Computer Engineering and Intelligent Systems
ISSN 2222-1719 (Paper) ISSN 2222-2863 (Online)
Vol.5, No.3, 2014

Adaptive Mean Value Function Based Quality Assessment of Software Reliable Growth Models

Chandra Mouli Venkata Srinivas Akana,
Research Scholar, Dept. of CSE, JNTUK, Kakinada, Andhra Pradesh, India. Email: mouliac@yahoo.co.in

Dr. C. Divakar
Principal, Pydah College of Engineering & Technology, Visakhapatnam, AP, India.
Email: divakar_c@yahoo.com

Dr. Ch. Satyanarayana
Professor, Dept of CSE, JNTUK, Kakinada., AP., India. Email: chsatyanarayana@yahoo.com

Abstract:

Software growth models aims for reliability of the application over a period of time. Assessment of such models is of great interest since many faults arises with the models during the operation over a span of time. In this paper a adaptive mean value function based testing and estimation of the parameters were discussed. The proposed approach is also compared against the conventional testing approaches and found that the proposed method able to detect the fault under different scenario and proves to give better performance under a constrained environment.

Keywords: Software reliability, Growth models, testing analysis, Fault detection and correction.

I.INTRODUCTION

IEEE defines the software quality as the degree to which the software possesses a desired combination of attribute [1].ISO defines the so the software quality as: “the totality of features and characteristics of a software product that bear on its ability to satisfy stated or implied needs” [2]. Software quality is described in the means of models which are called software quality models and these have their own quality attributes [3], ISO 9126 defines software quality with six software quality attributes as functionality, reliability, usability, effectiveness, maintainability and portability [2].

Software reliability Engineering (SRE) is the discipline that helps the organizations to improve the quality of their products and processes. The American Institute of Aeronautics and Astronautics (AIAA) defines SRE as "the application of statistical techniques to data collected during system development and operation to specify, predict, estimate, and assess the reliability of software-based systems"[4].

Among the attributes of software quality, reliability is generally accepted as one of the major factor in software quality since it quantifies the failures. There are many reasons why the organizations have to encourage this discipline and promote the usage of software reliable models. Finally we can conclude that a quality software model which depends on the focused software is needed to be successfully applied for different systems. This model attempt to match product properties with the software quality attributes. There are three basic elements as such product properties, quality attributes and linking product properties with quality attributes in this model. Product properties are correctness, internal, contextual and descriptive. Functionality and reliability are the attributes which would contribute to the correctness product property and the attributes of the internal product property are maintainability, efficiency and reliability. Maintainability, re-usability, portability and reliability are the attributes of contextual product property and the attributes which would contribute to descriptive product property are maintainability, re-usability, portability and usability. The mathematical expressions that specify the failure of software process is said to be software reliability estimation models or growth models (SRGMs).

Organizations attain many advantages through SRGM using these developers and customers will have the continuity and determination what they tend to have. When the software system development is done through the agreement between vendor and customer, the reliability objective of the software should be either a pre agreed

one of software quality metrics or it should be as a part of standard practice of the organization. By employing such reliability measures the validation and the quality of the product can be improved. Some of reliability issues during the requirement formulation focus on reducing the erroneous requirements in consideration, accounting of the risk of failure occurrences of each requirement, and the change management issues of future changes of the requirements. Designing and development phase is the most crucial and important phase and needed to be more reliable. Critical operations must be included to improve the quality and availability and the release time can be determined using SR during testing.

This paper is organized as follows

II .SOFTWARE GROWTH MODELS

There are three classes of SRGMs, they are

a. Exponential NHPP models

b. Non-exponential NHPP models

c. Bayesian models

A Poisson probability distribution function takes the form $f(t) = \lambda e^{-\lambda t}$ .The mean time function is $\mu(t) = \lambda$. Homogeneous Poisson Process models assume a constant mean time function while Non-Homogeneous Poisson Process models assume a mean time function to be non-liner.

a) EXPONENTIAL NHPP MODELS

Models in this type are based on shooman's model, musa's basic model, Jelinski and Moranda's model. Below is the probability distribution of these models

Shooman's model

$$f(t) = k[\frac{E_0}{I_T} - \epsilon_c(x)] \qquad (1)$$

Where E_0 is the initial number of faults in the program that will leads to failures

E_c is the number of faults in the program which have been found and corrected

K- is constant of proportionality

Musa's basic model

$$\mu(t) = \beta_0(1 - e^{-\beta_1 \tau}) \qquad (2)$$

Where β is Negative of derivative of failure rate divided by failure rate

Jelinski and Moranda's Model

$$\mu(t) = N(1 - e^{-\phi t}) \qquad (3)$$

Scheneidewind's models

$$\mu(t) = \frac{\alpha}{\beta}(1 - e^{-\beta t}) \qquad (4)$$

b) NON-EXPONENTIAL NHPP MODELS

Duane's Model

$$\mu(t) = \alpha t^{\beta}) \quad (5)$$

Brook and mohey's Poisson models

$$P(x = n_i) = \frac{(N_i\phi_i)^n \, e^{N_i\phi_i}}{n_i!}$$

Yamada's S-Model

$$\mu(t) = N\sum_{i=1}^{k} p_i[1 - e^{-\beta_i t}] \quad (6)$$

Musa and Okumoto model

$$\lambda_0 e^{-\phi\mu}$$

c) BAYESIAN MODELS

Little wood $\lambda_{linear}(t) = \frac{\alpha-1}{\sqrt{\beta_0^2+2\beta_1 t(\alpha-1)}}$

