, so the developers need to make sure that the complexity is considered while developing software with the ease of upgrading the software. Software reliability predicts the probability of performance of software despite of failure. It affects the overall system or project performance as well. In general, we can measure the total system with Fault Analysis Tree and Reliability Block diagrams. Reliability models are categorized into two types: software models and hardware models. Hardware models are statistical and probabilistic models. In addition to these two models, we also have some open source reliability models available. The key idea of writing this paper is to provide a reference for various reliability estimation models.

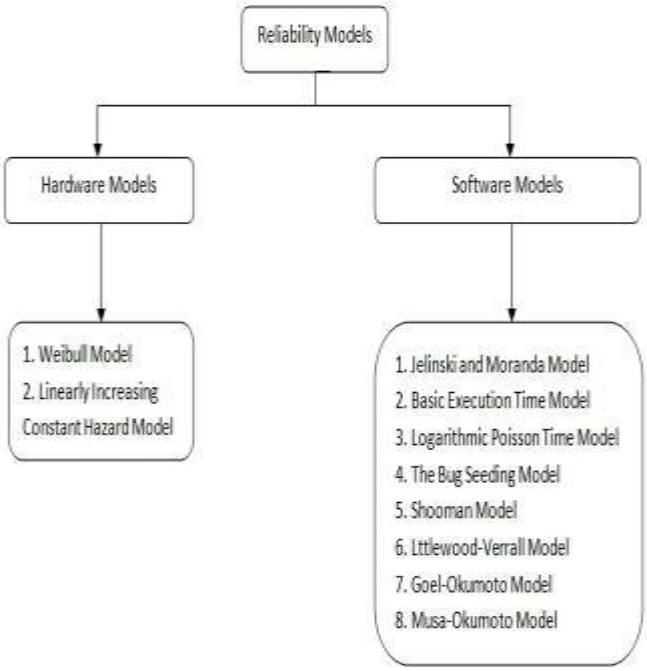

Fig-1: Reliability Models

In section 2, we will be discussing the above said models with respect to the effort cost, finite or infinite defect count along with exact time between failures. Section 3 discusses the metrics used for evaluating software reliability along with our interpretations towards it. Final section comprises of concluding remarks along with our focus on the future work that we are going to carry out in the upcoming research.

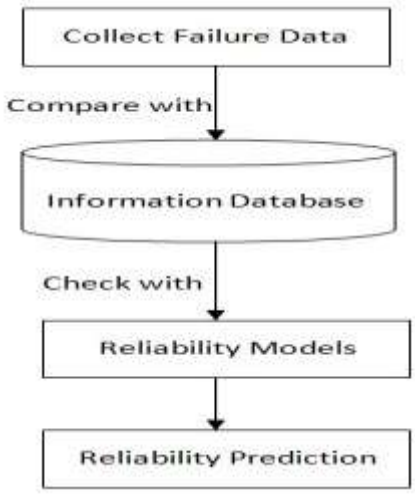

Fig-2: Reliability Prediction Scenario

2. BACKGROUND

- Weibull Model requires more effort for reliability maintenance as the defect count is not fixed with increasing fault rate.
- Linearly Increasing Constant Hazard Model requires high effort as the defect count is infinite.
- Jelinski and Moranda Model has medium effort rate as the defect count is finite or fixed with decreasing fault rate. It requires exact time between failures.
- Basic Execution Time Model needs medium effort as the defect count is finite which also require the exact time between failures.
- Logarithmic Poisson Time Model needs high effort as it is non-linearly decreasing even though the defect count is fixed.
- The Bug Seeding Model requires medium effort as it is increasing and then decreasing and even the defect count varies.
- Shooman model requires low effort as the defect count is fixed or finite and the time requirement between failures is needed.
- Littlewood-Verrall Model requires high effort as it is linearly decreasing with varying defect count.
- Goel Okumoto Model requires medium effort as the defect count is fixed which is linearly decreasing and need exact time between failures.
- Musa Okumoto Model needs low effort as it is non-linearly decreasing with finite and fixed defect count which also requires exact time between failures.
- Geometric Model needs high effort as the defect count is infinite with exact time between failures requirement.
- Duane Model need medium effort with infinite defect count which is linearly decreasing and it does not require the exact time between failures.
- Yamada and S-shaped model effort rate requirement is high as it is increasing and then decreasing with not finite defect count which even needs exact time between the failures.

