Model for Checking Reliability of Software with Multiple Releases of Software

Miss Nehal S. Morshe Dr. V. M. Thakare Prof. M. Atique

Abstract **:- As the software field is increasing rapidly. Almost all the products are depends on the software for the rapid and inaccurate results. As the functionality of computer operations becomes more complicated, and critical software applications increase in complexity, the reliability of computer software becomes more important, and errors in software design become more subtle. So the proposed model first selects the optimal software reliability growth model using distance based approach and then applies the selected model to the multiple releases of the software at the same time the Stastical Model Checking framework is also applied to each release of the software to prevent the further propagation of errors. This model can be used in many types of software including the safety critical systems.**

Keywords **: Software Reliability, Statistical Model Checking (SMC), Non-Homogeneous Poisson Process (NHPP), Distance Based Approach (DBA).**

I. INTRODUCTION

Nowadays the software field is increasing rapidly. Almost all the products are depends on the software for the rapid and inaccurate results. Software is embedded in many modern systems, including expensive scientific computing systems, financial banking systems, industrial applications, university computer centers, and home personal computers. Since the demands for complex and large-scale software systems are increasing more rapidly, the possibility of programmers' design errors in the systems will grow appreciably. Consequently, the possibility of crises due to software failures will continue to increase. These failures can generate enormous losses of revenue for many enterprises. Therefore, to determine system reliability, the software reliability must be carefully evaluated. Software development is a time consuming, costly process with high quality targets. The most effective way of handling software development is to develop software in phases, and to offer the complete functionality in multiple releases. These advantages include fast deliveries, early revenue generation, and increased market life of the product. As the functionality of computer operations becomes more complicated, and critical software applications increase in complexity, the reliability of computer software becomes more important, and errors in software design become more subtle.

Testing software to estimate reliability is most often done late in the product development cycle, when functionality is in place, and the application is relatively stable. It is at this phase when decision makers are most concerned about estimating field reliability. Understandably, such an estimate greatly impacts the decision to release. Consequently, it has become important to implement and acquire highly reliable software and to satisfy the safety requirements imposed by functional-safety standards.

II. BACKGROUND

The measurement of software reliability has become increasingly important because of both the impact of Software failures and the development costs of reducing them. To achieve highly dependable software, various approaches based on techniques of fault prevention, fault removal, fault tolerance, and fault forecasting have been proposed [1].The Dependent Model for Fault Tolerant Software Systems during Debugging is a special redundant model for describing the s-dependency of multi-version programming software during testing and debugging phases of the software development life cycle.

During the past few decades, the proportion of software in safety critical systems has significantly increased. So, to ensure high-level safety, it's essential to improve software reliability. Consequently, it has become important to implement and acquire highly reliable software and to satisfy the safety requirements imposed by functional-safety standards, such as IEC 61508 and ISO 26262. These standards define *safety integrity level* (SIL) and automobile SIL (ASIL) as measures of a system's quality or dependability. To develop a highly reliable software-intensive system, developers allocate a reliability goal for a target system according to a target SIL or ASIL after hazard analysis and risk assessment. To validate the reliability goals of safety-critical systems at an early stage the *statistical model checking* (SMC) framework is used [2]. The stastical model checking framework extends IEEE Standard 1633, which covers software reliability practices.

Under the problems of schedule pressure and resource limitations, developing reliable software is of great concern for the software development firms. SRGMs, like any other mathematical model, are simplifications of reality. Various significant metrics, such as initial number of faults, failure intensity, reliability within a specific period of time, number of faults remaining, can be effortlessly determined through SRGMs. During the last three decades, a large number of SRGMs have been proposed one of which is 2 dimensional multi release SRGM (2-D; M-R SRGM). The 2-D M-R SRGM's incorporate the mutual effect of testing time and resources for multiple releases of the software [3]. For modeling the reliability growth for multiple releases, it considers the failures of the upcoming upgraded release, plus the failures that were not debugged in the previous release, because the testing of software cannot continue indefinitely. The release time problem was also studied under the assumption that software comes in a single release. The optimal release planning problem was formulated for software with multiple releases. The problem not only determines optimal time but also the optimal amount of resources [3]. The optimal release planning problem is a complex, non linear

optimization problem, and is solved using a genetic algorithm (GA). The GA is a powerful tool for solving search and optimization problems.

