Volume 2, No. 12, December 2011
Journal of Global Research in Computer Science
ISSN-2229-371X
RESEARCH PAPER
Available Online at www.jgrcs.info

ENHANCED VERSION OF GROWTH MODEL IN WEB BASED SOFTWARE RELIABILITY ENGINEERING

D.Swamydoss*, Dr. Kadhar Nawaz
*Department of MCA Adhiyamaan College of Engineering, Hosur.
Director, Department of MCA Sona College of Tech., Salem

Abstract: The task of improving software reliability for large systems is becoming more difficult due to the highly-distributed nature of current products, and their deployment on dissimilar platforms with complex service functionalities. This coupled with increasing software complexities and large software size requires a very different approach in assessing software reliability. Software reliability in web based application is some what complex compared to other software systems. The customers of web application expect more reliability during the usage of application. So the reliability of the web based applications can be considered as a special case of distributed software running on distributed computer systems, over different kinds of network. Prediction of hardware failures is much more straightforward while prediction of software failures is more complex. Use of sophisticated software Reliability tools and techniques to predict the reliability of software products before they are released. This paper discussed the relevant aspect of reliability issues and some best practices of growth model when thinking in terms of web software reliability.

Keywords: Software Reliability, Growth model, MTTR

INTRODUCTION

Since the late 1960's there has been a variety of methods available for predicting software defects. Unfortunately, nearly all these methods are unusable until system testing has started. So they have limited benefit to a software engineer since the code has already been written by the time the prediction method can be used.

Software reliability engineering is centered on a key attribute, software reliability, which is defined as the reliability of failure free software operation for a specified period of time in a specified environment. Among other attributes of software quality such as functionality, usability, capability and maintainability etc., software reliability is generally accepted as the major factor in software quality since it quantifies software failures, which can make a powerful system in operative. Software reliability engineering (SRE) is therefore defined as the quantitative study based systems with respect to user requirements concerning reliability.

SRE techniques so far we used have number of weaknesses. They are;

a. The failure data collected during testing phase, it is too late for any changes required in the design phase.
b. The failure data collected in the in-house testing, the failures that would be uncovered under actual environment.
c. The existing SRE techniques are based on some unrealistic assumptions that make the reliability estimation too optimistic relative to real situation.

Thus, although SRE has been around for a while, credible software reliability techniques are still urgently needed, particularly for web based software. In the following section we have discussed the reliability in web based software, Growth model for software reliability. Finally we propose the role of growth model in web software reliability.

WEB BASED SOFTWARE RELIABILITY

When we discuss about web based software reliability, we have to take into account many technologies, each one having its own failure modes and sources of delay and unreliability. Web application, which is a collection of servlets, html pages, classes and other resources that can be bundled and run on multiple containers from multiple vendors. In the web application, the failure of any one system or service in the path between server and user will in effect cause failure of the entire application as far as the user is concerned.

The most heavily used websites are characterized by high reliability, high availability, and high security and more interactive. All these characters are independent on the web application. Most of web application software are built using layered architecture. Three tier architecture is widely used now-a-day. This architecture has three entities; client, server and database. The web application has n-tier architecture based on pattern design, plug-ins, with a modular structure using a specific user interface. An important factor influencing the web server reliability is the network reliability and availability. In web application, there is strong connection between the reliability an d the network perform ability. A global analysis considers both hardware and software fault categories when studying the web application reliability. Use of software architecture analysis to study the current software architecture of products and processes with evolution plans for enhancing the software reliability of newer products with minimal impact on existing products. Researched data shows that software reliability helps on the average by; increasing productivity by one third, increasing annual gain in early, detection of defects by one quarter, reduction in delivered defects by one fifth, etc.

According to ANSI, Software Reliability is defined as: the probability of failure-free software operation for a specified period of time in a specified environment. Software reliability engineering is therefore defined as the quantitative study of the operational behavior of software based systems with respect to user requirements concerning reliability. As a proven technique, SRE has been adopted either as standard or as best current practice by more than 50 organizations in their software projects and reports, including, AT & T, Lucent, IBM, NASA and in many other countries. However, this number is still relatively small compared to the large amount of software producers in the world.

SOFTWARE RELIABILITY GROWTH MODEL (SRGM)

Reliability growth models are based upon the assumption that the reliability of a program is a function of the number of faults that it contains. Such models apply statistical techniques to the observed failures during software testing and operation to forecast the product's reliability. In order to be effective, the data used in the growth model is taken from where the software will be deployed. For a general purpose software product, once the product is released to a large number of users, the possibility of providing an accurate measure of reliability, shortly after product release, becomes more feasible as large amount of failure data can be captured from the end users through various data collection processes.

For measuring reliability, a simple method that has been used earlier to determine the total number of failures experienced by all the operational units and then divide it by the number of units in the field. If we have N installations of the software, and a total of F failures are reportable at time T, the failure rate of the software can be computed as $\lambda = F/(N*T)$. However, once a failure occurs there is additional time lost as the faults causing the failure are located and repaired. Thus it is important to know the mean time to repair for a component that has failed. Combining this time with the mean time to failure tells us how long the system is unavailable for use; the mean time between failures (MTBF) is simply;

$$MTBF = MTTF + MTTR$$

The most important measures of reliability may be the probability of failure on demand. A software reliability growth model can be regarded to be a mathematical expression which fits the experimental data. It may be obtained simply by observing the overall trend of reliability growth. An analytically obtained model has the advantage that its parameters have specific interpretations in terms of the testing process. In this paper we have discussed the exponential approach of growth mode.

