Software Reliability Growth Model with Logistic-Exponential Test-Effort Function and Analysis of Software Release Policy.

Shaik.Mohammad Rafi [1] , Dr.K.Nageswara Rao[2] , Shaheda Akthar [3]

[1]Assoc. Professor Sri Mittapalli College Institute of Technology for Women, NH-5,Tummalapalem, Guntur, A.P,India. Tel: 09985083350

[2]Professor& H.O.D in P.V.P.S.I.T , Vijayawada affiliated to J.N.T.U, Kakinada, India. Tel: 09441037218

[3]Assoc. Professor Sri Mittapalli College of Engineering NH-5, Tummalapalem, Guntur, A.P,India. Tel: 09885834601

Abstract: software reliability is one of the important factors of software quality. Before software delivered in to market it is thoroughly checked and errors are removed. Every software industry wants to develop software that should be error free. Software reliability growth models are helping the software industries to develop software which is error free and reliable. In this paper an analysis is done based on incorporating the logistic-exponential testing-effort in to NHPP Software reliability growth model and also observed its release policy. Experiments are performed on the real datasets. Parameters are calculated and observed that our model is best fitted for the datasets.

Keywords: Software Reliability, Software Testing, Testing Effort, Non-homogeneous Poisson Process (NHPP), Software Cost.

ACRONYMS

NHPP : Non Homogeneous Poisson Process
SRGM : Software Reliability Growth Model
MVF : Mean Value Function
MLE : Maximum Likelihood Estimation
TEF : Testing Effort Function
LOC : Lines of Code
MSE : Mean Square fitting Error

NOTATIONS

m (t) : Expected mean number of faults detected in time (0,t]
λ (t) : Failure intensity for m(t)
n (t) : Fault content function
m_d (t) : Cumulative number of faults detected up to t
m_r (t) : Cumulative number of faults isolated up to t.
W (t) : Cumulative testing effort consumption at time t.
W*(t) : W (t)-W (0)
A : Expected number of initial faults
r (t) : Failure detection rate function
r : Constant fault detection rate function.
r_1 :Constant fault detection rate in the Delayed S-shaped model with logistic-Exponential TEF
r_2 :Constant fault isolated rate in the Delayed S-shaped model with logistic-Exponential TEF

1. Introduction

Software becomes crucial in daily life. Computers, commutation devices and electronics equipments every place we find software. The goal of every software industries is develop software which is error and fault free. Every industry is adopting a new testing technique to capture the errors during the testing phase. But even though some of the faults were undetected. These faults create the problems in future. Reliability is defined as the working condition of the software over certain time period of time in a given environmental conditions. Large numbers of papers are presented in this context. Testing effort is defined as effort needed to detect and correct the errors during the testing. Testing-effort can be calculated as person/ month, CPU hours and number of test cases and so on. Generally the software testing consumes a testing-effort during the testing phase [20 21].SRGM proposed by several papers incorporated traditional effort curves like Exponential, Rayleigh, and Weibull. The TEF which gives the effort required in testing and CPU time the software for better error tracking. Many papers are published based on TEF in NHPP models [4, 5, 8, 11, 120, 12, 20, 21]. All of them describe the tracking phenomenon with test expenditure.

This paper is organized in to six sections. Section 2 briefly describes the testing effort functions. Section 3 proposed the new software reliability growth model. Section 4 shows the model evaluation criteria. Section 5 & 6 describes the software release time based on software cost and reliability.

2. Software testing effort functions

Several software testing-effort functions are defined in literature. w(t) is defined as the current testing effort and W(t) describes the cumulative testing effort. The following equation shows the relation between the w(t) and W(t)

$$W(t) = \int_0^t w(t)dt \qquad (1)$$

The following are some of them

1) Exponential Testing effort function

The cumulative testing effort consumed in the time (0,t] is [20]

$$W(t) = N \times (1 - \exp[-bt]) \qquad (2)$$

2) Rayleigh Testing effort curve:

The cumulative testing effort consumed in the time (0,t] is [12,20]

$$W(t) = N \times (1 - \exp[-(b/2)\, t^{\wedge}2] \qquad (3)$$

The Rayleigh curve increases to the peak and descends gradually with decelerating rate.

3) Logistic-exponential testing-effort has a great flexibility in accommodating all the forms of the hazard rate function, can be used in a variety of problems for modeling software failure data.

The logistic-exponential cumulative TEF over time period (0,t] can be expressed as [27]

$$W(t) = \alpha \times \frac{(e^{\lambda \times t} - 1)^k}{(1 + (e^{\lambda \times t} - 1)^k)}, \qquad t>0 \qquad (4)$$

And its current testing effort is

$$w(t) = \frac{\alpha \times k \times \lambda \times e^{\lambda \times t} \times (e^{\lambda \times t} - 1)^{k-1}}{\left(1 + (e^{\lambda \times t} - 1)^k\right)^2} \quad t>0 \quad (5)$$

α is the total expenditure , k positive shape parameter and λ is a positive scale parameter

3 Software Reliability Growth Models

3.1 Software reliability growth model with logistic-exponential TEF

The following assumptions are made for software reliability growth modeling [1,8,11,20,21,22]

This paper we used logistic-exponential testing-effort curve and incorporated in the SRGM. The result shows that the SRGM with logistic-exponential testing-effort function gives better performance than other.