Many SRGM's are developed over the past 30 years for estimating reliability growth of products during the software development process. Each model could be shown to work well with a unique data set, but no model appeared to do well on all data sets. The distance based approach was proposed which is a simple objective method to select the best model. And this model was used to do well on all the data sets. The distance based approach was proposed for the for selection and ranking of sixteen different NHPP (non-homogeneous Poisson process) software reliability growth models based on a set of twelve contributing model selection criteria like Bias, MSE, MAE, MEOP, AE, Noise, PRR,Variance, RMSPE, Rsq, SSE, and TS [4].

The rest of the paper is organized as follows: **Section III** discusses work done on various methodologies. **Section IV** describes existing methodologies. **Section V** discusses attributes and parameters and how they affect the result. **Section VI** describes the proposed methodology. **Section VII** discusses possible outcomes and results. Finally **section VIII** Conclude this paper.

III. PREVIOUS WORK DONE

To achieve highly dependable software, various approaches based on techniques of fault prevention, fault removal, fault tolerance, and fault forecasting have been proposed. Fault prevention avoids fault occurrence by using requirements analysis and design methodologies. Fault removal finds and removes errors by design review, code inspection, testing, and verification during the development phase. Fault forecasting predicts fault content and reliability to determine when to stop testing, and how much more resources are required to achieve the objective reliability level. Fault tolerance provides defensive design and programming based on the concept of design diversity. In fault forecasting, software reliability models view debugging as a time oriented stochastic process [1]. NVP is defined as the logically independent generation of multiple software versions from a common specification. Some SRGMs consider the effect of imperfect debugging, the unsuccessful removal and introduction of new faults, during testing and debugging phases. In such a case, software reliability might grow only if a good debugging effort is executed. The fault tolerant model is a new approach for describing the s–dependency of 2 VP software during testing and debugging phases of the software development life cycle. [1].

During the past few decades, the proportion of software in safety critical systems has significantly increased. So, to ensure high-level safety, it's essential to improve software reliability. Consequently, it has become important to implement and acquire highly reliable software and to satisfy the safety requirements imposed by functional-safety standards, such as IEC 61508 and ISO 26262. These standards define *safety integrity level* (SIL) and automobile SIL (ASIL) as measures of a system's quality or dependability [2]. Validating Software Reliability Early through Statistical Model Checking employs statistical model checking to validate software reliability at an early stage. This can prevent the propagation of reliability allocation errors and design errors at later stages. Validating Software Reliability Early through Statistical Model Checking achieves safer, cheaper, and faster development of safety-critical systems [2].

Development of the software process for a firm is dependent on two major assets: time, and tangible resources. Two dimensional models are used to capture the joint effect of testing time and testing resources on the number of faults removed in the software. A two-dimensional multi release model incorporates the combined effect of testing time and resources in each release to remove the faults lying dormant in the software [3].

Over the past 30 years, many SRGMs have been proposed for estimating reliability growth of products during the software development process. Each model could be shown to work well with a unique data set, but no model appeared to do well on all data sets. The distance based approach is a simple objective method to select the best model. And this model is used to do well on all the data sets [4].

IV. EXISTING METHODOLOGY

A Dependent Model for Fault Tolerant Software Systems is a new approach for describing the s–dependency of 2 VP software during testing and debugging phases of the software development life cycle [1]. In NVP systems, s-dependency should not be ignored to avoid the overestimation of system reliability. By assuming positive s-dependency, the model is described by a bivariate counting process. This model emphasizes the evolution of the remaining number of faults for both versions, and thus considers the effect of imperfect debugging, unsuccessful removal and introduction of new faults. Dependent Model for Fault Tolerant Software Systems is used in many mission critical applications, such as air-traffic control, and nuclear reactor control software systems, where ultrahigh reliability is required.

