Proceedings of the National Conference; INDIACom-2007
Computing for Nation Development, February 23 – 24, 2007
Bharati Vidyapeeth's Institute of Computer Applications and Management, New Delhi

On the Development of S-shaped Models in Software Reliability

P.K.Kapur[&] V.B.Singh[@] P.C.Jha[&]

[&]Department of Operational Research, University of Delhi, Delhi, India
[@]Delhi College of Arts and Commerce, University of Delhi, Delhi, India
(pkkapur1@gmail.com,singh_vb@rediffmail.com,Jhapc@yahoo.com)

ABSTRACT

Software is an important component of computer systems. In toady's world, computers have invaded every sphere of human lives. Therefore it is necessary to produce reliable software for the smooth operation of any automated system. Several software reliability growth models (SRGM) based on NHPP are proposed in literature to measure the quality characteristic reliability during the testing phase and for other related decision making such as to determine when to stop testing and release the software. These SRGM are developed on the basis of different assumptions to explain various types of failure growth curves. Most of the existing SRGMs describe either of following two type of reliability growth curves- Exponential and S-shaped. In this paper we review some existing S-shaped models and propose two new models incorporating the concept of time lag between the failure observation and corresponding removal. The performance and application of proposed models are demonstrated on two real life data sets. Results are encouraging.

KEYWORDS

Software Reliability Growth Model (SRGM), Non Homogenous Poisson Process (NHPP), S-shaped growth curve, Debugging Time Lag.

1. INTRODUCTION

In the today's environment of global competition, manual systems are rapidly computerized in order to provide faster service, reduce cost and better quality. The dependence of human kind on computer systems has increased very much. A mere postponement of a function can led to big losses in terms of money and time. On the other hand technology is growing at a very fast rate, which resulted in successful automization of large, complex as well as critical systems. Automization of large and safety critical systems increased the need of developing high quality software. The term "Software Reliability Engineering (SRE)" was invented in the late 1960s after the realization that all the lesions learned about how to program well were not helping to build better software systems. While the field of programming had made tremendous progress-through the systematic study of algorithms and data structures and the invention of "structured programming"-there were still major difficulties in building large software systems [4]. Software development process models also called Software Development Life Cycle Models (SDLC models) were employed by the developers to produce more reliable software. The SDLCM describes all the phases viz. (i) Requirement analysis and specification (ii) Design (iii) Coding (iv) Testing (v) Implementation of software development in a controlled manner. Even though Faults can be introduced during any of these phases. Though faults are detected after each phase by techniques like inspection, some errors remain undetected. Ultimately, these remaining errors will be reflected in the code. Hence the final code is likely to have some requirements errors and design errors, in addition to errors during the coding activity.

It is the testing phase of SDLC where the main emphasis of developers is on testing the software and thereby enhancing the quality of software. During the testing phase test cases simulated on user requirement specification are executed. If the outcome of a test case doesn't match the expected results a failure is said to occur. On the detection of a failure the code is given to a coding team where the corresponding fault is identified and removed. It not only uncovers faults introduced during coding, but also removes errors introduced during the previous phases. To improve the software quality, SRE plays an important role throughout the entire SDLC. SRGMs are the important tools of SRE which helps in estimating the Quality of software during testing using the past failure data. Computer aided software engineering (CASE) tools have been developed for different phases of SDLC to produce quality-based reliable software.

Reliability is the most important quality metric. Software reliability is defined as the probability that software will provide failure free operation in a fixed environment for a fixed interval of time [11]. The future failure behavior of a software system is predicted by studying and modeling its past failure behavior. How to enhance the reliability of the software systems and reduce the cost to an acceptable level becomes the main focus of software industry [13]. During the last three decades, many SRGMs have been developed in literature. Most of the existing SRGMs describes either of following two type of reliability growth curves - Exponential and S-shaped. The first known attempt to model the software testing process is attributed to Jelinski and Moranda [5]. The model is based on simple assumptions. Musa in year 1987 [11] proposed the Basic Execution Time Model and logarithmic Poisson Model, which have assumptions similar to those of Jelinski, and Moranda model. These models made a major contribution to the understanding of the error removal phenomenon and its relation to the calendar and execution time. Goel and Okumoto [3] proposed the first Non Homogeneous Poisson Process

