Assessment of Distribution Based SRGM with the Effect of Change-Point and Imperfect Debugging Incorporating Irregular Fluctuations

Ompal Singh[1], Adarsh Anand[1,*], Jagvinder Singh[1] and P. K. Kapur[2]
[1,*]Department of Operational Research, University of Delhi, Delhi, India
[2]Amity International Business School, Amity University, Noida, U.P., India

Computer software has progressively turned out to be an essential component in modern technologies. Testability is the probability whether tests will detect a fault, given that a fault in the program exists. How efficiently the faults will be uncovered depends upon the testability of the software. Various researchers have proposed qualitative and quantitative techniques to improve and measure the testability of software. Furthermore, it is well known fact that the progress of software testing is influenced by various uncertainty factors like effort expenditure, skill of test personal, testing tool, defect density and irregular state of software fault-report phenomena on the bug tracking system. Hence, there is an irregular fluctuation in fault detection/removal rate during testing phase. In this paper, distribution based software reliability growth models have been developed by applying Itô type Stochastic Differential Equations in order to incorporate (i) the irregular fluctuation in the fault detection process due to various uncertainty factor during testing phase; (ii) two types of imperfect debugging and (iii) change-point concept. The proposed stochastic differential equation based models have been validated using real data sets. Various comparison criteria results for goodness of fit have also been presented in the paper.

Keywords: Hazard Rate, Imperfect Debugging, Non-Homogeneous Poisson Process, Stochastic Differential Equations (SDE).

1. INTRODUCTION

Software reliability engineering is centered on an extremely imperative software attribute which is termed as reliability. Software reliability is defined as the probability of failure-free software operation for a specified period of time in a specified environment[1]. It is one of the attributes of software quality which is commonly accepted as its key factor because it quantifies software failures, which can destroy a powerful system by making it inoperative. So, to quantify reliability, software is tested during the testing phase of software development life cycle.

Software testing is the method of raising the confidence that the software is free of flaws. But a major problem in testing software is that it cannot be made 100% bug free. This is not because programmers are careless or irresponsible, but because the nature of software code is complex and humans have only limited ability to manage complexity. Therefore, although testing helps in assessing and improving quality, but it cannot be performed indefinitely. Hence, time is considered as a very important controlling factor of testing phase. To monitor the relationship between the number of faults removed and time mathematically several Software Reliability Growth Models (SRGMs) are developed in literature[2,3,4,5].

Research has been conducted in software reliability engineering over the past three decades and many software reliability growth models (SRGM) have been proposed. The pioneering attempt in non-homogenous Poisson process based on SRGM was made by Goel and Okumoto (G-O)[3]. The model describes the failure observation phenomenon by an exponential curve. There are also SRGM that describe either S-shaped curves or a mixture of exponential and Sshaped curves (flexible). Some of the important contributions of these type of models are due to Yamada *et al*[6], Ohba[7], Bittanti *et al*[2], Kapur and Garg[5], Pham[1] etc. In most of the these models it is assumed that whenever an attempt is made to remove a fault, it is removed with certainty i.e. a case of perfect debugging. But the debugging activity is not always perfect because of number of factors like tester’s skill/expertise etc. In practical software development scenario, the number of failures observed/detected may not be necessarily same as the number of errors removed/ corrected. Kapur and Garg[5] have discussed in their error removal phenomenon model that as testing grows and testing team gains experience, additional numbers of faults are removed without them causing any failure. The testing team, however, may not be able to remove/correct fault perfectly on observation/detection of a failure and the original fault may remain leading to a phenomenon known as imperfect debugging, or replaced by another fault resulting in fault generation. In case of imperfect debugging the fault content of the software is not changed, but because of incomplete understanding of the software, the original detected fault is not removed perfectly. But in case of fault generation, the total fault content increases as the testing progresses because new faults are introduced in the system while removing the old original faults.

Model due to Obha and Chou[7] is an fault generation model applied on G-O model and has been also named as Imperfect debugging model. Kapur and Garg[8] introduced the imperfect debugging in G-O model. They assumed that the FDR per remaining faults is reduced due to imperfect debugging. Thus the number of failures observed/detected by time infinity is more than the initial fault content. Although these two models describe the imperfect debugging phenomenon yet the software reliability growth curve of these models is always exponential. Moreover, they assume that the probability of imperfect debugging is independent of the testing time. Thus, they ignore the role of the learning process during the testing phase by not accounting for the experience gained with the progress of software testing. Pham[1] developed an SRGM for multiple failure types incorporating fault generation. Zhang *et al*[9] proposed a testing efficiency model which includes both imperfect debugging and fault generation, modeling it on the number of failures experienced/observed/detected, however both imperfect debugging and fault generation are actually seen during fault removal/correction. Recently, Kapur *et al*[10] proposed a flexible SRGM with imperfect debugging and fault generation using a logistic function for fault detection rate which reflects the efficiency of the testing/removal team.