During the past few decades, the proportion of software in safety critical systems has significantly increased. So, to ensure high-level safety, it's essential to improve software reliability. Consequently, it has become important to implement and acquire highly reliable software and to satisfy the safety requirements imposed by functional-safety standards. Statistical Model Checker validate the reliability goals of safety-critical systems at an early stage by using statistical model checking(SMC) to obtain safety certification [2]. To develop a highly reliable software-intensive system, developers allocate a reliability goal for a target system according to a target SIL or ASIL after hazard analysis and risk assessment. Then, they allocate reliability goals to each software component early in the life cycle. Each component's reliability goal is usually validated through failure detection during software testing, which can result in high costs to correct defects. SMC validates a target system's reliability by computing the probabilities that an executable model of a target system satisfies given functional-safety requirements. Validating Software Reliability Early through Statistical Model Checking achieves safer, cheaper, and faster development of safety-critical systems. The Statistical Model Checking framework employs the following process.

- Specify the Functional-Safety Requirement
- Allocate the Reliability Requirement
- Validate the Reliability Requirement
- Validate the Reliability Goal
- Continue Validation or Reallocate

Under the problems of schedule pressure and resource limitations, developing reliable software is of great concern for the software development firms. During the last three decades, a large number of SRGMs have been proposed. The 2-D; M-R SRGM's is one of the model which incorporate the mutual effect of testing time and resources for multiple releases of the software [3]. For modeling the reliability growth for multiple releases, it considers the failures of the upcoming upgraded release, plus the failures that were not debugged in the previous release, because the testing of software cannot continue indefinitely. The release time problem was studied under the assumption that software comes in a single release. Here the optimal release planning problem is formulated for software with multiple releases [3]. The problem determines optimal time and optimal amount of resources. The optimal release planning problem is a complex, non linear optimization problem, and is solved using a genetic algorithm (GA).

A large number of SRGM's have been proposed during the past 30 years to estimate software reliability measures such as the number of remaining faults, software failure rate, and software reliability. Selection of an optimal SRGM for use in a particular case has been an area of interest for researchers in the field of software reliability. A deterministic quantitative model based on a distance based approach (DBA) method is developed, which is then applied for evaluation, optimal selection, and ranking of SRGMs [4]. DBA recognizes the need for relative importance of criteria for a given application, without which inter-criterion comparison could not be accomplished. It requires a set of model selection criteria, along with a set of SRGMs, and their level of criteria for optimal selection; and it successfully presents the results in terms of a merit value which is used to rank the SRGMs. The distance based approach is a simple objective method to select the best model which can be used to do well on all the data sets. The distance based approach is proposed for the for selection and ranking of sixteen different NHPP software reliability growth models based on a set of twelve contributing model selection criteria like Bias, MSE, MAE, MEOP, AE, Noise, PRR,Variance, RMSPE, Rsq, SSE, and TS.

V. ANALYSIS AND DISCUSSION

The dependent model for fault for Fault Tolerant Software Systems during Debugging is used to achieve highly dependable software, various approaches based on techniques of fault prevention; fault removal, fault tolerance, and fault forecasting are considered. This model accommodates the situation when correlated failures occur at different execution time points. It also considers the effect of imperfect debugging. This model also enables to assess the system reliability of 2 VP software without setting the limitations that simultaneous failures must be caused by related faults, and concurrent independent faults must be ignored. This model also accommodates the situation when correlated failures occur at different execution time points.

In the safety critical systems high level of safety is required therefore it is essential to improve the software reliability. Statistical Model Checker validate the reliability goals of safety-critical systems at an early stage by using *statistical model checking* (SMC) to obtain safety certification. Validating Software Reliability Early through Statistical Model Checking achieves safer, cheaper, and faster development of safety-critical systems. Validating software early through SMC can prevent the propagation of reliability allocation errors and design errors at later stages. But the SMC technique uses BIET technique which takes longer verification time.

The 2-D; M-R SRGM's incorporate the mutual effect of testing time and resources for multiple releases of the software. For modeling the reliability growth for multiple releases, it considers the failures of the upcoming upgraded release, plus the failures that were not debugged in the previous release, because the testing of software cannot continue indefinitely. The release time problem was studied under the assumption that software comes in a single release. Here the optimal release planning problem is formulated for software with multiple releases. The problem not only determines optimal time but also the optimal amount of resources. The optimal release planning problem is solved using a genetic algorithm (GA). The GA is a powerful tool for solving search and optimization problems [6].

