International Journal of Scientific Engineering and Applied Science (IJSEAS) – Volume-3, Issue-01 ,January 2017
ISSN: 2395-3470
www.ijseas.com

SDE Based Software Reliability Growth Model Amalgamating Learning Factor on Fault of Different Severity with Multiple Release

Suneeta Bhati[1], A.R.Prassanan[2], Jagvinder Singh[2], Yogesh Sharma[1]
[1]Department of Mathematics, Jodhpur National University, Rajasthan, India
[2]Department of Mathematics, Maharaja Agersen College, University of Delhi, Delhi, India

Abstract

Software reliability is the possibility of the failure free operation of software in a given period of time under some certain conditions. It is assumed in many researches available in literature, that a similar testing effort is needed on each debugging effort. Yet, in usance different types of faults may require different amounts of testing effort for their detection and removal. Thereupon, faults are sorted into three categories on the substratum of severity –simple, had and complex.

As the size of software system is large and the number of faults detected during the testing phase because large, so the change of the number of faults that are detected and removed through each debugging becomes sufficiently small compared with the initial fault content at the beginning of the testing phase. In such circumstances, we can model the software fault detection process as a stochastic process with continuous state space.

In this paper, we propose a new software reliability growth model based on It$\hat{}$o type of stochastic differential equation. We consider SDE-base Goel, Okumoto model for simple faults, Yamada s-shaped model for hard faults, and generalized Erlang model for complex faults with different types of learning factor on various types of faults on multiple release.

Keywords: *SDE-srgm, multirelease, faults of different severity-simple, hard and complex, different learning functions.*

1. Introduction

The Software reliability engineering is velociously flourishing field. The intricacy of business software application is also increasing due to high cost of fixing failures, safety concerns and legal liabilities; consequently, it urges the organization to actualize reliable software. There are multifarious methodologies to consummate Software that is reliable. Software reliability engineering (SRE) addresses all these issues, from design to testing to maintenance phases.

The software reliability growth model (SRGM) is a tool of SRE that can be bestow to assess the software quantatively, bechance test status, schedule status and analyze the changes in reliability performance [6]. The software consists of differential type of faults and each fault requires distinct strategies and different amounts of testing effort to efface it.

Ohba [4] refined the Goel –Okumoto model by assuming that the faults detection/removal rate increase with time and that there are two types of faults in the software. SRGM proposed by Bittani et al [13] and Kapur and Garg [6] has similar forms as that of Ohba [4] but is developed under different set of assumptions. Bittani et al [13] proposed an SRGM exploiting the fault removal (exposure) rate during the initial and final time echoes of testing. Whereas, Kapur and Garg [6] describe a fault removal phenomenon where they assumes that during a removal process of a fault some of the additional faults might be removed without these faults causing any failure . These models can describe both exponential and s-shaped growth curves and therefore are termed as flexible models [13, 6, 4].

Ohba [4] proposed the Hyper exponential SRGM, assuming that software consist of different modules. Each module has its idiosyncrasy and ergo, the faults detected in specific modules have their own peculiars. Hence, the fault removal rate for each module is not the similar .He contemplates that fault extraction procedure for each module is modeled distinctly and the total fault removal phenomenon is the addition of the fault removal mechanism of all the modules.

Kapur et al [6] proposed an SRGM with three types of faults. The first type is modeled by an Exponential model of Goel and Okumoto [1]. The second type is modeled by delayed s-shaped model of Yamada et al [14]. The third type is modeled by at three–stage Erlang model proposed by Kapur et al. [14]. The total removal phenomenon is again modeled by the superposition of the three SRGM's [6, 8]. Later they extended their model to cater for more types of faults [11] by incorporating logistic rate during the removal process.

Faults are detected and removed, after regress testing before the software is divulged into the market. Due to deduction of new faults by the emptor, companies release an updated version of the system. Yamada et al. [15] proposed a simple software reliability growth model to describe the fault detection process during the testing phase by applying Ito type of stochastic differential equation (SDE).In this paper, we utilize SDE to represent fault detection rate that incorporates an irregular fluctuations. We proposed a model that includes three different types of faults, for example, simple, hard and complex. Fault detection rate for simple, hard and complex faults are assumed to be time dependent that can incorporates learning as the testing progress on multiple release. The proposed model is applied on four multiple release.

2. Assumptions:

a. The fault detection / correction are modeled by on-homogenous process (NHPP/).
b. The number of faults detected at any time is proportional to the remaining number of faults is proportional to the remaining number of faults in the software.
c. The number of faults in the beginning of the testing phase is finite.
d. The software faults detection process modeled as a stochastic process with a continuous state space.
e. The number of faults (simple, hard and complex) in the software system gradually decreases as the testing procedure go on.
f. Software is subjects to failure during execution caused by faults remaining in the software.
g. The faults existing in the software are of three types simple, hard and complex. They are distinguished by the amount of effect needed to remove them.
h. During the fault isolation, no new fault is introduced into the system and faults are debugged perfectly.

3. Notations:

$M(t)$	Number of faults detected during the testing time t and is a random variable.
$E(m(t))$	The mean value function or the expected number of faults detected or removed by time t.
$F(t)$	Probability distribution function.
$F_{ij}(t)$	Probability distribution function for i^{th} release and j^{th} type of faults (i=1to4)
a	Total initial faults content in the software.
b_{ij}	Fault detection rate for type (j=1to3) release in each release (i=1to 4).
σ_i	Positive constant that represents the magnitude of the irregular fluctuation.
$\gamma(t)$	Standard Gaussian white noise.
β_i	Logistic learning constant for i^{th} release (i=1 to 4)
α_i	Linear learning constant for i^{th} release (i=1to 4)
c_i	Learning constant for i^{th} release (i=1to 4)
t_{i-1}	Time for i^{th} release (i= 1to 4)
a_i	Initial fault content in the software.
p_i	Fraction of new simple faults introduced in i^{th} release removed by new simple fault removal rate.
p_i'	Fraction of new hard faults introduced in i^{th} release removed by new hard fault removal rate.
$(1-p_i-p_i')$	Fraction of new complex fault introduced in i^{th} release removed by new complex fault removal rate.

