RESEARCH ARTICLE OPEN ACCESS

SRGM Analyzers Tool of SDLC for Software Improving Quality

Mr. Girish Nille, Prof. Bharat Tidake

Abstract—
Software Reliability Growth Models (SRGM) have been developed to estimate software reliability measures such as software failure rate, number of remaining faults and software reliability. In this paper, the software analyzers tool proposed for deriving several software reliability growth models based on Enhanced Non-homogeneous Poisson Process (ENHPP) in the presence of imperfect debugging and error generation. The proposed models are initially formulated for the case when there is no differentiation between failure observation and fault removal testing processes and then this extended for the case when there is a clear differentiation between failure observation and fault removal testing processes. Many Software Reliability Growth Models (SRGM) have been developed to describe software failures as a random process and can be used to measure the development status during testing. With SRGM software consultants can easily measure (or evaluate) the software reliability (or quality) and plot software reliability growth charts.

Keywords— Software Reliability Growth Model (SRGM), Testing Coverage (TC), Testing Time, Cobb Douglas Production Function, ENHPP (Enhanced Non-Homogenous Poisson Process), Fault Detection Rate (FDR).

I. INTRODUCTION

The concern for software reliability has grown over a period of time especially with the advent of real life systems such as satellite and shuttle control, telephone, internet and banking services. With the growing economy there has been a growing interest of companies to know about their competitors and have been spending a lot on strategic decision making. All these activities require complex software systems. It is important that these systems are thoroughly tested before implementation. There is a huge cost attached with fixing failures, safety concerns, and legal liabilities therefore organizations need to produce software that is reliable. There are several methodologies to develop software but questions that need to be addressed are how many times will the software fail and when, how to estimate testing coverage, when to stop testing, and when to release the software. Also, for a software product there is need to predict/estimate the maintenance effort; for example, how long the warranty period must be, once the software is released, how many defects can be expected at what severity levels, how many engineers are required to support the product, for how long, and so forth[1].

Software reliability assessment is an important issue for planning release of high-quality software products to users. Many developers have proposed software reliability growth models (SRGMs) to assess software quantitatively from fault data observed in software testing phase [1-2, 3-4]. Software reliability is defined as the probability of failure free software operation for a specified period of time in a specified environment. It is used to assess the reliability of the software during testing and operational phases. Software testing involves running the software and checking for unexpected behavior in software output. The successful test can be considered to be one, which reveals the presence of the latent faults. The process of locating the faults and designing the procedures to detect them was called the debugging process. The chronology of failure occurrence and fault detections can be utilized to provide an estimate of the software reliability and the level of fault content. In particular, the Enhanced non-homogeneous Poisson process (ENHPP) [10] based SRGMs are quite popular due to their mathematical tractability, and there have been a number of ENHPP-based SRGMs proposed by many Developers. In spite of the diversity and elegance of many of these, no single model can be readily recommended as best to represent the challenging nature of the software testing[9,7].

As reliability is of great concern for software products this field is catching attention of various researchers and practitioners. This has lead to a new concept Software Reliability Growth Modeling. An SRGM is defined as a tool that can be used to evaluate the software quantitatively, develop test status, schedule status, and monitor the changes in reliability performance. Software reliability assessment and prediction is important to evaluate the performance of software system. The reliability of the software is quantified in terms of the estimated number of faults remaining in the software system. During the testing phase, the emphasis is on reducing the remaining fault content hence increasing the software reliability. The SRGMs developed in literature are either dependent on testing time, testing effort or testing coverage. Testing time is the calendar time or the CPU time whereas testing effort takes into account the manpower time and the CPU time. The papers discuss some novel software reliability growth models dependent on time. Testing Coverage plays a very important role in predicting the software reliability. Testing Coverage is actually a structural testing technique in which the software performance is judged with respect to specification of the source

code and the extent or the degree to which software is executed by the test cases. TC can help software developers to evaluate the quality of the tested software and determine how much additional effort is needed to improve the reliability of the software besides providing customers with a quantitative confidence criterion while planning to use a software product. Hence, safety critical system has a high coverage objective.

II. LITERATURE SURVEY

Various investigative programming dependability shows have been proposed for evaluating the unwavering quality development of a programming item. The paper present an Enhanced non-homogeneous Poisson process (ENHPP) model [10] and show that awhile ago reported Non-homogeneous Poisson process (NHPP) based models, with limited mean esteem capacities, are extraordinary instances of the (ENHPP) model. The (ENHPP) model contrasts from past models in that it consolidates unequivocally the time differing test scope capacity in its analytical definition, and accommodates damaged blame discovery in the testing stage and test scope throughout the testing and operational stages . The (ENHPP) model is validated utilizing a few accessible inadequacy information sets.

