International Journal of Advances in Engineering and Management (IJAEM)
Volume 2, Issue 3, pp: 155-160 www.ijaem.net ISSN: 2395-5252

Overview of Analysis of Software Reliability Process

[1]B.R. Kavitha, [2]P.T. Jamuna Devi

[1]Assistant Professor, Department of Computer Science,Vivekanandha College of Arts and Sciences for Women,Elayampalayam.

[2]Chief Editor, Department of Computer Science, Kalaivani Research and Publication Center, Erode.

Corresponding author: B.R. Kavitha

Date of Submission: 15-07-2020 Date of Acceptance: 31-07-2020

ABSTRACT: The requirementforadvanced systems has improvedmore quickly. Over the past few decades, the complexity and size of the mobile have been increased in a very remarkable way. Software reliability has been is based upon themethods of engineering that incorporates theadvancement and maintenance of the software methodsfor which the reliability is detectable. Software reliability has been a major subject of scientific studiesaround the past many years, yet,research studies are happening. Software reliability is deemed to be a significant factor for quality of software. This is a systemreliabilityidea. System dependenceis growing by the daybecause of which Software reliability has become a considerable concern of users. SRE may be defined asthe survey of the procedures and outcomes of a software system that is the fundamental condition of all the customers. Numerous software reliability simulations have been detected since 1972. A lot of work was performed on software reliability estimation. This paper gives an overview of characters of Software reliability, software reliability growth models, factors that affect reliability, Software reliability activities, metrics, modeling, framework, improvement techniques, and testing tools.

KEYWORDS: Software reliability activities, Software reliability, System dependence, testing tools.

I. INTRODUCTION

A continuous availability is a requirement for critical business applications. software reliability is an important component of continuous application availability. Evolving reliable software is one of the maximumproblematicchallenges confronting the software industry. Plan pressure, resource restrictions, and impractical requirements can all negatively impact software reliability. Emerging reliable software is particularlydifficultonce the software components are mutually dependent as is the situation with much of the current software. It is also a difficult problemto find out whether or not the software will be delivered is reliable. Software reliability models try to deliver that information. Basically, there are two kinds of software reliability models - the ones that try toforecast software reliability from planning parameters and those that try to forecast software reliability from test data. The initialkind of model iscommonly referred to as "defect density" models and utilizes code attributeslike input/outputs, external references, nesting of loops, lines of code, and so on to assess the number of defects in the software. The next kind of model isnormallyreferred to as "software reliability growth" models. These models try to statistically link defect detection data with established functions like an exponential function. Ifthe relationship is excellent, the established function can be utilized to forecastupcomingbehavioral patterns. Software reliability growth models are the emphasis of this report. Most software reliability growth models have a factor that correlates to the total number of defects that are included in a set of code.

II. SOFTWARE RELIABILITY

Software reliability is of major concern for the investigatorsand the software developers since the high dependence of people on the software systems in their everyday life is detected recently. Because of this high dependency, software activities are attempting to createprogressively more reliable software. Software reliability is commonly described as the option of unsuccessful free operations carried out by the software within its accurate environment. The most widespread method to assess the software reliability in accordance with the conditions of failure is the implementation of SRGM. The reliability assessment through the SRGMs is reliant on the failure the data relating to any software throughout the testing phrase. Further, SRGMs attempted to create a meaningful connectionamong the faults discoveredthroughout the testing and logarithmic/exponentialstatistical functions.

III. SOFTWARE RELIABILITY GROWTH MODELS

Reliability is generallyspecified as the possibility that a system will manage without malfunctionfor a specific time periodin accordance with specificoperationalcircumstances. Reliability is anxious with the time among reciprocal or its failures, the failure rate. In this article, data is considered from a test environment, so defect detection rate was detected insteadof failure rate. Defect detection is generally a failure throughout a test, although test software may also reveal a defect although the test is continuing to function. Defects can also be noticedthroughout code inspections or design reviews. Time in a test environment is a synonym for amount oftesting, which can be calculated in numerous ways. Defect detection data is comprised of a time for group of defects or every defect and can be mapped as demonstrated in Figure 1.