International Journal of Scientific Engineering and Applied Science (IJSEAS) – Volume-3, Issue-01 ,January 2017
ISSN: 2395-3470
www.ijseas.com

λ_{ik}	Fraction of previous kth release simple fault removed by new ith release simple fault removal rate.
λ'_{ik}	Fraction of previous kth release simple fault removed by new ith release hard fault removal rate.
$(1-\lambda_{ik}-\lambda'_{ik})$	Fraction of previous kth release simple fault by new ith release complex fault removal rate.
q_{ik}	Fraction of previous kth release hard fault removed by new ith release hard fault removal rate.
$(1-q_{ik})$	Fraction of previous kth release hard fault removed by new ith release complex fault removal rate.

4. Acronyms

DS	Data Set
$R2$	Coefficient of Multiple Determinations
SPSS	Statistical Package for Social Sciences
MSE	Mean Square Fitting Error
PE	Prediction Error
RMSPE	Root Mean Square Prediction Error
FDR	Fault Detection Rate

5. SDE Based Modeling Of Up-Gradation For Each Release

In this section, we formulate software reliability growth models by applying Itô type stochastic differential equations that incorporates three different types of learning functions. Since the faults in the software systems are detected and eliminated during the testing phase, the number of faults remaining in the software system moderately decreases as the testing process goes on.

Let $\{m(t), t \geq 0\}$ be a random variable which represents the number of software faults detected in the software system up to testing time t. Suppose that m (t) takes on continuous real value. The NHPP models have treated the software faults detection process in the testing phase as discrete state space. So the corresponding differential equation is given by [11, 12, 10, 5]:

$$\frac{dm(t)}{dt}=\frac{f(t)}{1-F(t)}\left(a-m(t)\right)$$

It might happen that the rate is not known completely, but subject to some random environmental effect, so that we have:

$$r(t)=\frac{f(t)}{1-F(t)}+"noise"$$

Where, r (t) is the time dependent fault detection/correction rate.

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

$$\frac{dm(t)}{dt}=\left[\frac{f(t)}{1-F(t)}+\sigma\gamma(t)\right]\left(a-m(t)\right)$$

The above equation can be extended to the following stochastic differential equation of a $\hat{ito}$ type:

$$dm(t)=\left[\frac{f(t)}{1-F(t)}-\frac{\sigma^2}{2}\right]\left(a-m(t)\right)dt$$
$$+\sigma\left(a-m(t)\right)dW(t)$$

Where $w(t)$ is a one-dimensional Wiener process, which is formally defined as an integration of the white noise $\gamma(t)$ with respect to time t. Using the fact that the wiener process $w(t)$, is a Gaussian process and has the following properties:

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

And on applying initial condition m (0) =0; we get m (t) as follows:

$$m(t)=a[1-(1-(F(t))e^{-\sigma W(t)}]$$

As we know that the Brownian motion or wiener Process follows normal distribution. The density function of w (t) is given by:

$$f(w(t))=\frac{1}{\sqrt{2\pi t}}\exp\left\{-\frac{(w(t))^2}{2t}\right\}.$$

Thus the mean number of detected fault is given as:

$$m^*(t) = E(m(t)) = a[1-(1-(F(t))e^{\frac{\sigma^2 t}{2}}] \quad (1)$$

In this paper, we consider three different detection rates. We assume that simple faults are removed by exponential by incorporating learning function as a function of time. And hard faults are removed using a two stage fault removal phenomenon i.e. by learning function as a function of time and complex faults by using three stages Erlang method incorporating the learning function as a function of time [4].

5.1. Simple faults are modeled as:

$$\frac{dm(t)}{dt} = b(t)(a - m(t))$$

$$b(t) = \frac{\alpha + b^2 t}{1 + bt}$$

$$m(t) = a(1-(1+bt)^{(1-\frac{\alpha}{b})} e^{-bt})$$

$$m^*(t) = E(m(t)) = a(1-(1+bt)^{(1-\frac{\alpha}{b})} e^{(-bt+\sigma^2\frac{t}{2})})$$

5.2. Hard faults are modeled as:

Xie et al incorporated the concept of learning factor in the model developed by Goel and Okumoto [2], they have assumed that learning function is proportional to the experience of the tester which augments with time. In this paper, we have assumed different learning function in detection rate and correction process. We had assumed that the testing phase is a two stage process. For first stage of testing process the mean number of faults detection $m_d(t)$, is proportional to the mean number of undetected faults remaining in the software and can be expressed by following differential equation.

$$m_d'(t) = b(t)(a - m_d(t))$$

Where, $b(t) = \dfrac{\alpha + \beta t}{1 + bt}$

Solving the equation (1) with initial conditions $m_d(0) = 0$

We obtain,

$$m_d(t) = a(1-(1+bt)^{(\frac{\beta}{b^2}-\frac{\alpha}{b})} e^{-\frac{\beta}{b}t})$$