(i) The fault removal process follows the Non-Homogeneous Poisson process (NHPP)

(ii) The software system is subjected to failure at random time caused by faults remaining in the system.

(iii) The mean time number of faults detected in the time interval (t, t+Δt) by the current test effort is proportional for the mean number of remaining faults in the system.

(iv) The proportionality is constant over the time.

(v) Consumption curve of testing effort is modeled by a logistic-exponential TEF.

(vi) Each time a failure occurs, the fault that caused it is immediately removed and no new faults are introduced.

(vii) We can describe the mathematical expression of a testing-effort based on following

$$\frac{dm(t)}{dt} \times \frac{1}{w(t)} = r \times (a - m(t)) \qquad (6)$$

$$m(t) = a \times \left(1 - e^{-r \times (W(t) - W(0))}\right) \qquad \mathbf{(7)}$$

Substituting W(t) into Eq.(7), we get

$$m(t) = a \times \left(1 - \exp\left(-r \times \alpha \times \frac{(e^{\lambda \times t} - 1)^k}{\left(1 + (e^{\lambda \times t} - 1)^k\right)}\right)\right) \qquad (8)$$

This is an NHPP model with mean value function with the Logistic-exponential testing-effort expenditure.

Now failure intensity is given by

$$\lambda(t) = \frac{dm(t)}{dt} = a \times r \times w(t) \times e^{-r \times W(t)} \qquad (9)$$

$$\lambda(t) = \frac{a\, e^{-\frac{r\alpha(e^{\lambda t}-1)^k - \lambda t - \lambda t (e^{\lambda t}-1)^k}{1+(e^{\lambda t}-1)^k}}\, r\,\alpha\, k\, \lambda (e^{\lambda t}-1)^{k-1}}{\left(1 + (e^{\lambda t} - 1)^k\right)^2} \qquad (10)$$

The expected number of errors detected eventually is

$$m(\infty) = a \qquad (11)$$

3.2 Yamada Delayed S-shaped model with logistic-exponential testing-effort function

The delayed 'S' shaped model originally proposed by Yamada [24] and it is different from NHPP by considering that software testing is not only for error detection but error isolation. And the cumulative errors detected follows the S-shaped curve. This behavior is indeed initial phase testers are familiar with type of errors and residual faults become more difficult to uncover [1,6,15,16].

From the above steps described section 3.1, we will get a relationship between m(t) and w(t). For extended Yamada S-shaped software reliability model.

The extended S-shaped model [24] is modeled by

$$\frac{dm_d(t)}{dt} \times \frac{1}{w(t)} = r_1 \times \left[a - m_d(t)\right] \quad (12)$$

And

$$\frac{dm_r(t)}{dt} \times \frac{1}{w(t)} = r_2 \times \left[a - m_r(t)\right] \quad (13)$$

We assume $r_2 \neq r_1$ by solving 2 and 3 boundry conditions $m_d(t)=0$, we have

$$m_d(t) = a \times \left(1 - e^{\left[-r_1 \times W*(t)\right]}\right)$$

and

$$m_r(t) = a \times \left[1 - \frac{\left(r_1 \times e^{\left[-r_2 \times W*(t)\right]} - r_2 \times e^{\left[-r_1 \times W*(t)\right]}\right)}{r1 - r2}\right] \quad (14)$$

At this stage we assume $r_2 \approx r_1 \approx r$, then using 'L' Hospitals rule the Delayed S-shaped model with TEF is given by

$$m(t) \cong m_r(t) = a \times \left(1 - \left(1 + r \times W*(t)\right) \times e^{\left[-r \times W*(t)\right]}\right) \quad (15)$$

The failure intensity function for Delayed S-shaped model with TEF is given by

$$\lambda(t) = a \times r^2 \times w(t) \times W*(t) \times e^{\left[-r \times W*(t)\right]} \quad (16)$$

4) EVALUATION CRITERIA

a) The goodness of fit technique

Here we used MSE [5,11,17,23]which gives real measure of the difference between actual and predicted values. The MSE defined as

$$MSE = \sum_{i=1}^{k} \frac{\left[m(t_i) - m_i\right]^2}{k} \quad (17)$$

A smaller MSE indicate a smaller fitting error and better performance.

b) **Coefficient of multiple determinations (R^2)** which measures the percentage of total variation about mean accounted for the fitted model and tells us how well a curve fits the data. It is frequently employed to compare model and access which model provies the best fit to the data. The best model is that which proves higher R^2. that is closer to 1.