We execute the program in specific environment and improve its quality by detecting and correcting faults. Many SRGM assume that, during the fault detection process, each failure caused by a fault occurs independently and randomly in time according to the same distribution Musa *et al*[11]. But the failure distribution can be affected by many factors such as running environment, testing strategy, defect density and resource allocation. On the other hand, in practice, if we want to detect more faults for a short period of time, we may introduce new techniques or tools that are not yet used, or bring in consultants to make a radical software risk analysis. In addition, there are newly proposed automated testing tools for increasing test coverage and can be used to replace traditional manual software testing regularly. The benefits to software developers/testers include increased software quality, reduced testing costs, improved release time to market, repeatable test steps, and improved testing productivity. These technologies can make software testing and correction easier, detect more bugs, save more time, and reduce much expense. Altogether, we wish that the consultants, new automated test tools or techniques could greatly help us in detecting additional faults that are difficult to find during regular testing and usage, in identifying and correcting faults most cost effectively and in assisting clients to improve their software development process. Thus, the fault detection rate may not be smooth and can be changed at some time moment 'τ' called change-point. Many researchers have incorporated change point in software reliability growth modeling. Firstly Zhao[12] incorporated changepoint in software and hardware reliability. Huang *et al*[13] used change-point in software reliability growth modelling with testing effort functions. The imperfect debugging with change-point has been introduced in software reliability growth modeling by Shyur[14]. Kapur *et al*[15,16] introduced various testing effort functions and testing effort control with change-point in software reliability growth modeling. Goswami *et al*[17] and Kapur *et al*[18] proposed a software reliability growth model for errors of different severity using change-point. The multiple change-points in software reliability growth modeling for fielded software has been proposed by Kapur *et al*[18].

In literature various software reliability growth models based on stochastic differential equations have been proposed in order to incorporate irregular fluctuation in fault detection rate during testing phase. Firstly, Yamada *et al*[19] asserted that if the size of the software system is large than the number of faults detected during the testing phase also is large and the change in the number of faults which are corrected and removed through each debugging becomes small compared with the initial fault content at the beginning of testing. For instance, in the development of a computer operations system, number of faults are detected and removed during the long testing period, and the system is then released to the market. However, a number of faults are then found by the users, and the software company releases an updated version of the system. Thus, in this case, the number of faults that remain in the system can be considered to be a stochastic process with a continuous state space. He proposed a simple software reliability growth model to describe the fault-detection process during the testing phase by applying an Itô type Stochastic Differential Equation. Yamada *et al*[20] proposed a simple software reliability growth model to describe the fault detection process during the testing phase by applying Itô's type Stochastic Differential Equation (SDE) and obtain several software reliability measures using the probability distribution of the stochastic process. Lee *et al*[21] used

SDEs to represent a per-fault detection rate that incorporate an irregular fluctuation instead of an NHPP, and consider a per-fault detection rate that depends on the testing time *'t'*. Later on Yamada *et al* [22] proposed a flexible Stochastic Differential Equation Model describing a fault-detection process during the system-testing phase of the distributed development environment. Following Kapur *et al* [23,9] developed generalized and flexible software reliability growth models using SDE.

2. BASIC ASSUMPTIONS

A Software Reliability Growth Model (SRGM) explains the time dependent behavior of fault removal. Majority of these Software Reliability Growth Models can be categorized under Non Homogeneous Poisson Process (NHPP) models. The NHPP models are based on the assumption that the software system is subject to failures at random times caused by manifestation of remaining faults in the system. For modeling the software fault detection phenomenon, counting process $\{N(t); t \geq 0\}$ is defined which represents the cumulative number of software faults detected by testing time 't'. The SRGM based on NHPP is formulated as:

$$Pr\left\{N(t)=n\right\} = \frac{m(t)^n \cdot \exp\left(-m(t)\right)}{n!}, \quad n = 0,1,2,... \qquad \text{-(1)}$$

where m(t) is the mean value function of the counting process *N(t)*.