It can be observed that as $t \to \infty$, $b(t) \to \dfrac{\beta}{b}$

In the second stage, the fault correction rate is proportional to the mean number of faults detected but not yet corrected faults remaining in the system. In this stage fault correction rate is assumed as logistic learning function and it can be expressed in terms of the differential equation as:

$$\frac{dm_c(t)}{dt} = b(t)(m_d(t) - m_c(t))$$

where,

$$b(t) = \frac{(\frac{\beta}{b})}{1 + ce^{(-\frac{\beta}{b}t)}}$$

Solving equation (2), with initial conditions $m_c(t=0) = 0$,

The mean number of faults corrected is given by,

$$m_c(t) = a(1 - \{(1 + c + ((\frac{((1+bt)^{(\frac{\beta}{b^2}-\frac{\alpha}{b}+1)} - 1)}{(\frac{\beta}{b^2} - \frac{\alpha}{b} + 1)})(\frac{\beta}{b^2}))) / (1 + ce^{(-\frac{\beta}{b}t)})\}e^{(-\frac{\beta}{b}t)})$$

$$m^*(t) = E(m(t)) = a(1 - \{(1 + c + (\frac{((1+bt)^{(\frac{\beta}{b^2}-\frac{\alpha}{b}+1)} - 1)}{(\frac{\beta}{b^2} - \frac{\alpha}{b} + 1)})(\frac{\beta}{b^2})) / (1 + ce^{-\frac{\beta}{b}t})\}e^{(-\frac{\beta}{b}t + \frac{\sigma^2 t}{2})})$$

5.3. Complex fault:

complex faults is modeled as a three stage process to represent the severity of complex faults assuming fault removal rate per remaining fault b(t) to be logistic function to describe the learning of the testing team

$$\frac{d(m_f(t))}{dt} = b(a - m(t))$$

$$\frac{d(m_i(t))}{dt} = b(m_f(t) - m_i(t))$$

$$\frac{d(m_r(t)}{dt} = b(t)(m_i(t) - m_r(t))$$

Where,

$$b(t) = \frac{c}{1 + ce^{-bt}}$$

Thus,

$$m(t) = a(\frac{(1-(1+bt+\frac{b^2t^2}{2})e^{(-bt)})}{1+ce^{-bt}})$$

And,

$$m^*(t) = E(m(t)) = a(1 - (\frac{(1 + c + bt + \frac{b^2t^2}{2})}{1 + ce^{-bt}})e^{(-bt+\frac{\sigma^2 t}{2})}))$$

6. Modeling fault removal process for multiple software release

6.1 Release 1*:*

In release 1, simple faults are removed exponentially by incorporating the learning function as a function of time by testing team, simple faults by exponential model of Goel-Okumoto, hard faults by Yamada's s-shaped model amalgamate the learning function as a function of time and complex faults by Erlang method imbibing the learning function.

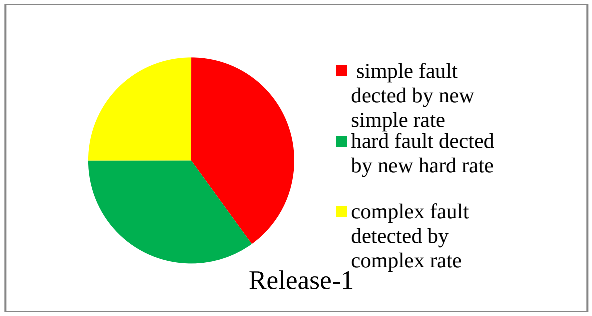
$$M_1(t) = p_1 a_1 F_{11}(t) + p_1' a_1 F_{12}(t)$$
$$+(1 - p_1 - p_1')a_1 F_{13}(t) \quad ,$$
$$0 < t < t_1$$

$$F_{11} = (1 - (1 + bt)^{(1-\frac{\alpha}{b})} e^{(-bt+\sigma^2\frac{t}{2})}) \text{...................}simplefaults$$

$$F_{12} = \frac{1}{(1+ce^{(-\frac{\beta}{b}t)})}(1 - (1 + (\frac{((1+bt)^{(\frac{\beta}{b^2}-\frac{\alpha}{b}+1)} - 1)}{(\frac{\beta}{b^2} - \frac{\alpha}{b} + 1)}(\frac{\beta}{b^2})))e^{(-\frac{\beta}{b}t+\frac{\sigma^2 t}{2})}) \text{......}Hardfaults$$

$$F_{13} = (1 - (\frac{(1 + c + bt + \frac{b^2t^2}{2})}{1 + ce^{-bt}})e^{(-bt+\frac{\sigma^2 t}{2})})) \text{................}complexfaults$$

The following pie–chart for subsequent for releases-1 is:



6.2 Release-2:

Accession of a few novel functionality to the software beget to modification of the code. These neoteric specifications in the code lead to codicil of the fault content. Now the testing team starts testing the upgraded system, apart from this the testing team heed dependency and effect of adding new functionalities with existing system. Amid testing the newly formed code, there is always a possibility that the testing team may find some faults (simple, hard and complex) which were present in formerly developed code. In this period left over simple faults $p_1 a_1(1 - F_{11}(t_1))$, left over hard fault $p_1 a_1(1 - F_{12}(t_1))$ and left over complex fault $(1 - p_1 - p_1')a_1(1 - F_{13}(t_1))$ of the first iteration interacts with new simple, hard and complex detection /correction rate. A fraction λ_{21} of remaining simple faults from first version interacts with new simple rate and a fraction λ_{21}' of remaining simple fault from first version