Exponential Approach:

Exponential model is a two parameter model. The exponential function follows randomness. Thus, for each i, we can express the distribution function as;

$F_i(t_i) = 1 - e^{-\lambda_i t_i}$

The inference procedure for computing λ_i is to calculate the average of the two previously observed value of t_i that is,

$$\frac{1}{\lambda_i} = \frac{t_{i-2} + t_{i-1}}{2}$$

$$thus, \quad \lambda_i = \frac{2}{t_{i-2} + t_{i-1}}$$

Using this formula the mean time to i^{th} failure by substituting our predicted value of λ_i in the model. The mean time to failure is $1/\lambda_i$, so we have the average of the two previously observed failure times.

Enhanced Exponential Approach:

In this model the current status of failure as calculated using three previous failure information. So that the current failure is not fully based the starting steps of the system. This approach will predict the failure accurately. Basic metrics that will be collected as a part of testing effort during each testing phases are listed in the table;

No. of test cases executed
No. of defects found
No. of defects blocked
No. of defects left open at the time of release of the phase

TOTAL FAT SCRIPTS									
		Executed				Not Executed			
	Total	Passed	Failed Deferred	Failed	Re-Run	Not Completed	Blocked	De-Scoped	No Run
Project Script/ BTC Total	86	28	NA	0	NA	58	0	0	0
Production Regression Script/ BTC Total	0	0	NA	0	NA	0	0	0	0
Total All SCRIPTS/ BTC's	**86**	**28**	**NA**	**0**	**NA**	**58**	**0**	**0**	**0**

The inference procedure to calculate the prediction as follows;

$$\frac{1}{\lambda_i} = \frac{t_{i-3} + t_{i-2} + t_{i-1}}{3}$$

$$thus, \quad \lambda_i = \frac{3}{t_{i-3} + t_{i-2} + t_{i-1}}$$

In the growth model the severity and priority of the defect is not considered. The prediction of the next defect has defined. Once the defect is identified, it be should analyzed, if it is less priority and minimum severity then the next occurrence of the defect can be predicted. In this growth model the next defect is fixed using the previous two defects, it will not adequate for web based applications. Since in web based applications some APIs will executed with different modules and different users. If any defect in the API will affect all the modules and the user result will be inconsistent or otherwise if any defect with the modules which are shared by many users will not immediately failed. So in this type of application the techniques used in growth

model will not predict the failure accurately. By observing the continuous failure we can predict when the next defect will occur.

CONCLUSION

As the cost of software application failures grows and as these failures increasingly impact business performance, software reliability will become progressively more important. Employing effective software reliability engineering techniques to improve product and process reliability would be the industry's best interests as well as major challenges. In this paper we have practiced a new approach of growth model particularly for web application. We have considered very few samples, in future this approach can be practiced for all kinds of applications.

REFERENCES

[1]. ANSI/IEEE, Standard Glossary of Software Engineering Terminology, STD-729-1991, ANSI/IEEE, 1991.

[2]. J.D. Musa, Software Reliability Engineering: More Reliable Software Faster and Cheaper (2nd Edition), AuthorHouse, 2004.

[3]. B. Littlewood and L. Strigini, "Software Reliability and Dependability: A Roadmap," in Proceedings of the 22nd International Conference on Software Engineering (ICSE'2000), Limerick, June 2000, pp. 177-188.

[4]. M.L. Shooman, Reliability of Computer Systems and Networks: Fault Tolerance, Analysis and Design, Wiley, New York, 2002

[5]. M. Chen, M.R. Lyu, and E. Wong, "Effect of Code Coverage on Software Reliability Measurement," IEEE Transactions on Reliability, vol. 50, no. 2, June 2001, pp.165-170.

[6]. Y.K. Malaiya, N. Li, J.M. Bieman, and R. Karcich, "Software Reliability Growth with Test Coverage," IEEE Transactions on Reliability, vol. 51, no. 4, December 2002, pp. 420-426

[7]. M.A. Vouk, "Using Reliability Models During Testing With Nonoperational Profiles," in Proceedings of 2nd Bellcore/Purdue Workshop on Issues in Software Reliability Estimation, October 1992, pp. 103-111.

[8]. Rome Laboratory (RL), Methodology for Software Reliability Prediction and Assessment, Technical Report RL- TR-92-52, volumes 1 and 2, 1992.

[9]. C. Liu, L. Fei, X. Yan, J. Han, and S. Midkiff, "Statistical Debugging: A Hypothesis Testing-based Approach," IEEE Transaction on Software Engineering, vol. 32, no. 10, October, 2006, pp. 831-848.

[10]. A.X. Zheng, M.I. Jordan, B. Libit, M. Naik, and A. Aiken, "Statistical Debugging: Simultaneous Identification of Multiple Bugs," in Proceedings of the 23rd International Conference on Machine Learning, Pittsburgh, PA, 2006, pp. 1105-1112.