The distance based approach is proposed for the for selection and ranking of sixteen different NHPP software reliability growth models based on a set of twelve contributing model selection criteria. The DBA is suitable for ranking SRGMs based on a number of conflicting criteria taken all together, and to perform sensitivity analysis tests to identify the most, and least dominating criteria. The DBA method uses a relatively simple mathematical formulation, and straight forward matrix operation. It is capable of solving complex multi-attributes decision problems, incorporating both quantitative and qualitative factors. But this model is applicable for single version software.

VI. PROPOSED METHODOLOGY

In proposed methodology firstly apply the distance based approach for selection and ranking of sixteen different NHPP software reliability growth models based on a set of twelve contributing model selection criteria. The twelve model selection criteria's are Bias, MSE, MAE, MEOP, AE, Noise, PRR, Variance, RMSPE, Rsq, SSE, and TS. After the selection of best model for the prediction of software reliability apply that model to the multiple releases of the software so that the remaining faults in the previous versions can be detected and helps the others teams to correct that faults. This model also applies the stastical model checking framework (SMC) at an early stage of the each releases of the software so that the reliability allocation errors and design errors can be prevented from propagation to later stages.

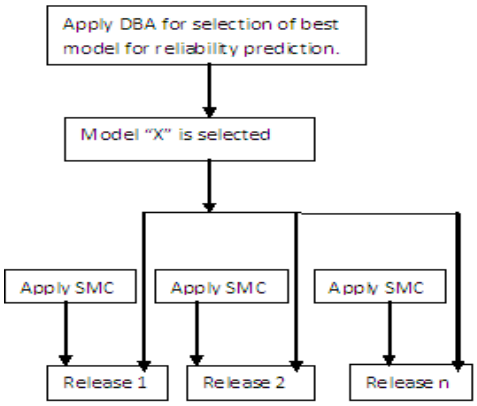

Fig.1: Block Diagram of Proposed Methodology

VII. POSSIBLE OUTCOMES AND RESULTS

The goal of this model is to predict the reliability of the multiple releases of the software. This model first selects the best model from the 16 NHPP models based on some criteria's and after selection of best model that model is applied to the multiple releases of the software to be checked. And also the stastical model checking framework is applied to each release of the software to prevent the further propagation of the design and other errors. This model can be used to predict the reliability of the given software in efficient manner. This model can be mainly used in safety critical software.

VIII. CONCLUSION

As the software field is increasing rapidly. Almost all the products are depends on the software for the rapid and inaccurate results. The most effective way of handling software development is to develop software in phases, and to offer the complete functionality in multiple releases. The development of software in multiple releases provides various advantages to the developing firms. These advantages include fast deliveries, early revenue generation, and increased market life of the product. As the functionality of computer operations becomes more complicated, and critical software applications increase in complexity, the reliability of computer software becomes more important, and errors in software design become more subtle. So the proposed model first selects the optimal software reliability growth model using distance based approach and then applies the selected model to the multiple releases of the software at the same time the Stastical Model Checking framework is also applied to each release of the software to prevent the further propagation of errors. This model can be used in many types of software including the safety critical systems.

IX. FUTURE SCOPE

In the future, this model may incorporate the effect of imperfect debugging in a multi release modeling framework. And it will also consider other testing factors like testing coverage, number of executed test cases, and others may also be studied.

REFERENCES

[1]. Rong-Tsorng Wang, "Dependent Model for Fault Tolerant Software Systems during Debugging" *IEEE TRANSACTIONS ON RELIABILITY, VOL.* 61, NO. 2, PP 504-515, JUNE 2012.

[2]. Youngjoo Kim, S-Core, Okjoo Choi, Moonzoo Kim, and JongmoonBaik, Tai-Hyo Kim, "Validating Software Reliability Early through Statistical Model Checking", *IEEE Software,*PP35 – 41, May/June 2013.

[3]. P. K. Kapur, H. Pham, Anu G. Aggarwal, and Gurjeet Kaur "Two Dimensional Multi-Release Software Reliability Modeling and Optimal Release Planning", *IEEE TRANSACTIONS ON RELIABILITY*, VOL. 61, NO. 3, PP 758 – 768, SEPTEMBER 2012.

[4]. Kapil Sharma, Rakesh Garg, C. K. Nagpal, and R. K. Garg, "Selection of Optimal Software Reliability Growth Models Using a Distance Based Approach" *IEEE TRANSACTIONS ON RELIABILITY*, VOL. 59, NO. 2, PP 266-276, JUNE 2010.