International Journal of Scientific Engineering and Applied Science (IJSEAS) – Volume-3, Issue-01 ,January 2017
ISSN: 2395-3470
www.ijseas.com

interacts with new hard rate whereas the remaining fraction $(1-\lambda_{21}-\lambda'_{21})$ of faults from first version interacts new detection/correction rate. Similarly, for the remaining hard faults $p_1 a_1(1-F_{12}(t_1))$ of the first iteration interacts with new detection/ correction rate. A fraction of $q_1 p_1 a_1(1-F_{12}(t_1))$ interacts with new hard rate and remaining fraction $(1-q_1)p_1 a_1(1-F_{12}(t_1))$ with new complex rate. Similarly, the remaining complex fault from first iteration $(1-p_1-p'_1)a_1(1-F_{13}(t_1))$ which is interacted with new complex detection/ correction rate. In addition, faults are generated due to the enhancement of the features, a fraction of these faults are also removed during the testing with new detection rate i.e. $F_{21}(t-t_1)$ for simple faults and $F_{22}(t-t_1)$ for hard faults and $F_{23}(t-t_1)$ for complex faults. The change in the fault detection is due to change in time, change in the complexity due to new features, change in testing strategies etc. The resulting equation can be written as:

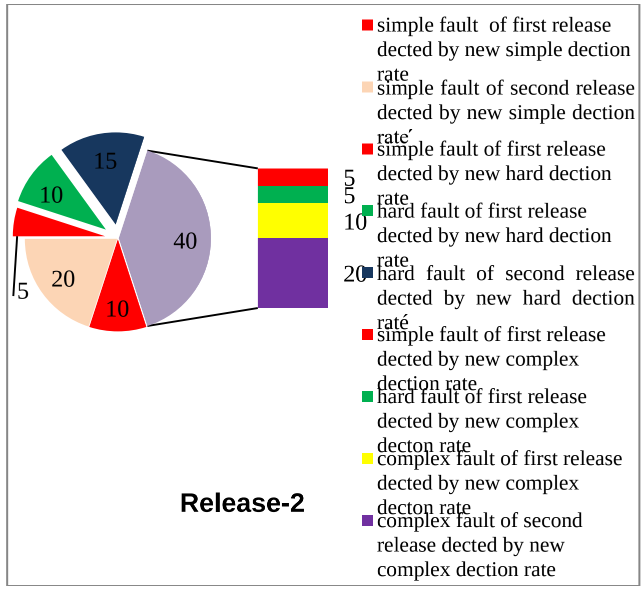
$$
\begin{aligned}
M_2(t) &= p_2 a_2 F_{21}(t) + p'_2 a_2 F_{22}(t-t_1) + (1-p_2-p'_2) \\
& a_2 F_{23}(t-t_1) + \lambda_{21} p_1 a_1 (1-F_{11}(t_1)) F_{21}(t-t_1) \\
& +\lambda'_{21} p_1 a_1 (1-F_{11}(t_1) F_{22}(t-t_1) + (1-\lambda_{21}-\lambda'_{21}) \\
& p_1 a_1 (1-F_{11}(t_1)) F_{23}(t-t_1) + q_{21} p'_1 a_1 (1-F_{12}(t_1)) \\
& F_{22}(t-t_1) + (1-q'_{21}) p'_1 a_1 (1-F_{12}(t_1)) F_{23}(t-t_1) \\
& +(1-p_1-p'_1) a_1 (1-F_{13}(t_1)) F_{23}(t-t_1)
\end{aligned}
$$

$$t_1 < t < t_2$$

$$F_{21} = (1-(1+bt)^{(1-\frac{\alpha}{b})} e^{(-bt+\sigma^2\frac{t}{2})}) \text{.....................} simplefaults$$

$$F_{22} = (1-((1+c+((\frac{((1+bt)^{(\frac{\beta}{b^2}-\frac{\alpha}{b}+1)}-1)}{(\frac{\beta}{b^2}-\frac{\alpha}{b}+1)})(\frac{\beta}{b^2}))) / (1+ce^{(-\frac{\beta}{b}t)}))e^{(-\frac{\beta}{b}t+\frac{\sigma^2 t}{2})}) \text{......} Hardfaults$$

$$F_{23} = (1-(\frac{(1+c+bt+\frac{b^2 t^2}{2})}{1+ce^{-bt}})e^{(-bt+\frac{\sigma^2 t}{2})})) \text{.................} complexfaults$$



6.3. Release-3:

Similarly for release 3, we consider faults generated in third release and remaining number of simple, hard and complex faults from the second release. As the parameters are more in the proposed model in release -3 compare to no. of available data points in tendem data .So, we increase the data points by taking series mean of data points of available no. of faults detected in tendem data which is 37.92. The proposed model and the corresponding mathematical equation can be represented as follows:

International Journal of Scientific Engineering and Applied Science (IJSEAS) – Volume-3, Issue-01 ,January 2017
ISSN: 2395-3470
www.ijseas.com

$$t_2 < t < t_3$$

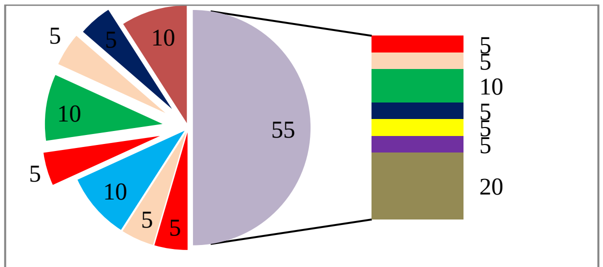
Faults which are removed by New simple detection rate :	Faults which are removed by New hard detection rate:
Leftover simple faults of 1st release	Leftover simple fault of 1st release
Leftover simple fault of 2nd release	Left over hard fault of 1st release
New simple fault of 3rd release	Left over simple fault of 2nd release
	Left over hard fault of 2nd release
	New hard fault of 3rd release
Faults which are removed by new complex detection rate:	
Leftover simple faults of 1st release	Left over hard fault of 1st release
Leftover simple faults of 2nd release	Left over hard fault of 2nd release
Left over complex fault of 1st release	Leftover complex fault of 2nd release
New complex fault of 3rd release	