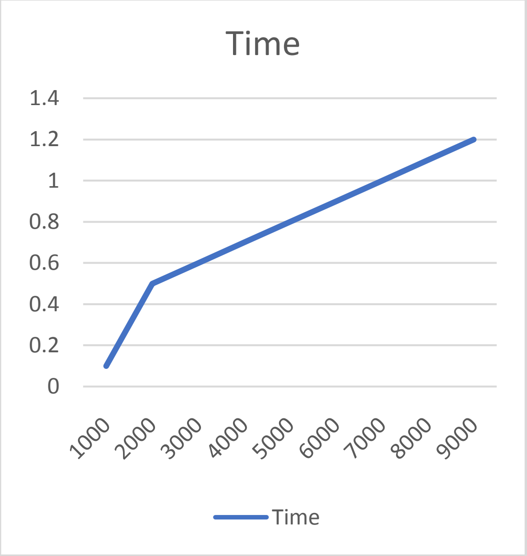

Figure 1 Time to failure data

IV. FRAMEWORK FOR RELIABILITY

The framework has four classifications for techniques comparison. These classificationshave different components and the questions associatedwith every component. The framework explains the attributesnecessary for the analysis techniques. The classifications of the framework are based upon the Normative Information Model-based Systems Analysis and Design (NIMSAD) framework. NIMSAD categorizes the techniquecomponents into four classifications:

- context,
- user,
- method content, and
- evaluation.

Table 1 List of Framework

Framework	Description
Context	The technique is analyzed from the angle of the problem situation
User	The technique is explored from the viewpoint of the intended method users
Method Content	The emphasis of the examination is the content of the method itself
Evaluation	This emphasis on the evaluation of the method context, user, and content. It confirms the experience of the method and the findings of the method.

V. SOFTWARE RELIABILITY MODELING

It is well established that evaluating the reliability of software applications is a most important issue in reliability engineering. Forecasting software reliability is not an easy function. The major problem is anxiousprincipally with design faults, which is an entirely dissimilarcondition from that operated by traditional hardware hypothesis. A fault implies to a manifestation in the code of an error made by the designer or programmer with regard to the design of the software. Stimulation of a fault by an input value results in an inaccurate output. Exposure of such an event links to an incidence of a software failure. The input values to the software modules (functions) either externally or internally may be deemed to bedisembarking to the software randomly. Even Though software failure may not be engendered stochastically, it may be discovered in such a way. Consequently, it defends the usage of stochastic models of the underlying random process that regulates the software failures.

Six types of models were deemed as potential candidates for modeling the reliability of software. Classification of software reliability models is offeredas per software development life cycle phases as shown in Figure 2. The six classificationsconsist ofinput domain models, reliability growth models, hybrid black box approach, hybrid white box approach, architectural based models, and early prediction models were listed in Table 2.

Table 2 Six types of Model

Types of Model	Description
Input domain model	utilizes the characteristics of the input domain of the software to develop a accuracy probability assess from test cases that executed appropriately.
Reliability growth model	This model captures failure behavior throughout testing and extrapolates it to behavior throughout operation. Consequently, this category of models utilizes trends a failure data detected in the failure data to derive reliability predictions.
Architectural based models	This model emphasis on the structural design of the software and derive reliability estimates by blending estimates acquired for the various modules of the software.
Early prediction model	utilizes characteristics of the software development process from requirements to test and extrapolates this information to behavior throughout operation.
Hybrid White box models	utilize selected characteristics from both black and white box models. Though, each group of models has its fundamental flaws once applying them to safety systems
Hybrid Black box models	It blends the characteristics of software reliability growth models and white box models.