The modeling framework proposed in this paper is based on the following assumptions:

(a) The software fault detection Process is modeled as a stochastic process with a continuous state space.
(b) Software is subject to failures at random times caused by errors remaining in the software.
(c) Failure rate is equally affected by all the faults remaining in the software.
(d) When a software failure occurs, an instantaneous repair effort starts and the following may occur:
  (i) Fault content is reduced by one with probability *p.*
  (ii) Fault content remains unchanged with probability *1-p*.
(e) During the fault removal process, whether the fault is removed successfully or not new faults are generated with a constant probability α .
(f) Fault detection/removal rate may change at any time moment.

3. NOTATIONS

m(t)	:	Number of faults detected during the testing time t and is a random variable.
E[m(t)]	:	Expected number of faults detected in the time interval (0, t) during testing phase.
a(t)	:	Total fault content of the software dependent on the time.
p_i	:	The probability of perfect debugging (i =1,2; for before and after the change point).

α_i	:	The rate at which the errors may be introduced during the debugging process per detected fault (i =1,2 for before and after the change point).
σ_i	:	Positive constant that represents the magnitude of the irregular fluctuation for faults before and after change point.
$F_i(t)$	:	distribution functions for fault removal/correction times.
$f_i(t)$	:	density functions for fault removal/correction times.
b_i (t)	:	Time dependent rate of fault removal per remaining faults. (i =1,2; for before and after the change point).
z(t)	:	Hazard rate function.
β_1, β_2	:	Constant parameter describing learning in the fault removal rate before and after the change point.

4. MODEL DEVELOPMENT

In this section, we formulate distribution based software reliability growth models by applying Itô type Stochastic differential equations[24] that incorporates change-point and two types of imperfect debugging. Since the faults in the software systems are detected and eliminated during the testing phase, the number of faults remaining in the software system gradually decreases as the testing procedure goes on. Thus under the common assumptions for software reliability growth modeling, we consider the following linear differential equation:

$$\frac{dm(t)}{dt}=\begin{cases}\dfrac{f_1(t)}{1-F_1(t)}.p_1[a(t)-m(t)] & for\ \ 0\le t\le\tau\\ \dfrac{f_2(t)}{1-F_2(t)}.p_2[a(t)-m(t)] & for\ \ t>\tau\end{cases} \qquad \text{- (2)}$$

Here we consider the fault detection rate as hazard rate *z(t)*. We further assume that faults can be introduced during the debugging phase with a constant fault introduction rate 'α'. Therefore, the fault content rate function, *a(t)*, is a linear function of the expected number of faults detected by time '*t*'. That is *a(t) = a +*α *m(t)*, so the equation (2) becomes:

$$\frac{dm(t)}{dt}=\begin{cases}\left(\dfrac{f_1(t)}{1-F_1(t)}\right).p_1[a+\alpha_1.m(t)-m(t)] & for\ \ 0\le t\le\tau\\ \left(\dfrac{f_2(t)}{1-F_2(t)}\right).p_2[a+\alpha_1 m(\tau)+\alpha_2(.m(t)-m(\tau))-m(t)] & for\ \ t>\tau\end{cases} \qquad \text{-(3)}$$

Rearranging the terms in equation (3) we have:

$$\frac{dm(t)}{dt}=\begin{cases} p_1.(1-\alpha_1)\left(\dfrac{f_1(t)}{1-F_1(t)}\right)[a_1-m(t)] & for\ \ 0\le t\le\tau \\ p_2.(1-\alpha_2)\left(\dfrac{f_2(t)}{1-F_2(t)}\right)[a_2-m(t)] & for\ \ t>\tau \end{cases} \qquad \text{-(4)}$$

where, $a_1=\dfrac{a}{(1-\alpha_1)}$ and $a_2=\dfrac{a+(\alpha_1-\alpha_2)m(\tau)}{(1-\alpha_2)}$