$$
\begin{aligned}
M_3(t) &= p_3 a_3 F_{31}(t-t_2) + p_3' a_3 F_{32}(t-t_2) \\
&+(1-p_3-p_3')a_3 F_{33}(t-t_2) + \lambda_{32}\{p_2 a_2 \\
&(1-F_{21}(t_2-t_1))\}F_{31}(t-t_2) + \lambda_{32}'\{p_2 a_2 \\
&(1-F_{21}(t_2))\}F_{32}(t-t_2) + (1-\lambda_{32}-\lambda_{32}') \\
&p_2 a_2 (1-F_{21}(t_2-t_1))F_{33}(t-t_2) + q_{32}\{p_2' a_2 \\
&(1-F_{22}(t_2-t_1))\}F_{32}(t-t_2) + (1-q_{32}')\{p_2' a_2 \\
&(1-F_{22}(t_2-t_1))\}F_{33}(t-t_2) + (1-p_2-p_2')a_2 \\
&(1-F_{23}(t_2-t_1))F_{33}(t-t_2) + \lambda_{31}\{p_1 a_1 \\
&(1-F_{11}(t_1)(1-(\lambda_{21}F_{21}(t_2-t_1) + \lambda_{21}' F_{22}(t_2-t_1) \\
&+(1-\lambda_{21}-\lambda_{21}')F_{23}(t_2-t_1)))\}F_{31}(t-t_2) + \lambda_{31}'\{p_1 a_1 \\
&(1-F_{11}(t_1))(1-(\lambda_{21}F_{21}(t_2-t_1) + \lambda_{21}' F_{22}(t_2-t_1) \\
&+(1-\lambda_{21}-\lambda_{21}')F_{23}(t_2-t_1)))\}F_{32}(t-t_2) \\
&+(1-\lambda_{31}-\lambda_{31}')\{p_1 a_1 (1-F_{11}(t_1))(1-(\lambda_{21}F_{21}(t_2-t_1) \\
&+\lambda_{21}' F_{22}(t_2-t_1) + (1-\lambda_{21}-\lambda_{21}')F_{23}(t_2-t_1)))\} \\
&F_{33}(t-t_2) + q_{31}\{p_1 a_1 (1-F_{12}(t_1))(1-(q_{21}F_{22}(t_2-t_1) \\
&+(1-q_{21}')F_{23}(t_2-t_1)))\}F_{32}(t-t_2) + (1-q_{31})
\end{aligned}
$$

$$F_{31} = (1-(1+bt)^{(1-\frac{\alpha}{b})} e^{(-bt+\sigma^2\frac{t}{2})})\text{..................}simplefaults$$

$$F_{32} = (1-((1+c+((\frac{((1+bt)^{(\frac{\beta}{b^2}-\frac{\alpha}{b}+1)}-1)}{(\frac{\beta}{b^2}-\frac{\alpha}{b}+1)})(\frac{\beta}{b^2})))/(1+ce^{(-\frac{\beta}{b}t)}))e^{(-\frac{\beta}{b}t+\frac{\sigma^2 t}{2})})\text{......}Hardfa$$

$$F_{33} = (1-(\frac{(1+c+bt+\frac{b^2t^2}{2})}{1+ce^{-bt}})e^{(-bt+\frac{\sigma^2 t}{2})}))\text{................}complexfaults$$



Release-3

6.4. Release-4:

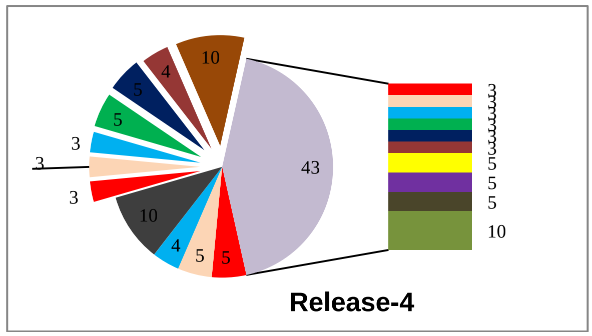
The procedure of inclusion of new functionalities is an ongoing process. These add-ons keep on occurring till software is present in the market. As a result these proceedings help in ameliorableness of software as well as in accretion of reliability of the product because bounteous faults are removed when testing and integration of code is rendered. We have discussed a case when the new features are added in the software for the third time:

$$
\begin{aligned}
M_4(t) &= p_4 a_4 F_{41}(t-t_3) + p'_4 a_4 F_{42}(t-t_3) + (1-p_4-p'_4) \\
& a_4 F_{43}(t-t_3) + \lambda_{43} p_3 a_3 (1-F_{31}(t_3-t_2))F_{41}(t-t_3) \\
& +\lambda'_{43} p_3 a_3 (1-F_{31}(t_3-t_2)F_{42}(t-t_3) + (1-\lambda_{43}-\lambda'_{43}) \\
& p_3 a_3 (1-F_{31}(t_3-t_2))F_{43}(t-t_3) + q_{43} p'_3 a_3 (1-F_{32}(t_3-t_2)) \\
& F_{42}(t-t_3) + (1-q_{43}) p'_3 a_3 (1-F_{32}(t_3-t_2))F_{43}(t-t_3) \\
& +(1-p_3-p'_3) a_3 (1-F_{33}(t_3-t_2))F_{43}(t-t_3) + \lambda_{42}\{p_2 a_2 \\
& (1-F_{21}(t_2-t_1))(1-(\lambda_{32}F_{31}(t-t_2)+\lambda'_{32}F_{32}(t-t_2) \\
& +(1-\lambda_{32}-\lambda'_{32})F_{33}(t-t_2)))\}F_{41}(t-t_3) + \lambda'_{42}\{p_2 a_2 \\
& (1-F_{21}(t_2-t_1))(1-(\lambda_{32}F_{31}(t-t_2)+\lambda'_{32}F_{32}(t-t_2) \\
& +(1-\lambda_{32}-\lambda'_{32})F_{33}(t-t_2)))\}F_{42}(t-t_3) + (1-\lambda_{42}-\lambda'_{42}) \\
& \{p_2 a_2 (1-F_{21}(t_2-t_1))(1-(\lambda_{32}F_{31}(t-t_2)+\lambda'_{32}F_{32}(t-t_2) \\
& +(1-\lambda_{32}-\lambda'_{32})F_{33}(t-t_2)))\}F_{43}(t-t_3) + q_{42}\{p'_2 a_2 \\
& (1-F_{22}(t_2-t_1))(1-(q_{32}F_{32}(t_3-t_2)+(1-q_{32}) \\
& F_{33}(t_3-t_2)))\}F_{42}(t-t_3) + (1-q_{42})\{p'_2 a_2 \\
& (1-F_{22}(t_2-t_1))(1-(q_{32}F_{32}(t_3-t_2)+(1-q_{32}) \\
& F_{33}(t_3-t_2)))\}F_{43}(t-t_3) + (1-p_2-p'_2)a_2 \\
& (1-F_{23}(t_2-t_1))(1-F_{33}(t_3-t_2))F_{43}(t-t_3) + \lambda_{41} \\
& \{p_1 a_1 (1-F_{11}(t_1))(1-(\lambda_{21}F_{21}(t_2-t_1)+\lambda'_{21}F_{22}(t_2-t_1) \\
& +(1-\lambda_{21}-\lambda'_{21})F_{23}(t_2-t_1)))(1-(\lambda_{31}F_{31}(t_3-t_2) \\
& +\lambda'_{31}F_{32}(t_3-t_2)+(1-\lambda_{31}-\lambda'_{31})F_{33}(t_3-t_2)))\} \\
& F_{41}(t-t_3) + \lambda'_{41}\{p_1 a_1 (1-F_{11}(t_1))(1-(\lambda_{21}F_{21}(t_2-t_1) \\
& +\lambda'_{21}F_{22}(t_2-t_1)+(1-\lambda_{21}-\lambda'_{21})F_{23}(t_2-t_1))) \\
& (1-(\lambda_{31}F_{31}(t_3-t_2)+\lambda'_{31}F_{32}(t_3-t_2)+(1-\lambda_{31}-\lambda'_{31}) \\
& F_{33}t_3-t_2)))\}F_{42}(t-t_3) + (1-\lambda_{41}-\lambda'_{41})\{p_1 a_1 (1-F_{11}(t_1) \\
& (1-(\lambda_{21}F_{21}(t_2-t_1)+\lambda'_{21}F_{22}(t_2-t_1)+(1-\lambda_{21}-\lambda'_{21}) \\
& F_{23}(t_2-t_1)))(1-(\lambda_{31}F_{31}(t_3-t_2)+\lambda'_{31}F_{32}(t_3-t_2) \\
& +(1-\lambda_{31}-\lambda'_{31})F_{33}(t_3-t_2)))\}F_{43}(t-t_3) + q_{41}\{p'_1 a_1 \\
& (1-F_{11}(t_1))(1-(q_{21}F_{22}(t_2-t_1)+(1-q_{21})F_{23}(t_2-t_1))) \\
& (1-(q_{31}F_{32}(t_3-t_2)+(1-q_{31})F_{33}(t_3-t_2)))\} \\
& F_{42}(t-t_3) + (1-q_{41})\{p'_1 a_1 (1-F_{11}(t_1))(1-(q_{21}F_{22}(t_2-t_1) \\
& +(1-q_{21})F_{23}(t_2-t_1)))(1-(q_{31}F_{32}(t_3-t_2)+(1-q_{31}) \\
& F_{33}(t_3-t_2)))\}F_{43}(t-t_3) + (1-p_1-p'_1)a_1 \\
& (1-F_{13}(t_1))(1-F_{23}(t_2-t_1))(1-F_{33}(t_3-t_2))F_{43}(t-t_3)
\end{aligned}
$$

$$F_{41} = (1-(1+bt)^{(1-\frac{\alpha}{b})} e^{(-bt+\sigma^2\frac{t}{2})})\ldots\ldots\ldots\ldots\ldots\ldots simplefaults$$

$$F_{42} = (1-((1+c+((\frac{(1+bt)^{(\frac{\beta}{b^2}-\frac{\alpha}{b}+1)}-1}{(\frac{\beta}{b^2}-\frac{\alpha}{b}+1)})(\frac{\beta}{b^2}))) / (1+ce^{(-\frac{\beta}{b}t)}))e^{(-\frac{\beta}{b}t+\frac{\sigma^2 t}{2})})\ldots\ldots Hardfaults$$

$$F_{43} = (1-(\frac{(1+c+bt+\frac{b^2t^2}{2})}{1+ce^{-bt}})e^{(-bt+\frac{\sigma^2 t}{2})}))\ldots\ldots\ldots\ldots\ldots complexfaults$$



Faults which are removed by New simple detection rate:	Faults which are removed by New hard detection rate:
leftover simple fault of 1st release	leftover simple fault of 1st release
leftover simple fault of 2nd release	leftover simple fault of 2nd release
leftover simple fault of 3rd release	leftover simple fault of 3rd release
new simple fault of 4th release	left over hard fault of 1st release
	left over hard fault of 2nd release
	leftover hard fault of 3rd release
	new hard fault of 4th release
Faults which are removed by new complex detection rate:	
leftover simple fault of 1st release	leftover simple fault of 2nd release
leftover simple fault of 3rd release	left over hard fault of 1st release
left over hard fault of 2nd release	leftover hard fault of 3rd release
leftover complex fault of 1st rerelease	leftover complex fault of