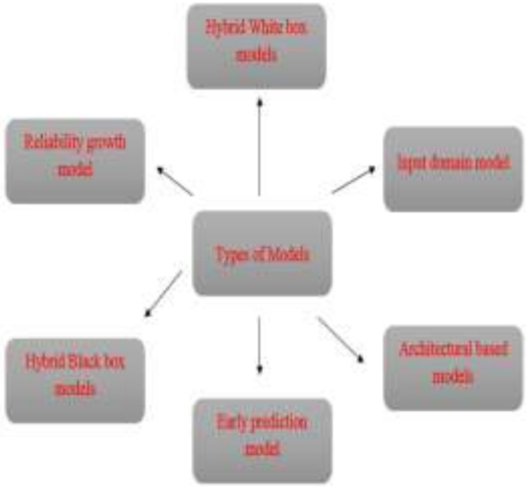

Figure 2 Classification of Models

VI. SOFTWARE MODELING TECHNIQUE

This technique can be split into twosubgroups.
1. Prediction Modeling
2. Estimation Modeling

- Important results can be achieved by implementingthe appropriate models.
- Theories and conceptscould be made to streamline the difficulties and no single model will appropriate for all the circumstances.

The major differences betweenthe two models are: -

Issues	Prediction Models	Estimation Models
Data Reference	It utilizes historical data	It utilizesexisting data from software development.
Onceutilized in Development Cycle	It will be mostly formed before the testing or development stages.	It will be commonly utilizedin the subsequent phase of the Software Development Life Cycle.
Time Frame	It will anticipate the reliability in the future.	It will forecast the reliability both for the present time and in the future time.

.Factors that Affect Software Reliability

1. The quantity of faults describes in the software
2. The manner in which users manage the system

Characteristics of Software Reliability

The three characteristics of software reliability are

1. Failure appears primarily because of design faults
2. Reliability is not time reliant
3. Absence of wear-out trend

Characteristic	**Description**
Failure appears primarily because of design faults	Design is adapted for renovations to make it vigorous against conditions that can identify a failure.
Reliability is not time reliant	Failure occurs because of the error susceptible to execution. The development of reliability is detected as errors are discovered and rectified.
Absence of wear-out trend	Software errors happen without any alert. Due to errors, while performingimprovements, the “Old” code may lead to a greater number of failure rates.

VII. SOFTWARE RELIABILITY ACTIVITIES

The software reliability process involves software development, processes, and maintenance. A software reliability process involvesexpenses, upgrading, errors, corrections, defects, and faultson the resource like the workforceattempt. Some of the Reliability activities are as follows:

Software reliability activities	**Description**
Construction	Generating new document and code items.
Combination	It is an emphasis on reusability of code components and old documents with the new one
Correction	Examining and eliminating code and document linked to the defects by examining the test items.
Preparation	Creation of various test items
Testing	Executing the test cases, to realize the trigger points in which failure happens often
Identification	Each bug or error if old or newly confrontedis to be classified
Repair	Faults are eliminated which probablypresents new faults for which regression testing is performed.
Validation	Performingtests to make sure that repairs are efficient and have not adversely affectedadditional components of the software
Retest	Executing the cases to verify fora defined completion of the repair. Whether it is unfinished, new test cases may be necessary to fix them further.

VIII. SOFTWARE RELIABILITY METRICS

Measurement of Software reliability is an unresolvedproblembecause we don't have a sufficient understanding of the software’s environment. Some of the metrics were utilized to evaluate the software reliability are

Metrics	**Description**
Project management metrics	A good qualityproject could be accomplished byretainingproper managementthat leads to in completion of projects as well aswith the aims of the quality. If designers havenot sufficient processes, the rise in the expenseis