And $\lambda_{quadratic}(t) = \frac{v_1}{\sqrt{t^2+v_2}}\left(\left(t + (t^2+v_2)^{\frac{1}{2}}\right)^{\frac{1}{2}} - (t - (t^2+v_2)^{1/2})^{\frac{1}{2}}\right)$ (7)

The accuracy of the models in the same class is generally the same, as the general reliability function of them is same. Hence, it is enough to argue about the accuracy if at least one model in each class is considered.

III. FAILURE DETECTION MODELS

a) Detection Models

One feature of the failure detection estimation models like the Schneidewind detection model [5], [6] is that it can also model failure detection processes. The Schneidewind detection model is a recommended model among various software reliability estimation models in the IEEE 1633 standard. It is validated based on the failure data of National Aeronautics and Space Administration (NASA) in the U.S. The Schneidewind detection model uses the detected failure counts within the same time interval and calculates the current failure rate based on the historical failure rate to predict future failures accurately. The Schneidewind detection model considers that the failure detection process can be changed when the test is performed and suggests a basic approach as well as two additional approaches considering that recent failure counts are more useful than historical failure counts to predict near-future failures. It is possible to select one approach among the three approaches based on the purpose.

Three approaches of the Schneidewind model are as follows.

- Approach 1: use all of the failure counts from interval 1 through t (i.e., s = 1).
- Approach 2: use failure counts only in intervals s through t (i.e., $1 \leq s \leq t$).
- Approach 3: use cumulative failure counts in intervals 1 through s-1 and individual failure counts in intervals s through t (i.e., $2 \leq s \leq t$).

In order to use this model, it is necessary to follow the next process shown below.

• Assumptions for data collection
  - ▪ Perfect debugging
  - ▪ Removal time is ignored

• Data collection
  - ▪ Detected failures

- Test time

• Estimation of parameters for the mean value function
  - Parameter for the total failure counts
  - Parameter from the failure occurrence rate

• Reliability validation
  - Estimation of failure counts that is undetected
  - Decision of the time point for test or release

b) Failure removal estimation model

The failure removal estimation model is modeled from the failure removal process and the Schneidewind removal model and JungHua's removal model are representative models among failure removal estimation models [9, 10]. The Schneidewind removal model was developed from a modification of the basic Schneidewind detection model because the basic Schneidewind model has the unrealistic limitation in which the "removal time is ignored". Therefore, a delay time is introduced; this is the time between failure detection and removal. JungHua's removal model was developed from the basic Goel-Okumoto model. It considers the delay time and the failure correction rate. JungHua's removal model shows the failure removal process and considers the failure detection process and the removal process. According to these two functions, failure detection and correction processes can be estimated.

To use these models, it is necessary to follow the process shown below.

• Assumptions for data collection
  - Perfect debugging
  - Delay time occurs

• Data collection
  - Detected failures and removed failures
  - Test time and delay time

• Estimation of parameters for the mean value function
  - Parameter for total failure counts
  - Parameter from the failure occurrence rate and failure correction rate

• Reliability validation
  - Estimation of failure counts removed
  - Remaining uncorrected failure counts and time point of all detected failures removed

IV PROPOSED MODEL

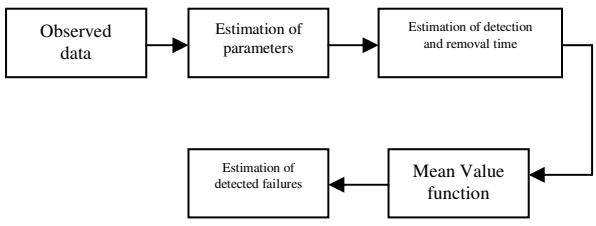

Figure 1: Block diagram of the proposed model

Data Collection

To conduct an experiment of the proposed model, actual data collected from an organization was used instead of hypothetical data. The actual data was collected from the in-progress project of a CMMI level

5. To collect the actual data, the two data collection templates were used. Developers collected the actual data using forms and sent them to us.

Estimation of parameters

$$\alpha = \frac{\sum_{i=1}^{n} f_i}{1-e^{-\beta t_k}} = \sum_{i=1}^{n} \frac{f_i (t_i e^{-\beta t_{i-} t_{i-1} e^{-\beta t_{i-1}}})}{e^{-\beta t_i - 1 - e^{-\beta t_i}}} \qquad (8)$$

Where α is the number of estimated failures of the Goel –Okumoto model and α_p is the number of estimated total failures of the proposed model. β is the failure occurrence rate of the Goel-Okumotp model, t is the detection time and t_p is the total failure time MVF is the cumulative number of detected failures between 0 and time t .