	2nd release
■ New complex fault of 3rd release	

7. Model validation, Data Set and Data Analysis

To check the validity of the proposed model and to describe the software reliability growth, it has been tested on tandem computer four release data set. Also we have used non linear least square technique in SPSS software for estimation of parameters. Estimated value of parameters of each releases are given in **Table 1**. **Table 2** shows the comparison criterion of the four software releases. Based on data available given in Table1, the performance analysis of proposed model is measured by the four common criteria that we define as below:

7.1. Criteria for comparisons

To give quantitative comparisons, some criteria were used to judge the performance of the proposed model. Here we let n represent the sample size of selected data set, y_i represent the actual number of faults by time t_i and $m(t_i)$ represent the estimated number of faults by time t_i .in all mentioned criteria the lower value indicate less fitting error.

7.1.1 The **Bias** is defined as:

$$\text{Bias} = \sum_{i=1}^{n}\left(\frac{(\hat{m}(t_i) - y_i)}{n}\right)$$

The difference between the observation and prediction of number of failures at any instant of time i is known as $PE_{i.}$ (Prediction error). The average of PEs is known as bias. Lower the value of Bias better is the goodness of fit.

7.1.2 The **Variation** is defined as:

$$Variation = \sqrt{\frac{\left(\sum_{i=1}^{n}\left((\hat{m}(t_i) - y_i) - Bias\right)^2\right)}{(n-1)}}$$

The average of the prediction errors is called the prediction *Bias*, and its standard deviation is often used as a measure of the variation in the predictions.

7.1.3 The **Root Mean Square Prediction Error (RMSPE)** is defined as:

$$RMSPE = \sqrt{\left(Bias^2 + Variation^2\right)}$$

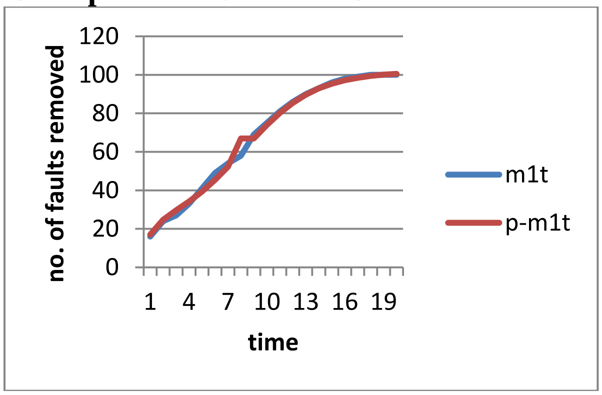
RMSPE is a measure of the closeness with which the model predicts the observation.

7.1.3 The **Mean Square Error (MSE)** is defined as:

The difference between the expected values, $\hat{m}(t_i)$ and the observed data y_i is measured by MSE as follows: $$MSE = \sum_{i=1}^{k}\frac{(\hat{m}(t_i) - y_i)^2}{k}$$

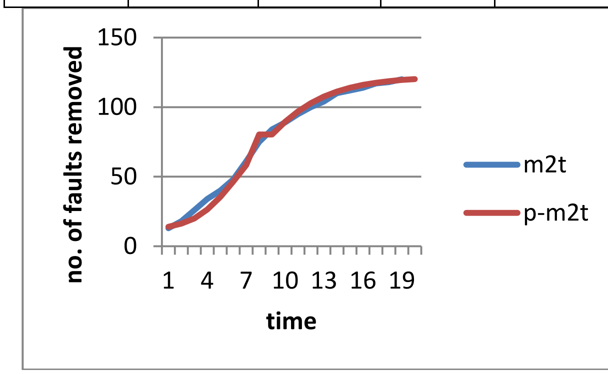
Where k is the number of observations. The lower MSE indicates less fitting error, thus better goodness of fit.

8. Graphs for four release:



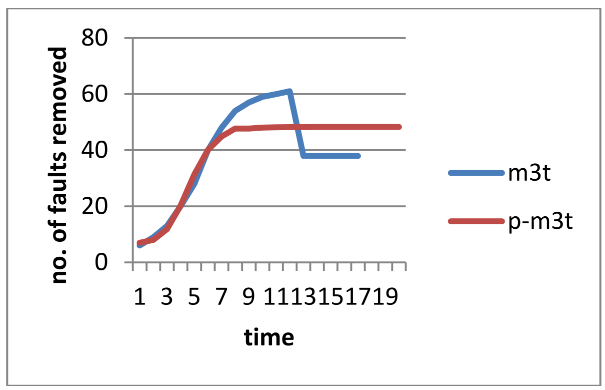
Goodness of fit curve for release-1

Table 2: Comparison Criteria				
	I	**II**	**III**	**IV**
M.S.E	**5.8635**	**10.8520**	**65.0924**	**10.7718**
Bias	**.0301**	**-0.3574**	**0.000**	**0.000**
Variation	**2.48416**	**3.364521**	**8.316287**	**3.367335**
R2	**.993**	**.992**	**.783**	**.939**
RMSPE	**2.484343**	**3.38345**	**8.316287**	**3.367335**