In the proposed modeling framework, we assume that the hazard rate *z(t)* is reduced by factor $p.(1-\alpha)$ and hence we call it 'Reduced Hazard Rate (RHR)' given as:

$$z(t)=p.(1-\alpha)\left(\frac{f(t)}{1-F(t)}\right) \qquad \text{-(5)}$$

It might happen that the rate is not known completely known but subject to some random environment effects such as the testing effort expenditure, the skill level of the testers, the testing tool and so on and thus might have irregular fluctuation in *'RHR'*. Therefore equation (5) can be actually expressed as:

$$z(t)=p.(1-\alpha)\left(\frac{f(t)}{1-F(t)}\right)+"noise" \qquad \text{-(6)}$$

$$\frac{dm(t)}{dt}=\begin{cases} \left\{p_1.(1-\alpha_1)\left(\dfrac{f_1(t)}{1-F_1(t)}\right)+"noise"\right\}[a_1-m(t)] & for\ \ 0\le t\le\tau \\ \left\{p_2.(1-\alpha_2)\left(\dfrac{f_2(t)}{1-F_2(t)}\right)+"noise"\right\}[a_2-m(t)] & for\ \ t>\tau \end{cases} \qquad \text{-(7)}$$

Let $\gamma(t)$ be a standard Guassian white noise and σ be a positive constant representing a magnitude of the irregular fluctuations. So equation (7) can be written as:

$$\frac{dm(t)}{dt}=\begin{cases} (z_1(t)+\sigma.\gamma(t)).[a_1-m(t)] & for\ \ 0\le t\le\tau \\ (z_2(t)+\sigma.\gamma(t)).[a_2-m(t)] & for\ \ t>\tau \end{cases} \qquad \text{-(8)}$$

Using the unit step function given by:

$$U(x) = \begin{cases} 0 \;\; if \;\; x < 0 \\ 1 \;\; if \;\; x > 1 \end{cases}$$

The hazard rate function given in equation (7) can be written as:

$$z(t) = \left\{ p_1.(1-\alpha_1)\left(\frac{f_1(t)}{1-F_1(t)}\right) + \sigma\gamma(t) \right\} U(\tau - t)$$
$$+ \left\{ p_2.(1-\alpha_2)\left(\frac{f_2(t)}{1-F_2(t)}\right) + \sigma\gamma(t) \right\} .U(t-\tau) \qquad \text{-(9)}$$

the unit-step function uses $U(t-\tau)$that is used to denote that an event occurring at time '*t*' cannot take place before the time instant $\tau$and vice-versa.The unit step function described above incorporates the probability of perfect debugging and error generations as well.

Now using equation (9), and applying the concept of Itô type equation, equation (8) can be solved as follows:

$$dm(t) = \begin{cases} \left(z_1(t) - \tfrac{1}{2}\sigma^2\right).[a_1 - m(t)] \text{ dt}+\sigma[a_1 - m(t)]dw(t) & for \;\; 0 \le t \le \tau \\ \left(z_2(t) - \tfrac{1}{2}\sigma^2\right).[a_2 - m(t)] \text{ dt}+\sigma[a_2 - m(t)]dw(t) & for \;\; t > \tau \end{cases} \qquad \text{-(10)}$$

where $w(t)$is a one-dimensional Wiener process before and after the change-point, which is formally defined as an integration of the white noise$\gamma(t)$ with respect to time.

Using Itô formula, solution to equation (10) with initial condition:
At t=0, m(0)=0 and at t = τ, m(t) = m(τ)

we get *m(t)* as follows:

$$m(t) = \begin{cases} \left\{ \frac{a}{(1-\alpha_1)}.\left(1-\left(1-F_1(t)\right)^{p_1.(1-\alpha_1)}.e^{-\sigma.W(t)}\right) \right\} for \;\; 0 \le t \le \tau \\ \left\{ \frac{a}{(1-\alpha_2)}.\left(1-\left(1-F_1(\tau)\right)^{p_1.(1-\alpha_1)}\right).\left(\frac{1-F_2(t)}{1-F_2(\tau)}\right)^{p_2.(1-\alpha_2)} .e^{-\sigma.W(t)} \right\} + \\ \qquad \left(\frac{\alpha_1 - \alpha_2}{1-\alpha_2}\right).m(\tau)\, for \;\; t > \tau \end{cases} \qquad \text{-(11)}$$

Using the fact that the wiener process $w(t)$ is a Gaussian process and has the following properties:

$$\Pr[w(0)=0]=1,$$
$$E[w(t)]=0;$$
$$E[w(t)w(t')]=\min[t,t']$$

The expected number of faults removed is given by:

$$E(m(t))=\begin{cases}\left\{\dfrac{a}{(1-\alpha_1)}\cdot\left(1-\left(1-F_1(t)\right)^{p_1.(1-\alpha_1)}.e^{\frac{\sigma^2.t}{2}}\right)\right\} for\ 0\le t\le\tau \\ \left\{\dfrac{a}{(1-\alpha_2)}\cdot\left(1-\left(1-F_1(\tau)\right)^{p_1.(1-\alpha_1)}\right)\cdot\left(\dfrac{1-F_2(t)}{1-F_2(\tau)}\right)^{p_2.(1-\alpha_2)}.e^{\frac{\sigma^2.t}{2}}\right\}+ \\ \left(\dfrac{\alpha_1-\alpha_2}{1-\alpha_2}\right).m(\tau)\,for\ \ t>\tau\end{cases} \qquad \text{-(12)}$$