Mean Value function

$$mvf = \alpha_p [1 - e^{-\beta_p t_p}] \qquad (9)$$

To compare estimation results to actual data, Mean Relative Error (MRE) and Mean Square Error (MSE) are used

V.EXPERIMENTAL RESULTS

To conduct the experiment, the collected failure data of a unit among 10 available units was used. The experiment produced several graphs of the results from the Goel-Okumoto and from the proposed model for validation. Shown first are the MVF result of the Goel-Okumoto model,

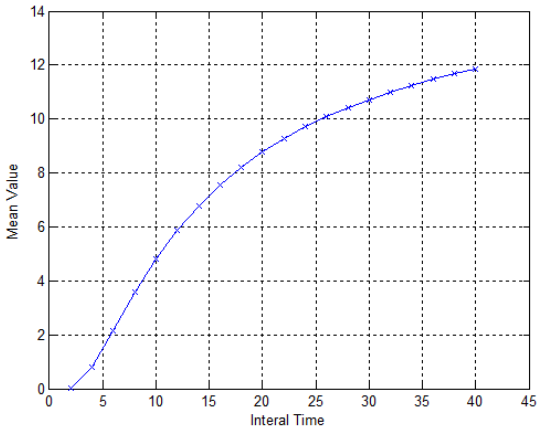

Figure 2: Estimation mean value function of Goel-Okumoto model

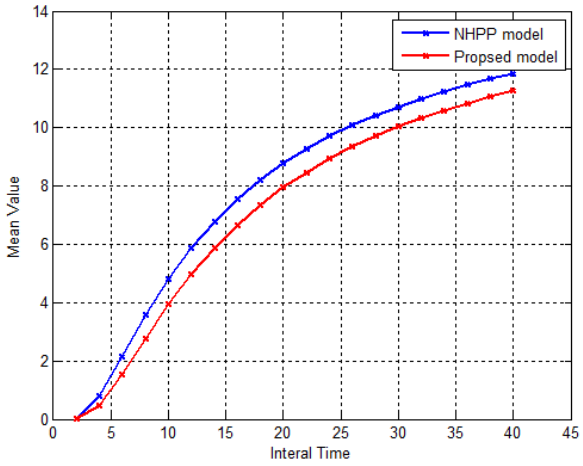

Figure 3: Estimation of mean value for the proposed model

Developers essentially spend a certain amount of time finding failures. They also spend a certain amount of time to remove faults that are the causes of these failures. In this experiment developers spent 120 to remove all of the faults. Therefore, the proposed model considers the removal time to reflect delay, as shown in the figure 3. Developers essentially spend a certain amount of time finding failures. They also spend a certain amount of time to remove faults that are the causes of these failures.

Table I. Comparison between the G-O model and the Proposed

Parameter	Goel-Okumoto	Proposed
Time	1250	1250
Detection rate α	12.34	12.34
β	0.048	0.043
MRE	0.128	0.095
MSE	0.597	0.29
Total Intervals	38	38
Failures	10.56	10.1

VI. CONCLUSION

A new reliability estimation model was developed to consider the characteristics of the early test phases. Current existing reliability estimation models are normally used during the late test phases, which typically include system testing and operational testing. Therefore, the current existing reliability estimation models can be divided into failure detection estimation models and failure removal estimation models. Failure detection estimation models consider the test time to estimate future failure trends using the detected failure counts per time interval. Failure removal estimation models estimate that future failure will be removed using the removed failures detected per time interval. These models do not consider that developers perform testing and debugging activities. Currently, the proposed model is based on Exponential models. Therefore, data that is fitted to S-shaped [9] models cannot be used with the proposed model. This is a limitation of the proposed model

REFERENCES

[1] IEEE Std 610.12 (1990). IEEE Standard Glossary of Software Engineering Terminology, NY.
[2] ISO/IEC Std 9126-1," Software Engineering – Product Quality ", Part 1: Quality Models, International Organization for Standards 2001.
[3] McCall J.A, Richards P.K & Walters G.F," Factors in Software Quality", Nat'1 Tech Information Services, -Vol 1-2, 1977
[4] AIAA/ANSI "Recommended Practice for Software Reliability, The American Institute of Aeronautics and Astronautics", Washington DC, Aerospace Center, R-013, 1992,-ISBN 1-56347-024-1.

[5] "IEEE Recommend Practice on Software Reliability", IEEE Reliability Society, June, 2008.
[6] Norman F. Schneidewind, "Reliability Modeling for Safety-Critical Software", IEEE Transactions on Reliability, March, 1997.
[7] Norman F. Schneidewind, "Modeling the Fault Correction Process", 12th International Symposium on software Reliability, November, 2001.
[8] Jung-Hua Lo, Chin-Yu Huang, "An Integration of Fault Detection and Correction Processes in Software Reliability Analysis", The Journal of Systems and Software, 2006.
[9] S. Yamada, M. Ohba, O. Osaki, "S-Shaped Reliability Growth Modeling for Software Error Detection, IEEE Transactions on Reliability", Vol, R-32, no. 5475-5478, 1983.
[10] William Farr, Oliver Smith, "SMERFS-Statistical Modeling and Estimation of Reliability Functions for Systems", 1996.