Many Developers have incorporated change point in software reliability growth modeling. Firstly Zhao [5] incorporated change-point in software and hardware reliability. Huang et al. [4] used change-point in software reliability growth modeling with testing effort functions. The imperfect debugging with change-point has been introduced in software reliability growth modeling by Shyur. Kapur et al. [6, 8] introduced various testing effort functions and testing effort control with change-point in software reliability growth modeling. The multiple change-points in software reliability growth modeling for fielded software have been proposed by Kapur et al. [6, 7]. A testing coverage based SRGM was proposed by Malaiya [6]. Inoue and Yamada [5] developed SRGM to describe a time-dependent behavior of a testing-coverage attainment process with the testing-skill of test-case designer.

III. STATEMENT OF AIM AND OBJECTIVE

Aim of this model is achieve the reliability of the software and Increases role of software in real life systems. Objectives are to manage and improve:

- The reliability of the software
- Check the efficiency of development activities
- Evaluate the software reliability at the end of validation activities and in operation
- Estimate the maintenance effort to "correct" faults activated during development and residual faults in operation.

IV. PROPOSED SYSTEM MODEL

The paper has propose a generalized framework for deriving several testing time and testing coverage depends on Enhanced Non-Homogeneous Poisson Process software reliability growth model which incorporate Change point. Change point is one of the interesting phenomenons on observed during software development. Inclusion of change point in software reliability growth modeling enhances the predictive accuracy of the model.

The proposed models are based upon the following basic assumptions.

- The failure observation and fault removal phenomenon is modeled using an ENHPP [10].
- Software is subject to failures during execution caused by faults remaining in the software.
- Each time a failure is observed, an immediate debugging effort takes place to find the cause of the failure to remove it.
- The failure rate is equally affected by all the faults remaining in the software.
- When a software failure occurs, an instantaneous repair effort starts, and then either (a) the fault content is reduced by one with probability , or (b) the fault content remains unchanged with probability .
- During the fault removal process, whether the fault is removed successfully or not, new faults are generated with a constant probability.

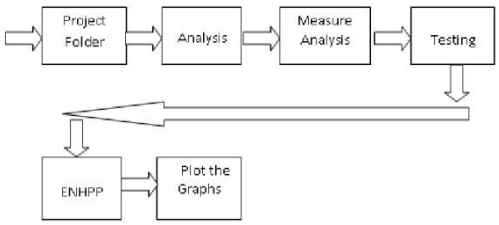

Fig. 1 ENHPP Model

A] ENHPP Model Phases
i] Phases of Analyzer:

1. Identifying Phase [11]
This phase consist identifying software such as TNOA, WMPC, NOCP etc. as shown below.
TNOC: It counts total of attributes for a class.
WMPC: It counts all class methods per class.

NOCP: It Count the Number of classes in a package.
MIT: It calculates the longest path from the class to the root of the inheritance tree.
CCIM: It measures the number of linearly independent paths through a program module .i.e. the amount of decision logic in a single software factor. It is calculated per class.
SIZE: It refers to the count total lines of code.

2. Evaluation Phase [11]
This phase consists of evaluating the various software measure written as a part of "Analyzer" tool development which are provided for examination

3. Interpretation and Advising Phase [11]
The interpretation part of the measure Analyzer takes the extracted measure values in measure Evaluation Phase from the source code and compares these values with the threshold values of the corresponding metric values. If the extracted metric values lies below the corresponding metric threshold value means that no occurrence of the issue that is being observed and extracted measure values lies above the corresponding metric threshold value means maximum occurrence of the observed issue.

4. Application Phase [11]
In this phase, software refactoring is done .Our tool "measure Analyzer" applies the Object Oriented metrics on the code base and these metric values are then interpreted. Then various refactoring techniques were used to improve the code design and along with that paper has also studied the impact of refactoring on the software quality through various measures.

V. METHODOLOGIES AND TECHNIQUES

ENHPP (Enhanced Non-Homogeneous Poisson Process) model – [10]

The ENHPP model provides a unifying framework for finite failure software reliability growth models According to this model, the expected number of faults detected by time t, called the mean value function, n (t) is of the form:

$$n(t) = b * c(t)$$

Where b is the needed number of shortcomings in the programming (before testing/debugging starts), and c (t) is the scope capacity. The ENHPP model utilized by SREPT gives by default to reflect four sorts of inadequacy event rates for every fault. Inter flop times information acquired from the testing stage could be utilized to parameterize the ENHPP (Enhanced Non Homogeneous Poisson Process) model to acquire gauges of the inadequacy force, number of flaws remaining, dependability after discharge, and scope for the programming. When complexity metrics are available, the total number of faults in the software can be estimated using the fault density approach or the regression tree model. If the number of lines of code in the software is NL, the expected number of faults can be estimated as, [4]:

$$F = NL * FD$$

The relapse tree model is an objective arranged statistical method, which endeavors to foresee the amount of deficiencies in a programming module dependent upon the static unpredictability measurements. The structure of the ENHPP model might additionally be utilized to join the appraisal of the aggregate number of issues acquired after the testing stage dependent upon unpredictability measurements (parameter a), with the scope informative content got throughout the testing stage (c (t)). The ENHPP model [10] can also use inter failure times from the testing phase to obtain release times (optimal time to stop testing) for the software on the basis of a specified release criteria.