4.1. SRGM-I

The following exponential distribution function is used to model SRGM-I:

$$F_1(t)=1-e^{-b_1.t}\ \ for\ t\le\tau \text{ and } F_2(t)=1-e^{-b_2.t}\ \ for\, t>\tau$$

Substituting these values in equation (12), we get:

$$E[m(t)]=\begin{cases}\dfrac{a}{(1-\alpha_1)}(1-e^{-p_1(1-\alpha_1)b_1t+\frac{1}{2}\sigma^2t}) & for\ 0\le t\le\tau \\ \dfrac{a}{(1-\alpha_2)}(1-e^{-p_1(1-\alpha_1)b_1\tau-p_2(1-\alpha_2)b_2(t-\tau)+\frac{1}{2}\sigma^2t})+\dfrac{(\alpha_1-\alpha_2)}{(1-\alpha_2)}m(\tau)\,for\ t>\tau\end{cases} \qquad \text{-(13)}$$

The above model can be reduced to the model given by Shyur[14] if we consider the perfect debugging, no fault generation and to Kapur *et al*[25] if we consider no irregular fluctuation.

4.2. SRGM-II

Let F(t) follow a two stage Erlangian distribution function i.e.

$$F_1(t)=1-(1+b_1.t)e^{-b_1.t}\ \ for\ t\le\tau\ and\ F_2(t)=1-(1+b_2.t)e^{-b_2.t}\ \ for\, t>\tau$$

Similarily, on substituting the value of F_1(t) and F_2(t) in equation (12) we get:

$$E[m(t)]=\begin{cases}\dfrac{a}{(1-\alpha_1)}(1-(1+b_1t)e^{-p_1(1-\alpha_1)b_1t+\frac{1}{2}\sigma^2t}) & for\ 0\le t\le\tau\\ \dfrac{a}{(1-\alpha_2)}(1-(1+b_1\tau)^{p_1(1-\alpha_1)}.\left(\frac{1+b_2t}{1+b_2\tau}\right)^{p_2(1-\alpha_2)}e^{-p_1(1-\alpha_1)b_1\tau-p_2(1-\alpha_2)b_2(t-\tau)+\frac{1}{2}\sigma^2t})+ & \\ \qquad\dfrac{(\alpha_1-\alpha_2)}{(1-\alpha_2)}m(\tau)\ for\ t>\tau & \end{cases} \qquad \text{-(14)}$$

The above model can be reduced to the model given by Archana[26] if we consider the case of perfect debugging and no fault generation and to kapur et al if we do not consider irregular fluctuation in the hazard rate.

4.3. SRGM-III

Let *F(t)* follow a logistic distributiontwo stage Erlangian distribution function i.e.

$$F_1(t)=\frac{1-e^{-b_1.t}}{1+\beta_1.e^{-b_1.t}}\ for\ t\le\tau\ and\ F_2(t)=\frac{1-e^{-b_2.t}}{1+\beta_2.e^{-b_2.t}}\ for\,t>\tau$$

On substituting the values of F_1(t) and F_2(t) in equation (12) we get:

$$E[m(t)]=\begin{cases}\dfrac{a}{(1-\alpha_1)}(1-\left(\frac{(1+\beta_1)}{(1+\beta_1e^{-b_1t})}\right)^{-p_1(1-\alpha_1)}.e^{-p_1(1-\alpha_1)b_1t+\frac{1}{2}\sigma^2t})\ for\ 0\le t\le\tau\\ \dfrac{a}{(1-\alpha_2)}(1-\left(\frac{(1+\beta_1)}{(1+\beta_1e^{-b_1\tau})}\right)^{p_1(1-\alpha_1)}.\left(\frac{(1+\beta_2e^{-b_2\tau})}{(1+\beta_2e^{-b_2t})}\right)^{p_2(1-\alpha_2)}.e^{-p_1(1-\alpha_1)b_1\tau-p_2(1-\alpha_2)b_2(t-\tau)+\frac{1}{2}\sigma^2t})+\\ \qquad\dfrac{(\alpha_1-\alpha_2)}{(1-\alpha_2)}m(\tau)\ for\ t>\tau\end{cases} \qquad \text{-(15)}$$