	happening. Enhancedand effective reliability is directedby enhancingthe risk management process, development processes,strategy management, procedure management,and so on.
Product metrics	KLOC (i.e. LOC in thousands) or LOC (i.e. Lines of Code), is a method for measuring the volume of the software. Normally, statements and the comments are non-executable which are not included in the calculation, and source code will be utilized. The function point metric determines the features and capabilities of a suggested software developingas per the count of interfaces, outputs, and inputs. It contributes to the appropriate measuresof the essential functions of the software. Complexity-oriented metrics define thecomplexity of program structure, by the simplification of code into the graphical view as complexity is associated with the reliability of the software, so the sophisticated representation is essential.
Fault and failure metrics	Fault and failure metrics assist in achieving the free execution of software without failure. A lot of errors were discovered at the testing stageinstead ofproviding it to the customer as well as the number of faults confronted and informed by the users following software delivery are merged, examined and reviewedto accomplish this main goal.
Process metrics	Process metrics areutilized for an estimate, monitor and improve the reliability and quality of software
Efficiency	The number of resources and data processing time necessary for the software to accomplish the needed function is a crucial element indistinguishing betweenhigh and low-quality software.
Integrity	The software access through a hacker or an unauthorized person can be manipulated byenhancing the Integrity methods.
Flexibility	The capability of software to be compliant with various hardware distinguishes its flexibility
Maintainability	The software needs time to time maintaining, and its expenditure is also extremely high.

Tools for Testing Reliability

Some of the Tools utilized for Software Reliability are

1. RCM:-Reliability Centered Maintenance
2. RGA:- Reliability Growth Analysis
3. WEIBULL++:- Reliability Life Data Analysis

IX. CONCLUSION

Software Reliability is regarded as the most effectivefeature for the overall quality of software. Hardware can rust or age, but software does not. An unreliable behaviorof S\softwareis essentiallybecause of errors or faults in the design of the established software. Several Models exist, but not single model can characterize the required or expected behavior of the software under varioussituation. Any of the model was generally accepted for all kinds of software. In this article, an overview of characters of Software reliability, software reliability growth models, factors that affect reliability, Software reliability activities, metrics, modeling, framework, improvement techniques, and testing tools were defined.

REFERENCES

[1]. Quyoum, Aasia, M. D. Dar, and S. M. K. Quadri. "Improving software reliability using software engineering approach-a review." International Journal of Computer Applications 10.5 (2010): 41-47.

[2]. Kaur, Gurpreet, and Kailash Bahl. "Software reliability, metrics, reliability improvement using agile process." IJISET-International Journal of Innovative Science, Engineering & Technology 1.3 (2014): 143-147.

[3]. Jatain, Aman, and Yukti Mehta. "Metrics and models for software reliability: A systematic review." 2014 International Conference on Issues and Challenges in Intelligent Computing Techniques (ICICT). IEEE, 2014.

[4]. Ali-Shahid, Malik Muhammad, and Shahida Sulaiman. "Improving reliability using software operational profile and testing profile." 2015 International Conference on Computer, Communications, and Control Technology (I4CT). IEEE, 2015.

[5]. Romankevich, Alexei, et al. "Fault-tolerant multiprocessor systems reliability estimation using statistical experiments with GL-models." International Conference on Computer Science, Engineering and Education Applications. Springer, Cham, 2018.

[6]. Cusick, James J. "The First 50 Years of Software Reliability Engineering: A History of SRE with First Person Accounts." arXiv preprint arXiv:1902.06140 (2019).

[7]. Xu, Jianli. "Evaluating and balancing reliability and performance properties of software architecture using formal modeling techniques." 2006 30th Annual IEEE/NASA Software Engineering Workshop. IEEE, 2006.

[8]. Sharafi, Sayed Mehran. "SHADD: A scenario-based approach to software architectural defects detection." Advances in Engineering Software 45.1 (2012): 341-348.

[9]. Woodside, Murray, Greg Franks, and Dorina C. Petriu. "The future of software performance engineering." Future of Software Engineering (FOSE'07). IEEE, 2007.

[10]. Bertolino, Antonia. "Software testing research: Achievements, challenges, dreams." Future of Software Engineering (FOSE'07). IEEE, 2007.

[11]. Ahmed, Waseem, and Yong Wei Wu. "A survey on reliability in distributed systems." Journal of Computer and System Sciences 79.8 (2013): 1243-1255.