c) The predictive Validity Criterion

The capability of the model to predict failure behavior from present & past failure behavior is called predictive validity. This approach, which was proposed by [26], can be represented by computing RE for a data set

$$RE = \frac{\left(m(t_q) - q\right)}{q} \quad (18)$$

In order to check the performance of the logistic-exponential software reliability growth model and make a comparison criteria for our evaluations [14].

d) **SSE criteria:** SSE can be calculated as :[17]

$$SSE = \sum_{i=1}^{n} \left[y_i - m(t_i)\right]^2 \quad (19)$$

Where y_i is total number of failures observed at a time t_i according to the actual data and $m(t_i)$ is the estimated cumulative number of failures at a time t_i for i=1,2,…..,n.

$$PE_i = Actual(observed)_i - Predicted(estimated)_i \quad (20)$$

$$Bias = \sum_{i=1}^{n} \frac{PE_i}{n} \quad (21)$$

$$Variation = \sqrt{\sum_{i=1}^{n} \frac{(PE_i - Bias)^2}{n-1}} \quad (22)$$

$$MRE = \frac{|M_{estimated} - M_{actual}|}{M_{actual}} \quad (23)$$

5) Model Performance Analysis

DS1: the first set of actual data is from the study by Ohba 1984 [15].the system is PL/1 data base application software , consisting of approximately 1,317,000lines of code .During nineteen weeks of experiments, 47.65 CPU hours were consumed and about 328 software errors are removed. Fitting the model to the actual data means by estimating the model parameter from actual failure data. Here we used the LSE (non-linear least square estimation) and MLE to estimate the parameters. Calculations are given in appendix A.

All parameters of other distribution are estimated through LSE. The unknown parameters of Logistic-exponential TEF are α=72(CPU hours), λ=0.04847, and k=1.387. Correspondingly the estimated

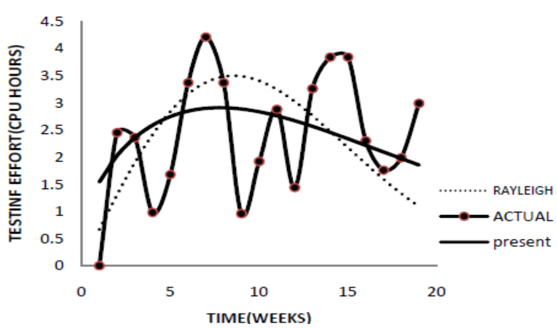

Fig 1. Observed/estimated logistic-exponential and Rayleigh TEF for DS1.

parameters of Rayleigh TEF N=49.32 and b=0.00684/week. Fig.1 plots the comparison between observed failure data and the data estimated by Logistic-exponential TEF and Rayleigh TEF. The PE, Bias ,Variation, MRE and RMS-PE for Logistic-exponential and Rayleigh are listed in Table I. From the TABLE I we can see that Logistic-exponential has lower PE, Bias, Variation, MRE and RMS-PE than Rayleigh TEF. We can say that our proposed model fits better than the other one. In the table II we have listed estimated values of SRGM with different testing-efforts. We have also given the values of SSE, R^2 and MSE. We observed that our proposed model has smallest MSE and SSE value when compared with other models. The 95% confidence limits for the all models are given in the Table III. All the calculations can found in the appendix. Fig .4 shows the RE curves for the different selected models.

TABLE-1
COMPARISION RESULT FOR DIFFERENT TEF APPLIED TO DS1

TEF	Bias	Variation	MRE	RMS-PE
Logistic-exponential	0.2245	1.297	0.033	1.27
Rayleigh	0.830337	2.169314	0.052676	2.004112