For further simplifications, the estimation procedure we may assume:

$$\alpha_1=\alpha_2=\alpha \text{ and } \rho_1=\rho_2=\rho\ .$$

5. MODEL VALIDATION, COMPARISON CRITERIA AND DATA ANALYSIS

5.1. Model Validation

To illustrate the estimation procedure and application of the SRGM (existing as well as proposed) we have carried out the data analysis of real software data set. The parameters of the models have been estimated using statistical package SPSS and the change-point of the data sets have been judged by using change-point analyzer.

Data Set 1 (DS-1)

The first data set (DS-1) had been collected during 35 months of testing a radar system of size 124 KLOC and 1301 faults were detected during testing. This data is cited from Brooks and Motley[27]. The change-point for this data set is 17th month.

Data Set 2 (DS-2)

The second data set (DS-2) had been collected during 19 weeks of testing a real time command and control system and 328 faults were detected during testing. This data is cited from Ohba[7]. The change-point for this data set is 6th week.

5.2. Comparison Criteria for SRGM

The performance of SRGM are judged by their ability to fit the past software fault data (goodness of fit) and predicting the future behavior of the fault. The performance analysis of proposed model is measured by the four common criteria *Bias, Variation, RMSPE* and *MSE*.

Table 1a: Model Parameter Estimation Results (DS-I)

Models	a	b_1	b_2	α	p	β_1	β_2	σ
SRGM-I	1685	0.061	0.073	0.516	0.377	-	-	0.018
SRGM-II	1649	0.099	0.101	0.001	0.910	-	-	0.349
SRGM-III	1334	0.123	0.245	0.001	0.724	2.03	26.9	0.014

Table 1b: Model Comparison Results (DS-I)

Models	R^2	*MSE*
SRGM-I	0.974	5608.0900
SRGM-II	0.988	2544.4300
SRGM-III	0.999	207.4532

Table 2a: Model Parameter Estimation Results (DS-II)

Models	a	b_1	b_2	α	p	β_1	β_2	σ
SRGM-I	385	0.203	0.210	0.365	0.329	-	-	0.022330
SRGM-II	465	0.602	0.562	0.001	0.156	-	-	0.000105
SRGM-III	360	0.352	0.280	0.063	0.567	6	4	0.000014

Table 2b: Model Comparison Results(DS-II)