3. METRICS AND INTERPRETATIONS

Software Reliability cannot be measured exactly. Even though we use some metrics to assess software reliability, it is just to give the characteristics. As software is a complex program involving several factors, it is quite difficult to measure the reliability directly. But the most commonly used measurements are:

- Product metrics can be assessed using Lines of Code (LOC) or in thousands (KLOC) which is the simplest method. In addition to this, we can even use Function Point (FP) metric which is used to measure the functionality of software.
- Project management metrics can also be used to assess the reliability of software. High reliability is assured based on the process model used along with risk management process and configuration management process involved.
- Process metrics can be used to optimize the quality and reliability of software using the Quality Management Standards (QMS) as developed by ISO.
- Failure metrics can also be one of the measures to assess the reliability of software. The key goal of this metric is to find the software's failure-free execution. Mean Time between Failures (MTBF) and Mean Time to Repair (MTTR) are some of the parameters used to measure software reliability.
- Efficiency is one of the key metrics to measure the software reliability which is determined based upon the computing time and resource utilization that separates high quality and low quality software's.
- Integrity is another important factor used to determine the reliability of a software as it is the responsibility of the developing team to protect the software against hackers and misuse by unauthorized people.
- Flexibility is the ability of software to transfer from one plat form to another.
- Interoperability is the effort involved in coupling one system to other based on few sub-features like adaptability, replacebility, conformance and insatiability.
- Maintainability is the most important measurement used to assess the software reliability. It deals mainly with stability and changeability nature.

4. CONCLUSION AND FUTURE WORK

We have presented this paper to discuss various models used to predict software reliability along with pros and cons involved with each and every model. We have also given the reliability prediction scenario. We have given the metrics that are used to measure software reliability. But during the research done in writing this paper, we have gone through many papers and many works carried by researchers. Our observation identified that there are models available with the use of machine learning techniques but they can be optimized further. So we would like to focus on using machine learning techniques along with fuzzy logic for predicting the software reliability in a better and optimized way.

REFERENCES

[1] Dr. Shelbi Joseph, Akhil P V Abdul, "A Survey of Software Reliability Models", International Journal for Research in Applied Science & Engineering Technology (IJRASET), Volume 5 Issue XII December 2017.

[2] Ejem, Agbaeze, Diala, S. O, Okpalla C. L, "A SURVEY OF SOFTWARE RELIABILITY: MODELING, MEASUREMENT AND IMPROVEMENTS", International Journal of Computer Trends and Technology (IJCTT) – volume 31 Number 1 – January 2016.

[3] Joicymara Xavier, Autran Macêdo, Rivalino Matias, Lucio Borges, "A Survey on Research in Software Reliability Engineering in the Last Decade", 29th ACM Symposium On Applied Computing, At Gyeongju, Korea, March 2014.

[4] Bonthu Kotaiah, Dr. R.A. Khan, "A Survey on Software Reliability Assessment by Using Different Machine Learning Techniques", International Journal of Scientific & Engineering Research, Volume 3, Issue 6, June-2012.

[5] Apoorva Singhal, Ankur Singhal, "A Systematic Review of Software Reliability Studies", Software Engineering : An International Journal (SEIJ), Vol. 1, No. 1, September 2011.

[6] Aasia Quyoum, Mehraj – Ud – Din, S. M. K. Quadri, "Improving Software Reliability using Software Engineering Approach- A Review ", International Journal of Computer Applications, Volume 10– No.5, November 2010.

[7] Siddhartha R. Dalal, "Software Reliability Models: A Selective Survey and New Directions".

[8] James J. Cusick, "The First 50 Years of Software Reliability Engineering: A History of SRE with First Person Accounts".