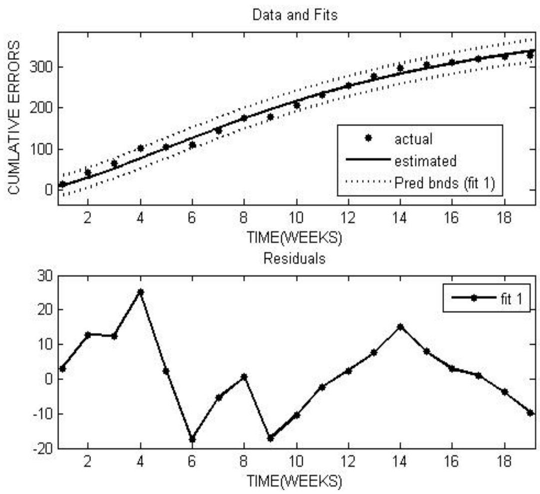

Fig 2. Cumulative and residual error for SRGM with Logistic-exponential for DS1

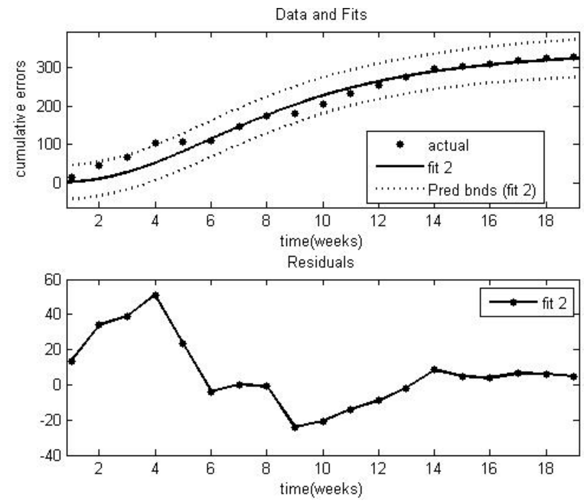

Fig 3. Cumulative and residual error for delayed S shaped model with logistic-exponential for DS1

Table II

ESTIMATED PARAMETER VALUES AND MODEL COMPARISION FOR DS1

Models	a	r	SSE	R^2	MSE
SRGM with Logistic-exponential TEF	578.8	0.01903	2183	0.9889	128.36
Delayed S shaped model with Logistic-exponential TEF	353.7	0.08863	7546	0.9615	443.94
SRGM with Rayleigh TEF	459.1	0.02734	5100	0.974	299.98
Delayed S shaped model with Rayleigh TEF	333.2	0.1004	15170	0.9226	892.2
G-O model	760.5	0.03227	2656	0.9865	156.2
Yamada Delayed S shaped model	374.1	0.1977	3205	0.9837	188.51

Table III

95% CONFIDENCE LIMIT FOR DIFFERENT SELECTED MODELS(DS1)

Models	a		r	
	Lower	Upper	Lower	Upper
SRGM with Logistic-exponential TEF	441.5	716	0.01268	0.02538
SRGM with Rayleigh TEF	348.6	569.6	0.01651	0.03817
Yamada Delayed S shaped Model with Logsitic-exponential TEF	314.5	392.8	0.07288	0.1044
Yamada Delayed S shaped Model with Rayleigh TEF	288.7	377.7	0.07507	0.1258
G-O model	465.4	1056	0.01646	0.04808
Yamada Delayed S shaped model	343.7	404.4	0.1748	0.2205

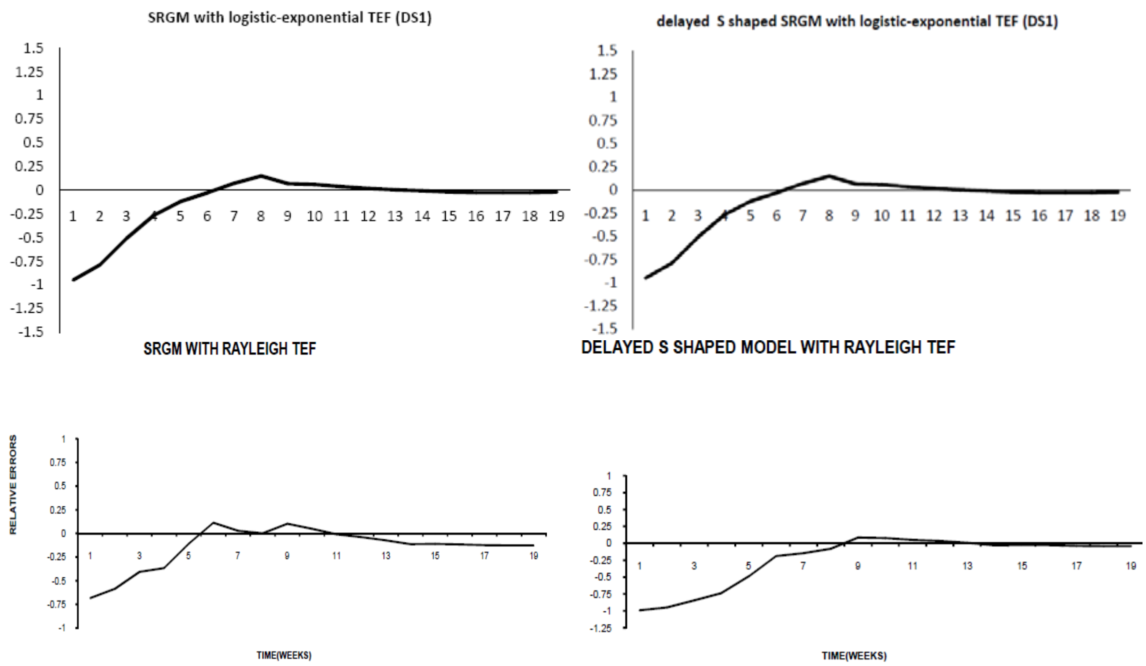

Fig.4 RE curves of selected models compared with actual failure data(DS1)