(NHPP) SRGM under the assumption that failure intensity is linearly proportional to the remaining number of errors and defined the constant of proportionality as the fault detection rate (FDR) per remaining error. All these models describes exponential gwoth curve. In many software development projects it was observed that the relation between the cumulative number of errors removed and the testing time is S-shaped. The causes of S-shapedness are many and have been discussed by Yamada et al.[15], Ohba[12], Bittanti et al.[1], Kapur et al.[6] and others The NHPP based S-shaped model describes S-shaped reliability growth curve, which means that the curve crosses the exponential curve from below and the crossing occurs once and only once. The FDR becomes maximum at a certain time after testing begins, after which it decreases exponentially. In other words, some faults are covered by other faults, and before theses faults are actually removed, the uncovered faults remains undetected. Yamada et al. [15,16] claimed hat the software testing process usually involves a learning process where tester becomes familiar with the software products, environments, and software specifications.

Kapur et.al [8,9,10] contributed their research in the development of S-shaped models, taking into account the effect of the experience and learning gained by the testing team as testing progresses. During the testing process software is tested on the test cases built on user specifications and requirement by the testing team. If on the execution of a test case the outcome doesn't match the desired result a failure is said to occur. The failure is notified to the coding team who first isolates the cause of failure and then subsequently removes it. So there is a time lag between the failure occurrence and the corresponding fault removal. This time lag depends upon the complexity of the fault, the skills of the debugging team, the available manpower and software development environment's. In this paper we propose two SRGMs considering the time lag between the failure observation and corresponding fault removal describing S-shaped reliability growth curve.

This paper is organized as follows: In section 2 first a general description of a NHPP based SRGM is given. In section 2.2 we review some of the existing S-shaped SRGMs. In section 2.3 two new SRGMs are proposed considering the time lag between the failure occurrence and the corresponding fault removal describing S-shaped reliability growth curve. Further in section 3 we we validate the models on two real life data sets. Finally conclusions and scope of future research are indicated in section 4.

2. MODEL DEVELOPMENT

2.1 NHPP BASED SRGM-A GENERAL DESCRIPTION

If we define the expected no. of faults $N(t)$ whose mean value function is known as $m(t)$, then SRGM based on NHPP can be formulated as a Poisson process:

$$\Pr\{N(t) = n\} = \frac{(m(t))^n}{n!}\exp(-m(t)), \qquad n = 0,1,2 \qquad \ldots(1)$$

and $$m(t) = \int_0^t \lambda(x)dx$$

The intensity function $\lambda(x)$ (or the mean value function $m(t)$) is the basic building block of all the NHPP models existing in the software reliability engineering literature.

Notations

$m_f(t)$: is the expected number of failures observed in time t

$m_r(t)$: is the expected the number of faults removed in time t.

a : Constant, representing the number of faults lying dormant in the software at the beginning of testing.

β : Constant

$b(t)$: Fault removal/detection rate as a function of testing time

b_1, b_2, : Fault detection and removal rate per remaining fault.

Basic Assumption:

1. Software is subject to failures at random times caused by errors remaining in the software.
2. The fault removal process follows NHPP.
3. Identified faults are removed perfectly. No additional faults are introduced during removal process.
4. The faults existing in the software are modeled by two-stage removal process.
5. Each time a failure is observed, an immediate (or, delayed) effect take place to decide the cause of the failure and to remove it.
6. There is time lag between fault detection and corresponding fault removal.

2.2 LITERATURE REVIEW

Majority of Software Reliability Growth Models can be categorized under Non Homogeneous Poisson Process (NHPP) models as they assume NHPP to describe the failure phenomenon. The SRGMs reviewed and proposed in this paper describe S-shaped reliability growth curve. In this section, we reviewed some existing S-shaped SRGMs.