Models	R^2	*MSE*
SRGM-I	0.987	121.985
SRGM-II	0.990	105.906
SRGM-III	0.992	82.130

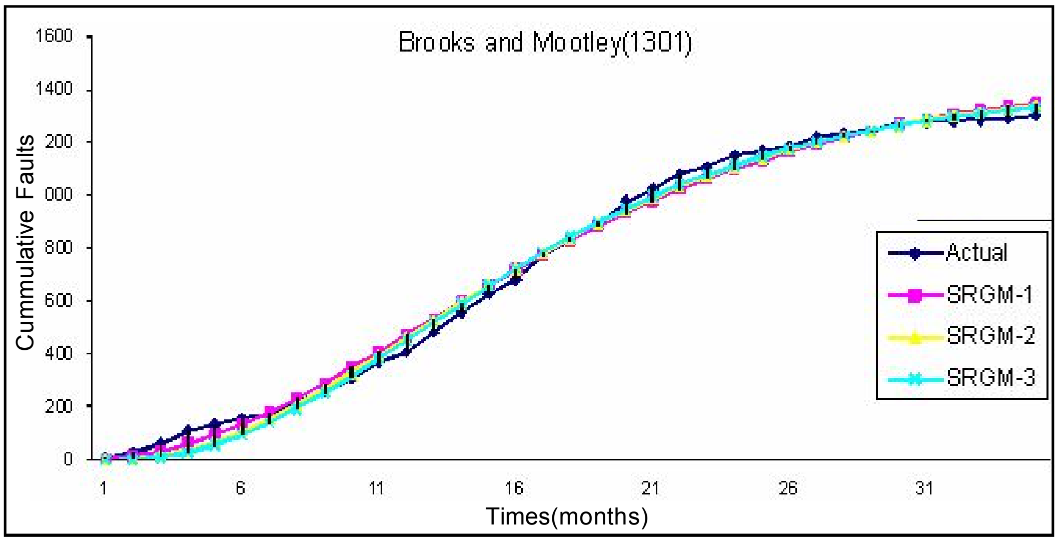

Fig. 1: Goodness of Fit Curve for DS-I

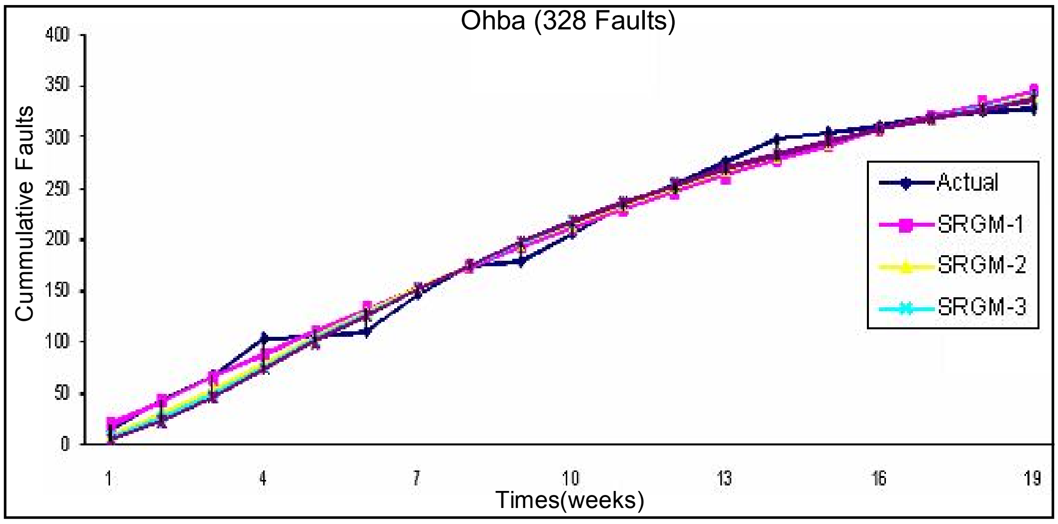

Fig. 2: Goodness of Fit Curve for DS-II

6. CONCLUSION

In this paper, distribution based software reliability growth models have been developed by applying Itô's type Stochastic Differential Equations in order to incorporate the concept of change point problem with two types of imperfect debugging in software reliability. All these models have been validated and verified using real data sets. Parameter estimates, comparison results and goodness of fit curves have also been presented.

7. FUTURE SCOPE

In future, we will try to develop more models in the same line by using normal, weibul and gamma distribution functions. Models can be extended for multiple change-points problem.

REFERENCES

[1] Pham, H.; "*Software Reliability*", Wiley Encyclopedia of Electrical and Electronics Engineering, Springer-Verlag Singapore, 2000.

[2] Bittanti, S., Bolzern P., Pedrotti E., Pozzi, N. and Scattolini R.; "*A flexible modeling approach for software reliability growth*", Goos, G. and Harmanis, J.; (Ed.), Software Reliability Modelling and Identification, Springer Verlag, Berlin, Vol. 341, pp. 101-140, 1988.

[3] Goel, A.L. and Okumoto, K.; "T*ime dependent error detection rate model for software reliability and other performance measure*", IEEE Transactions on Reliability, Vol. R 28(3), 1979.

[4] Jelinski, Z. and Moranda, P.B.; "*Software reliability research*", Statistical Computer Performance Evaluation, Freiberger W. (ed), New York, Academic Press, 1972.

[5] Kapur, P.K. and Garg, R.B.; "*A Software Reliability Growth Model for an Error Removal Phenomenon*", Software Engineering Journal, Vol. 7(4), pp. 291-294, 1992.

[6] Yamada, S., Obha M. and Osaki S.; "*S-shaped software reliability growth modeling for software error detection*", IEEE Transaction on Reliabilty, Vol. R-32(5), pp. 475-484, 1983.

[7] Ohba, M. and Yamada, S.; "*S-shaped Software Reliability Growth Model*", International Colloquium on Reliability and Maintainability, 4th, Tregastel, France, Proceedings of the 4th International Conference on Reliability and Maintainability, pp. 430-436, 1984.