DS2: the dataset used here presented by wood [2] from a subset of products for four separate software releases at Tandem Computer Company. Wood Reported that the specific products & releases are not identified and the test data has been suitably transformed in order to avoid

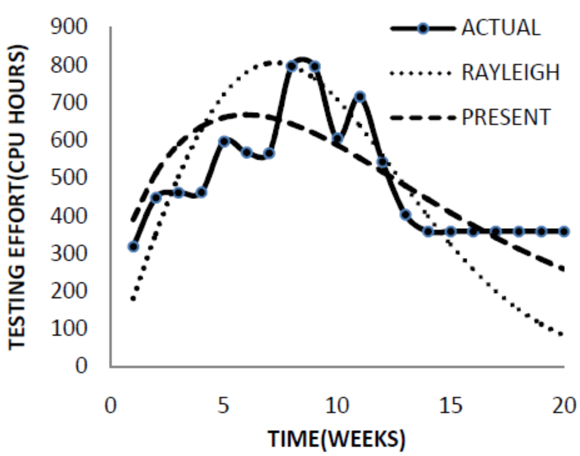

Fig 5. Observed/estimated Logistic-exponential and Rayleigh TEF for DS2.

Confidentiality issue. Here we use release 1 for illustrations. Over the course of 20 weeks, 10000 CPU hours are consumed and 100 software faults are removed. Similarly the least square estimates of the parameters for logistic-exponential TEF in the case of DS2 are α=12600(CPU hours), λ=0.06352, and k=1.391. Correspondingly the estimated parameters of Rayleigh TEF N=9669 and b=0.009472/week. The computed Bias, Variation, MRE, and RMS-PE for logistic-exponential TEF and Ralyleigh TEF are listed in the table IV ,fig 5 graphically illustrate the comparisons between the observed failure data, and the data estimated by the Logistic-exponential TEF and Rayleigh TEF. From the figure 5 we can observe the logistic-exponential curve covers the maximum points like other TEFs. Now from the table V we can conclude that our TEF is better fit than other. Their 95% confidence bounds are given in the table VI. From the above we can see that SRGM with logistic-exponential TEF have less MSE than other models.

Table IV
COMPARISION RESULT FOR DIFFERENT TEF APPLIED TO DS2

TEF	Bias	Variation	MRE	RMS-PE
Logistic-exponential	15.57	139.04	0.008	138.57
Rayleigh	121.61	322	0.055	298.23

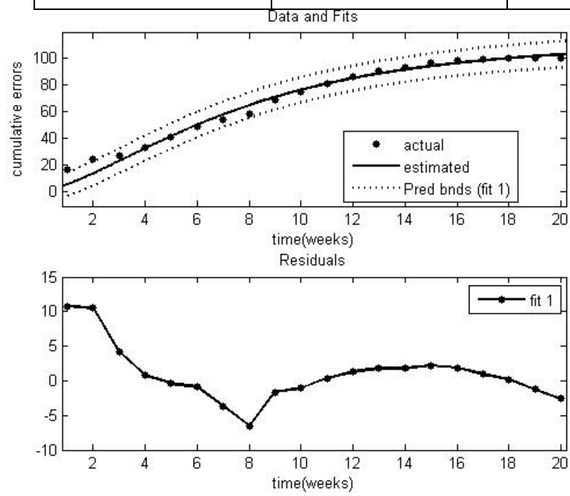

Fig:6 cumulative and residual error for SRGM with Logistic-exponential TEF(DS2)

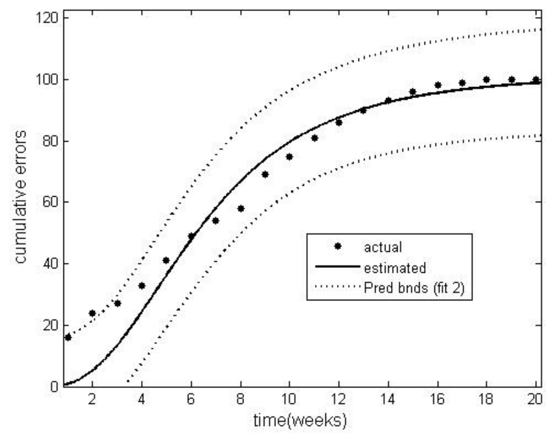

Fig 7. Cumulative and residual error for delayed S shaped model with logistic-exponential for DS2

Table V
ESTIMATED PARAMETER VALUES AND MODEL COMPARISION FOR DS2

Models	a	r	SSE	R^2	MSE
SRGM with Logistic-exponential TEF	135.6	0.0001423	331.4	0.9796	18.41
Delayed S shaped model with Logistic-exponential TEF	103.9	0.0004813	1044	0.9358	57.97
SRGM with Rayleigh TEF	120.9	0.0001791	792.5	0.9513	44.03
Delayed S shaped model with Rayleigh TEF	99.4	0.0005434	1930	0.8813	107.1