Model 1:

Yamada [15, 16] proposed a SRGM considering the concept of failure observation and corresponding fault removal as a two stage process. The differential equations describing the failure and removal phenomenon are given as

$$\frac{dm_f(t)}{dt} = b\,(a - m_f(t)) \qquad \ldots(2)$$

and $$\frac{dm_r(t)}{dt} = b\,(m_f(t) - m_r(t)) \qquad \ldots(3)$$

Solving it with the initial condition $m_f(0) = m_r(0) = 0$ we get

$$m_r(t) = a\left(1 - \left(1 + bt\right)\exp(-bt)\right) \quad \ldots(4)$$

Model 2:

Fault removal process is greatly influenced by the skill, efficiency, experience and learning of the removal team. In order to incorporate these facts about the testing team Kapur et.al [8,9,10] proposed logistic fault removal rate. The differential equations desribing the model proposed by then for failure observation and corresponding fault removal are given as

$$\frac{dm_f(t)}{dt} = b\,(a - m_f(t)) \quad \ldots(5)$$

and $$\frac{dm_r(t)}{dt} = b(t)\,(m_f(t) - m_r(t)) \quad \ldots(6)$$

Where $b(t) = \dfrac{b}{1 + \beta\exp(-bt)}$

Solving it with the initial condition $m_f(0) = m_r(0) = 0$ we get

$$m_r(t) = a\left(\frac{1 - (1 + bt)\exp(-bt)}{1 + \beta\exp(-bt)}\right) \quad \ldots(7)$$

Model 3:

In the previous model fault detection and removal rates are assumed to be same however in real life situations these may vary. Kapur et al modified the above model assuming different rates of fault detection and removal per remaining error. The differential equations for the model are described as

$$\frac{dm_f(t)}{dt} = b_1\,(a - m_f(t)) \quad \ldots(8)$$

and $$\frac{dm_r(t)}{dt} = b_2(t)\,(m_f(t) - m_r(t)) \quad \ldots(9)$$

Where $b_2(t) = \dfrac{b_2}{1 + \beta\exp(-b_2 t)}$

Solving it with the initial condition $m_f(0) = m_r(0) = 0$ we get

$$m_r(t) = a\left(\frac{1 - \frac{1}{(b_2 - b_1)}(b_2\exp(-b_1 t) - b_1\exp(-b_2 t))}{1 + \beta\exp(-bt)}\right) \quad \ldots(10)$$

The model take into account the effect of the experience and learning gained by the testing team as testing progresses.

Model 4:

All of the previous models describe the fault removal process as a two-stage process. In this model Kapur et al [8,9,10] described the fault removal process as a three stage process namely- Failure observation, fault isolation and fault removal. The time lag between the failure observation and fault isolation / removal represents the severity of the fault. Harder the fault, longer is the time lag. The differential equations describing the model are given as

$$\frac{dm_f(t)}{dt} = b\,(a - m_f(t)) \quad \ldots(11)$$

$$\frac{dm_i(t)}{dt} = b\,(m_f(t) - m_i(t)) \quad \ldots(12)$$

$$\frac{dm_r(t)}{dt} = b(t)\,(m_i(t) - m_r(t)) \quad \ldots(13)$$

Where $b(t) = \dfrac{b}{1 + \beta\exp(-bt)}$

Solving it with the initial condition $m_f(0) = m_i(0) = m_r(0) = 0$ we get

$$m_r(t) = a\left(\frac{1 - \left(1 + bt + \frac{b^2 t^2}{2}\right)\exp(-bt)}{1 + \beta\exp(-bt)}\right) \quad \ldots(14)$$

In the next section we propose two models describing S-shaped reliability growth curve considering the time lag between the failure occurrence and corresponding removal.

2.3 THE PROPOSED FRAMEWORK

Here we assume that $m_f(t)$ describes the number of faults detected & isolated by time t. Since the testing of software is influenced by the efficiency and experience of the testing team and as the testing progresses the testing team gains experience with the code due to which leaning occurs. As such the differential equation describing the failure occurrence and isolation is given as

$$\frac{dm_f(t)}{dt} = b(t)\,(a - m_f(t)) \quad \ldots(15)$$

Where $b(t) = \dfrac{b}{1 + \beta\exp(-bt)}$, which describe learning of the testing team as testing progresses.