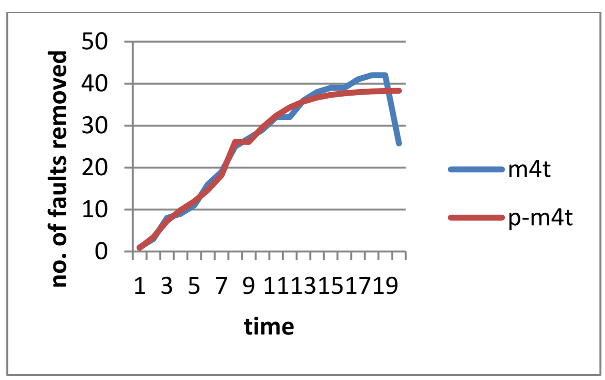


Goodness of fit curve for release-2

Table 1: Parameter Estimate				
i=1to4	1	2	3	4
a_i	194.000	20.081	45.837	38.417
b_{i1}	.005	2.376	3.298	11.606
b_{i2}	.004	.002	.002	23.280
b_{i3}	.004	.399	.002	.509
p_i	.134	.300	.139	.007
p'_i	.390	.300	.010	.223
β_i	.001	5.000	6.000	38.347
α_i	1.000	1.000	12.000	5.000
c_i	19.586	5.000	90.000	41.593
σ_i	4.657E-006	4.302E-005	.000	.000
λ_{i1}	-	.070	.100	.200
λ'_{i1}	-	.200	.200	.300
q_{i1}	-	.300	.100	.469
λ_{i2}	-	-	.100	.200
λ'_{i2}	-	-	.300	.300
q_{i2}	-	-	.400	.400
λ_{i3}	-	-	-	.100
λ'_{i3}	-	-	-	.300
q_{i3}	-	-	-	.400



Goodness of fit curve for release-3



Goodness of fit curve for release-4

Conclusion:

In this paper we have developed the SRGM incorporating stochastic differential equation of It$\hat{o}$ type using different learning functions different severity of faults on multiple release. In future we propose to develop a SRGM model of stochastic differential equation of It$\hat{o}$ type incorporating learning function on n-types of faults on four multiple releases.

References:

[1] L. Goel and K. Okumoto, "Time-dependent error-detection rate model for software reliability and other performance measures," IEEE Transactions on Reliability, vol. 28, no. 3, pp. 206–211, 1979.

[2] B. Øksendal, "Stochastic Differential Equations: An Introduction with Applications", Universitext, Springer, Berlin, Germany, 6th edition, 2003.

[3] H.Pham, System Software Reliability. Verlag, Springe, 2006.

[4] M. Ohba, "Software reliability analysis models," IBM Journal of Research and Development, vol. 28, no. 4, 1984, pp. 428–443,

[5] P. K. Kapur, S. S. Handa, Deepak Kumar, P. C. Jha. "On The Development of Flexible Discrete SRGM with Two Types of Imperfect Debugging", in 3rd National Conference; INDIACom-2009 Computing For Nation Development, February 26 – 27, 2009 .

[6] P. K. Kapur, R. B. Garg, and S. Kumar, "Contributions to Hardware and Software Reliability", World Scientific, Singapore, 1999.

[7] P. K. Kapur and R. B. Garg, "Software reliability growth model for an error-removal phenomenon," Software Engineering Journal, vol. 7, no. 4, 1992, pp. 291–294,

[8] P. K. Kapur, S. Younes, and S. Agarwala, "Generalized Erlang model with n types of faults," ASOR Bulletin, vol. 14, no. 1, 1995 pp. 5–11.

[9] P.K.Kapur, V.B.Singh, andB.Yang, "Software reliability growth model for determining fault types" in 3rd International Conference on Reliability and Safety Engineering (INCRESE '07), December 2007, pp. 334–349.

[10] P. K. Kapur, Sameer Anand, Shigeru Yamada, and Venkata S. S. Yadavalli, "Stochastic Differential Equation-Based Flexible Software Reliability Growth Model, Research Article",Hindawi Publishing Corporation Mathematical Problems in Engineering, Volume 2009, Article ID 581383, 15 pages doi:10.1155/2009/581383

[11] P.K.Kapur, A.Tandon, G.Kaur, "Multi Up-gradation Software reliability Model," 2nd international conference on reliability, safety hazard (ICRESH-2010), 2010, pp. 468-474.

[12] P.K Kapur. , Mashaalah Basirzadeh ,Shinji Inoue and Shigeru Yamda "stochastic differential equation based SRGM for errors of deferent severity with testing -effort" International Journal of Reliability, Quality and Safety Engineering ,Vol. 17, No. 3 (2010) ,pp. 179–197

[13] S. Bittanti, P. Bolzern, E. Pedrotti, and R. Scattolini, "A flexible modeling approach for software reliability growth" in Software Reliability Modeling and Identification, G. Goos and J. Harmanis, , Berlin, Germany Springer, pp. 101–140, 1998.

[14] S. Yamada, M. Ohba, and S. Osaki, "S-shaped software reliability growth models and their applications" IEEE Transactions on Reliability, vol. 33, no. 4, 1984, pp. 289–292.

[15] S. Yamada, A. Nishigaki, and M. Kimura, "A stochastic deferential equation model for software reliability assessment and its goodness of fit" International Journal of Reliability and Application, vol.4, no. 1, 2003, pp. 1–11.

[16] S. Yamada, Nishigaki A, Kimura M. "A Stochastic Differential Equation Model for Software Reliability Assessment and its Goodness of Fit" International Journal of Reliability and Applications Vol.4, No. 1, 2003, pp. 1-11.

[17] S. Yamada and Y. Tamura. "A Flexible Stochastic Differential Equation Model in Distributed Development Environment"

European Journal of Operational Research 2006, Vol. 168 pp. 143-152.

[18] Y. Tamura and S. Yamada, "A flexible stochastic deferential equation model in distributed development environment" European Journal of Operational Research, vol.168, no.1, 2005, pp.143–152.