Table VI
95% CONFIDENCE LIMIT FOR DIFFERENT SELECTED MODELS(DS2)

Models	a		r	
	Lower	Upper	Lower	Upper
SRGM with Logistic-exponential TEF	115.1	156	0.0001423	0.0001801
SRGM with Rayleigh TEF	98.4	143	0.0001122	0.0002461
Yamada Delayed S shaped Model with Logistic-exponential TEF	94.02	113.7	0.0003908	0.0005718
Yamada Delayed S shaped Model with Rayleigh TEF	88.24	110.6	0.0003991	0.0006877

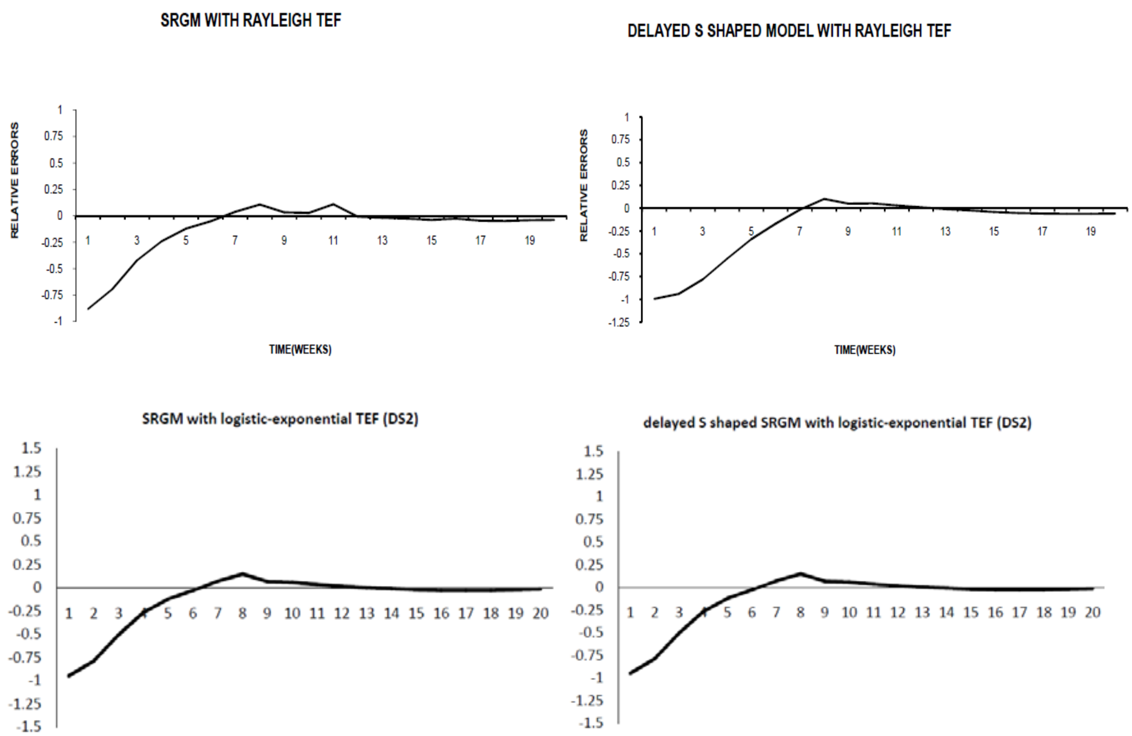

Fig.8 RE curves of selected models compared with actual failure data(DS2)

6) OPTIMAL SOFTWARE RELEASE POLICY

6.1 Software Release-Time Based on Reliability Criteria

Generally software release problem associated with the reliability of a software system. Here in this first we discuss the optimal time based on reliability criterion. If we know software has reached its maximum reliability for a particular time. By that we can decide right time for the software to be delivered out. Goel and Okumoto [1] first dealed with the software release problem considering the software cost-benefit. The conditional reliability function after the last failure occurs at time t is obtained by

$$R(t+\Delta t/t)=\exp(-[m(t+\Delta t/t)-m(t)])$$

$$=\exp\left(-m(\Delta t)\times\exp(-r\times W*(t))\right) \qquad (24)$$

Taking the logarithm on both sides of the above equation and rearrange the above equation we obtain

$$\ln R=-m(\Delta t)\times\exp(-r\times W*(t)) \qquad (25)$$

thus

$$W*(t)=\frac{1}{r}\left[\ln m(\Delta t)-\ln\ln\left(\frac{1}{R}\right)\right] \qquad (26)$$

By solving the eq 26 we can reach the desired reliability level. For DS1 Δt=0.1 R=0.91 at T=42.1weeks

6.2 Optimal release time based on cost-reliability criterion

This section deals with the release policy based on the cost-reliability criterion. Using the total software cost evaluated by cost criterion, the cost of testing-effort expenditures during software testing/development phase and the cost of fixing errors before and after release are: [9,13,25]

$$C(T)=C_1\,m(T)+C_2\left[m(T_{LC})-m(T)\right]+C_3\left(\int_0^T w(x)\,dx\right) \qquad (27)$$

Where C_1 the cost of correcting an error during testing, C_2 is the cost of correcting an error during the operation, $C_2 > C_1$, C_3 is the cost of testing per unit testing effort expenditure and T_{LC} is the software life-cycle length.