[8] Kapur, P.K. and Garg, R.B.; "*A Software reliability growth model under imperfect debugging*", R.A.I.R.O, Vol. 24, pp. 295-305, 1990,

[9] Zhang, X., Teng, X. and Pham, H.; "*Considering Fault Removal Efficiency in Software Reliability Assessment*", IEEE Transaction on Systems, Man and Cybernetics-Part A: Systems and Humans, Vol. 33(1), pp. 114-120, 2003.

[10] Kapur, P.K., Kumar, D., Gupta, A. and Jha, P.C.; "*On How to Model Software Reliability Growth in the Presence of Imperfect Debugging and error Generation*", Proceedings of 2nd International Conference on Reliability and safety Engineering, pp. 515-523, 2006.

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

[12] Zhao, M.; "*Change-point Problems in Software and Hardware Reliabilty*", Communications in Statistics-Theory and Methods, Vol. 22(3), pp. 757-768, 1993.

[13] Huang, C.; "*Performance Analysis of Software Reliability Growth Models with Testing Effort and Change-Point*", Journal of Systems and Software, Vol. 76(2), pp. 181-194, 2005.

[14] Shyur, H.; "*A stochastic software reliability model with imperfect debugging and change-point*", The Journal of System and Software, Vol. 66, pp. 135-141, 2003.

[15] Kapur, P.K., Singh, V.B., Sameer A. and Yadavalli, V.S.S.; "*Software Reliability Growth Model*

with Change-Point and Effort Control Using a Power Function of Testing Time", International Journal of Production Research, Vol. 46(3), pp. 771-787, 2008.

[16] Kapur, P.K., Gupta, A., Shatnawi, O. and Yadavalli, V.S.S.; "*Testing Effort Control Using Flexible Software Reliability Growth Model with Change Point*", International Journal of Performability Engineering-Special issue on Dependability of Software/ Computing Systems, Vol. 2(3), pp. 245-263, 2006.

[17] Goswami, D.N., Khatri, Sunil K. and Kapur R.; "*Discrete Software Reliability Growth Modeling for Errors of Different Severity incorporating Change-Point Concept*", International Journal of Automation and Computing, Vol. 4(4), pp. 396-405, 2007.

[18] Kapur, P.K., Singh, V.B. and Anand S.; "*Software Reliability Growth Model of Fielded Software Based on Multiple Change-Point Concept Using a Power Function of Testing Time*", Quality Reliability and Infocom Technology, (Eds. Kapur P.K. and Verma A.K.), MacMillan India Ltd., pp. 171-178, 2007.

[19] Yamada, S., Kimura M., Tanaka, H. and Osaki S.; "*Software reliability measurement and assessment with stochastic differential equations*", IEICE Transactions on Fundamentals of Electronics and Computer Sciences, Vol. E77-A(1), pp. 109-116, 1994.

[20] Yamada, S., Nishigaki, A. and Kimura, M.; "*A Stochastic Differential Equation Model for Software Reliability Assessment and its Goodness of Fit*", International Journal of Reliability and Applications, Vol. 4(1), pp. 1-11, 2003.

[21] Lee, C.H., Kim, Y.T. and Park, D.H.; "*S-Shaped Software Reliability Growth Models derived from Stochastic Differential Equations*", IIE Transactions, Vol. 36(12), pp. 1193-1199, 2004.

[22] Yamada, S. and Tamura, Y.; "*A Flexible Stochastic Differential Equation Model in Distributed Development Environment*", European Journal of Operational Research, Vol. 168(1), pp. 143-152, pp. 143-152, 2006.

[23] Kapur, P.K., Singh, V.B. and Anand S.; "*Effect of Change-Point on Software Reliability Growth Models Using Stochastic Differential Equations*", published in the proceedings of 3rd International Conference on Reliability and Safety Engineering, (Eds. Misra, R.B., Naikan, V.N.A., Chaturvedi, S.K. and Goyal, N.K.), (INCRESE-2007), Udaipur, pp. 320-333, 2007.

[24] Oksendal, B.K.; "*Stochastic Differential Equations-An Introduction with Applications*", Springer, 2003.

[25] Anand, S.; "*Generalized Framework in software Reliability Modeling*", Ph.d Thesis, university of Delhi, delhi, 2010.

[26] Kumar, A.; "*A Study in Software Reliability Growth Modelling Under Distributed Development Environment*", Ph.D. Thesis Department of Operational Research, University of Delhi, Delhi, 2007.

[27] Brooks, W.D. and Motley R.W.; "*Analysis of Discrete Software Reliability Models-Technical Report (RADC-TR-80-84)*", Rome Air Development Center, New York, 1980.