Solving equation (15) with the initial condition $m_f(0) = 0$ we get

$$m_f(t) = a\left(\frac{1 - \exp(-bt)}{1 + \beta\exp(-bt)}\right) \quad \ldots(16)$$

After the isolation of fault the removal team start the fault removal process. Hence there is a time lag between the isolation and the removal process. Considering a definite time lag the differential equation describing the removal process is given as

$$m_r(t) = m_f(t - \Delta t) \quad \ldots(17)$$

Proposed Model 1:

For the fault, which has less impact, we define time lag is given by

$$\Delta t = \frac{1}{b}\ln(1 + bt).$$

Substituting Δt in equation (17) we get:

$$m_r(t) = a\left(\frac{1-(1+bt)\exp(-bt)}{1+\beta(1+bt)\exp(-bt)}\right) \quad \ldots(18)$$

Proposed Model 2:

The time lag between the fault detection and removal represents the severity of the fault. Harder the fault, longer is the time lag. For a fault, which has more impact, we define time lag as follows:

$$\Delta t = \frac{1}{b}\ln\left(1+bt+\frac{b^2t^2}{2}\right)$$

Substituting Δt in equation (17) we get:

$$m_r(t) = a\left(\frac{1-(1+bt+\frac{b^2t^2}{2})\exp(-bt)}{1+\beta(1+bt+\frac{b^2t^2}{2})\exp(-bt)}\right) \quad \ldots(19)$$

3. Model Validation and Parameter Estimation

The success of software reliability growth model depends heavily upon quality of failure data collected. The parameters of the SRGMs are estimated based upon these data. Method of least squares or maximum likelihood has been suggested and widely used for estimation of parameters of an SRGM. The models discussed in this paper are non-linear model and it is difficult to find solution for nonlinear models using Least Square method and require algorithms to solve it. Statistical software packages such as SPSS help to overcome this problem.

3.1 COMPARISON CRITERIA

The performance of the SRGM is judged by its ability to fit the past software fault and to predict satisfactorily the future behavior of the software fault removal process. Therefore, we use following comparison criteria.

1. The Mean Square Fitting Error (MSE): The model under comparison is used to simulate the fault data. The difference between the estimated values, $m(t_i)$ and the observed values y_i is measured by MSE as follows.

$$MSE = \frac{\sum_{i=1}^{k}(m(t_i)-y_i)^2}{k}$$

2. Coefficient of Determination (R^2): This goodness of fit measure can be used to investigate whether a significant trend exists in the observed failure intensity. We define this coefficient as the ratio of the sum of squares resulting from the trend model to that from constant model subtracted from 1.

$$R^2 = 1 - \frac{corrected\ SS}{residual\ SS}$$

R^2 measures the percentage of the total variation about the mean accounted for the fitted curve. It ranges in value from 0 to 1. Small values indicate that the model does not fit the data well. The larger R^2, the better the model explains the variation in the data.

3.2 MODEL VALIDATION

To validate the models two real life data sets cited in literature are chosen. First data set is from Brooks and Motley Brooks [1]. The fault data set is for a radar system of size 124 KLOC (kilo lines of code) tested for 35 months in which 1301 faults were identified. The second data set is from Pham of a real time system tested for 21 weeks in which 26 faults were identified. Table-1(a) shows the estimated values of parameters for the SRGMS reviewed and proposed in this paper and two comparison criteria: R^2 and MSE for data set 1. while theTable-1(b) shows the estimation results for data set 2. Estimation results shows that proposed SRGM 1 best describes the two data sets. Goodness of fit curves are shown in figures 1 and 2.

4. CONCLUSION:

In this paper, we first reviewed some exiting S-shaped Software reliability growth models developed under the different assumptions. We have also proposed S-shaped Software reliability growth model incorporating different debugging time lag. Experimental results show that the proposed framework to incorporate debugging time lag concept for a SRGM has a fairly accurate goodness of fit. Consequently, the development and production of software can be improved significantly. In future, we will try to incorporate software development team competency and human reliability factor in to software reliability modeling.

REFERENCES

[1] Bittanti S, Bolzern P, Pedrotti E, Scattolini R. "A flexible modelling approach for Software reliability growth" *Software Reliability Modelling and Identification (Ed.) G. Goos and J. Harmanis*, Springer Verlag, Berlin, **1988**, 101-140.

[2] Brooks WD, Motley RW. "Analysis of discrete software reliability models - *Technical Report (RADC-TR-80-84)*" **1980**; New York: Rome Air Development Center.

[3] Goel AL, Okumoto K. "Time dependent error detection rate model for software reliability and other performance measures" *IEEE Transactions on Reliability* **1979**; R-28 (3).