From reliability criteria, we can obtain the required testing time needed to reach the reliability objective R_0. Our aim is to determine the optimal software release time that minimizes the total software cost to achieve the desired software reliability. Therefore, the optimal software release policy for the proposed software reliability can be formulated as Minimize C(T) subjected to $R(t+\Delta t/t) \geq R_0$ for $C_2 > C_1$, $C_3 > 0$, $\Delta t > 0$, $0 < R_0 < 1$.

Differentiate the equation (30) with respect to T and setting it to zero, we obtain

$$\frac{d}{dT} C(T) = C_1 \left(\frac{d}{dT} m(T)\right) + C_2 \left[\frac{\partial}{\partial T} m(T_{LC}) - \left(\frac{d}{dT} m(T)\right)\right] + C_3\, w(T) \quad (28)$$

$$\frac{\partial}{\partial T} m(T_{LC}) = 0$$

$$\frac{d}{dT} C(T) = C_1 \left(\frac{d}{dT} m(T)\right) + C_2 \left[-\left(\frac{d}{dT} m(T)\right)\right] + C_3\, w(T) = 0 \quad (29)$$

$$\frac{d}{dT} m(T) = \lambda(T)$$

$$\frac{\lambda(T)}{w(T)} = \frac{c3}{c2 - c1}$$

$$a\, r\, e^{-r\,W(t)} = r\,(a - m(T)) \quad (30)$$

When T=0 then m(0)=0 and $\frac{\lambda(T)}{w(T)} = a\, r$

When T->∞, then $m(\infty) = a$

And $\frac{\lambda(T)}{w(T)} = a \times r \times e^{-r \times \alpha}$ therefore $\frac{\lambda(T)}{w(T)}$ is monotonically decreasing in T.

To analyze the minimum value of C(T) Eq. (27) is used to define the two cases of $\frac{\lambda(T)}{w(T)}$

at T=0. 1) if $\frac{\lambda(0)}{w(0)} = a \times r \leq \frac{C3}{C2 - C1}$, then $\frac{\lambda(T)}{w(T)} \leq \frac{C3}{C2 - C1}$ for $0<T<T_{LC}$ it can be obtained at dC(T)/dT>0 for $0<T<T_{LC}$ and the minimal value can found at C(T) can be found at T=0.

if $\frac{\lambda(0)}{w(0)} = a \times r > \frac{C3}{C2 - C1} > \frac{\lambda(T)}{w(T)} = a \times r \times e^{-r \times \alpha}$ there can be found a finite and unique real number

$$T_0 = \frac{\ln\left(\left(-\frac{\ln\left(-\frac{C3}{a\,r\,(-C2 + C1)}\right)}{r\,\alpha + \ln\left(-\frac{C3}{a\,r\,(-C2 + C1)}\right)}\right)^{\frac{1}{k}} + 1\right)}{\lambda} \quad (31)$$

because dC(T)/dT<0 for $0<T<T_0$ and dC(T)/dT>0 for $T> T_0$, the minimum of C(T) is at $T=T_0$ for $T_0 \leq T$

we can easily get the required testing time needed to reach the reliability objective R_0. here our goal is to minimize the total software cost under desired software reliability and then the optimal software release time is obtained. That is can minimize the C(T) subjected to $R(t+\Delta t/t) \geq R_0$ where $0< R_0 <1$ [9,25]

T^* =optimal software release time or total testing time $=\max\{T_0, T_1\}$.Where T_0 =finite and unique solution T satisfying Eq.(31) T_1 =finite and unique T satisfying $R(t+\Delta t/t)=R_0$

By combining the above analysis and combining the cost and reliability requirements we have the following theorem. Theorem 1: Assume C2

<C1<0, C3<0, Δt>0, and $0<R_0<1$. Let T*be the optimal software release time

a) if $\frac{\lambda(0)}{w(0)} > \frac{C3}{C2-C1}$ and $\frac{\lambda(T)}{w(T)} = a \times r \times e^{-r \times \alpha} \le \frac{C3}{C2-C1}$ then

$$T^* = \begin{cases} \max(T_0, T_1) \textbf{ for } R\left(\frac{\Delta t}{0}\right) < R_0 < 1 \\ T_0 \textbf{ for } 0 < R_0 < \left(\frac{\Delta t}{0}\right) \end{cases}$$

b)

$$\textbf{if } \frac{\lambda(0)}{w(0)} \ge \frac{C3}{C2-C1} \textbf{ then } T^* \ge \begin{cases} T_1 \quad \textbf{for } R\left(\frac{\Delta t}{0}\right) < R_0 < 1 \\ 0 \textbf{ for } 0 < R_0 < R\left(\frac{\Delta t}{0}\right) \end{cases}$$

c)

$$\textbf{if } \frac{\lambda(0)}{w(0)} \le \frac{C3}{C2-C1} \textbf{ then } T^* = \begin{cases} T_1 \quad \textbf{for } R\left(\frac{\Delta t}{0}\right) < R_0 < 1 \\ 0 \textbf{ for } 0 < R_0 < R\left(\frac{\Delta t}{0}\right) \end{cases}$$