Release criteria [10] could be of the following types –

Number of remaining faults - In this case, the release time is when a fraction of all detectable faults has been removed.
Failure intensity requirements - The criterion based on failure intensity suggests that the software should be released when the failure intensity measured at the end of the development test phase reaches a specified value f.
Reliability requirements - This criteria could be used to specify that the required conditional reliability in the operational phase is, say Rr at time t0 after product release.
Cost requirements - From a knowledge of the expected cost of removing a fault during testing, the expected cost of removing a fault during operation, and the expected cost of software testing per unit time, the total cost can be estimated. The release time is obtained by determining the time that minimizes this total cost.
Availability requirements - The release time can be estimated based on an operational availability requirement [2].

The Cobb-Douglas [10] employs production function to demonstrate the effect of both testing time and testing coverage in removing the faults in the software.

The SRGMs developed in literature are either dependent on testing time, testing effort or testing coverage. Testing time is the calendar time or the CPU time whereas testing effort takes into account the manpower time and the CPU time.

The mathematical form of the production function is follows: $Y = AL^{\beta}K^{\alpha}$

Where:

- Y = total production (the monetary value of all goods produced in a year)

- *L* = labor input (the total number of person-hours worked in a year)
- *K* = capital input (the monetary worth of all machinery, equipment, and buildings)
- *A* = total factor productivity
- α and β are the output elasticity's of capital and labor, respectively. These values are constants determined by available technology.

SRGMs have been proposed to measure the reliability during the testing phase. Most of these can be categorized under Non Homogeneous Poisson Process (NHPP) model [10].

The SRGM based on NHPP is formulated as:

There is rate function is λ(t) means rate parameter may change over time and expected number of events between time a and time b is

$$\lambda_{a,b} = \int_a^b \lambda(t)\, dt.$$

The number of the events or arrivals in the time interval (a, b], given as N (b)-N (a). Follow a poisson distribution associated with parameter λa,b

$$P[(N(b) - N(a)) = k] = \frac{e^{-\lambda_{a,b}}(\lambda_{a,b})^k}{k!} \qquad k = 0, 1, \ldots.$$

P [(N (b)-N (a)) =k] is number of events in time interval (a, b]

CONCLUSION

This paper has proposed SRGM in testing phase of SDLC to ensure the most reliable software and quality. The paper have to develop ENHPP (Enhanced Non-Homogeneous Poisson Process) model to find out errors and faults in the current and existing system .It will improve the quality and reliability of the software and moreover, it will eliminate the errors and faults in the current and existing system. Therefore the SRGM testing will become more authentic

REFERENCES

[1] A Detailed Study of Nhpp Software Reliability Models. Journal Of Software, Vol. 7, No. 6, June 2012.

[2] Two Dimensional Flexible Software Reliability Growth Model With Two Types Of Imperfect Debugging. P.K. Kapur1, Anu G. Aggarwal And Abhishek Tandon3. Department Of Operational Research, University Of Delhi.

[3] Analysis of Discrete Software Reliability Models- Technical Report (Radctr- 80-84" 1980; New York: Rome Air Development Center.

[4] Comparing Various Sdlc Models and The New Proposed Model On The Basis Of Available Methodology Vishwas Massey, Prof. K.J.Satao.

[5] Inoue S, Yamada S "Testing-Coverage Dependent Software Reliability Growth Modeling" International Journal Of Reliability, Quality And Safety Engineering,Vol. 11, No. 4, 303-312, 2004.

[6] S.Saira Thabasum, "Need For Design Patterns And Frameworks For Quality Software Development", International Journal Of Computer Engineering & Technology (Ijcet), Volume 3, Issue 1, 2012, Pp. 54 - 58, Issn Print: 0976 – 6367, Issn Online: 0976 – 6375.

[7] S.Manivannan And Dr.S.Balasubramanian, "Software Process And Product Quality Assurance In It Organizations", International Journal Of Computer Engineering & Technology (Ijcet), Volume 1, Issue 1, 2010, Pp. 147 - 157, Issn Print: 0976 – 6367, Issn Online: 0976 – 6375.

[8] Software Reliability Growth Model Based On Fuzzy Wavelet Neural Network- *2010 2nd* International Conference On Future Computer And Communication.

[9] International Journal Of Computer Engineering And Technology (Ijcet), ISSN 0976-6367(Print), ISSN 0976 – 6375(Online) Volume 4, Issue 2, March – April (2012), © Iaeme.

[10] Unification of Finite Failure Non-Homogeneous Poisson Process Models through Test Coverage* Swapna S. Gokhale, Teebu Philip, Peter N. Marinos, Kishor S. Trivedi, Center for Advanced Computing and Communication Department of Electrical and Computer Engineering Duke University Durham, NC 27708-0291

[11] International Journal of Software Engineering & Applications (IJSEA), Vol.3, No.4, July 2012 DOI: 10.5121/ijsea.2012.3402 13 Effective Implementation Of Agile Practices – Object Oriented Metrics Too To Improve Software Quality Veerapaneni Esther Jyothi, Kaitepalli Srikanth and K. Nageswara Rao