[4] Ghezzi Carlo,Jazayeri and Mandrioli "Fundamentals of Software Engineering",**2002;**Prentice Hall of India Pvt. Ltd. New Delhi

[5] Jelinski,Z. and Moranda P.B.Moranda "Software Reliability Research" *Statistical computer performance evaluation*,ed.W.Freiberger,pp.465-484.New York: Academic Press

[6] Kapur PK, Garg RB. "A software reliability growth model for an error removal phenomenon" *Software Engineering Journal* **1992**; 7: 291-294.

[7] Kapur PK, Garg RB, Kumar S. "Contributions to hardware and software reliability" **1999**; Singapore: World Scientific Publishing Co. Ltd.

[8] Kapur P.K., Jha P.C. and Gupta Amit. "Optimal Release Policy of a Software Fuzzy Environment" Published in the Proceeding *Mathematical Modeling, Application Issue and Analysis,* held BITS Pilani,8-9 th oct.-2004,pp125-134

[9] Kapur,P.K.Goswami,Gupta Amit "A Software Reliability Growth Model for Distributed Development Environment with Learning Function and Errors of DiffferentSeverity" Published in the Proceeding *Mathematical Modeling ,Application Issue and Analysis,*BITS Pilani,8-9 th oct.-2004,pp 99-110

[10] Kapur P.K., Goswami D.N., Bardhan A.K. "A Flexible Delayed s-Shaped Software Reliability Growth Model", accepted CSI

[11] Musa JD, Iannino A, Okumoto K. "Software reliability: Measurement, Prediction, Applications" **1987**; New York: Mc Graw Hill

[12] Ohba M. "Software reliability analysis models" IBM Journal of Research and Development **1984**; 28: 428-443.

[13] Pham H. "Software Reliability" **2000**; Springer -Verlag Singapore Pte. Ltd

[14] Xie M. "Software reliability modelling" **1991**; World Scientific

[15] Yamada S, Ohba M., Osaki S. "S-shaped software reliability growth modelling for software error detection" *IEEE Trans. On Reliability*, **1983**; R-32(5): 475-484.

[16] Yamada S, Osaki S. and Narithisa,H. " A software reliability growth model with two types of errors" *RAIRO*, **1985**; Vol.19,pp.87-104.

Table 1(a)

Data set	Models under Comparison	Parameter Estimation			Comparison Criteria	
		a	b	β	R^2	MSE
DS-1	Model-1	1689	.089848	-	.9872	2713
	Model -2	1361	.173788	7	.9969	647
	Model -3	1337	$b_{1=}$ .339324 $b_{2=}$ .188881	14	.9974	540
	Model -4	1401	.18117241	1	.9948	1098
	Proposed M1	1330	.2327098	5	.9984	340
	Proposed M2	1331	.2578622	2	.9970	637

Table 1(b)

	Models under Comparison	Parameter Estimation			Comparison Criteria	
		a	b	β	R^2	MSE
DS-2	Model-1	39	.1104082	-	.98240	1.3
	Model -2	27	.2436316	7.30	.98914	0.83
	Model -3	28	$b_{1=}$ .2189 $b_{2=}$ .2431	6.75	.98897	0.84
	Model -4	29	.2358853	.444	.98644	1.0
	Proposed M1	26	.3423815	6.58	.99129	0.66
	Proposed M2	29	.3852893	2.57	.98917	0.83

Fig.1 (Data Set-I)

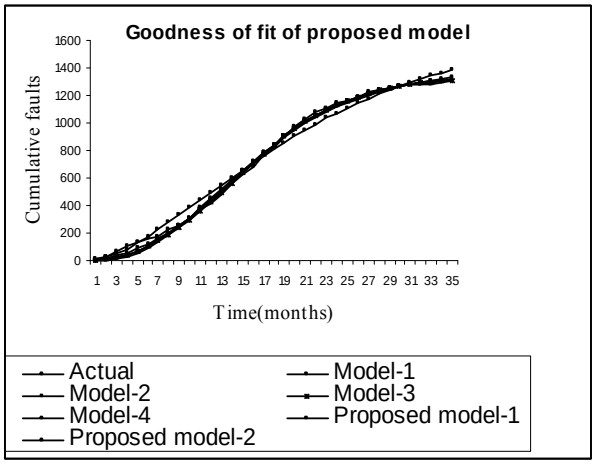

Fig.1 (Data Set-II)