From the dataset one estimated values of SRGM with Logistic-exponential TEF α=72(CPU hours), λ=0.04847 /week, k=1.387, a=578.8 and r=0.01903 when Δt=0.1 R_0 =0.85 and we let C_1=2, C2 =50, C_3 =150 and T_{LC} =100 the estimated time T_1=37.1 weeks and release time from eq 30 T_0 =39.5 weeks. Now optimal Release Time max(37.1,39.5) is T*=39.5 weeks. Fig 10 shows the change in software cost during the time span. Now total cost of the software at optimal time 8354.

From the dataset two estimated values of SRGM with Logistic-exponential TEF α=12600(CPU hours), λ=0.06352 /week, k=1.391, a=135.6 and r=0.0001432 when Δt=0.1 R_0 =0.85 and we let C_1=1, C2 =200, C_3 =2 and T_{LC} =100 the estimated time T_1=18.1 weeks and release time from Eq 31 T_0 =8.05 weeks. Now optimal Release Time max (8.05, 18.1) is T*=18.1 weeks. Fig 11 shows the change in software cost during the time span. Now total cost of the software at optimal time 20,100.

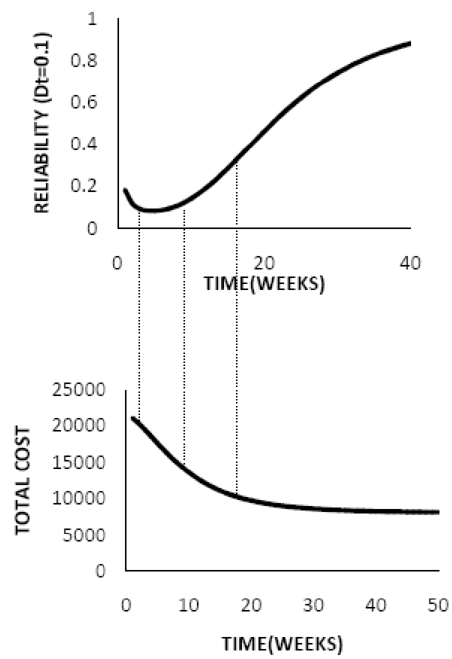

Fig 9 Reliability and Total Cost curve (DS1)

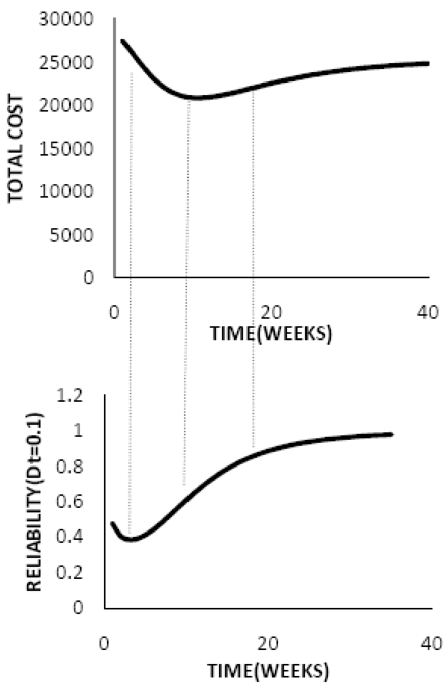

Fig 10 Reliability and Total Cost Curve (DS2)

7. Conclusions

In this paper, we proposed a SRGM incorporating the Logistic-exponential testing effort function that is completely different from the logistic type Curve. We Observed that most of software failure is time dependent. By incorporating testing-effort into SRGM we can make realistic assumptions about the software failure. The experimental results indicate that our proposed model fits fairly well.

Appendix –A

$$S = \sum_{k1=1}^{n} \left(\ln(W_{k1}) - \ln(\alpha) - k \times \ln \frac{\left(e^{\lambda \times t_{k1}} - 1\right)^k}{\left(1 + \left(e^{\lambda \times t_{k1}} - 1\right)^k\right)} \right)^2 \quad (32)$$

$$\frac{\partial S}{\partial \alpha} = 0$$

$$0 = \frac{2\,n \ln(\alpha)}{\alpha} + \sum_{k1=1}^{n} \left(-\frac{2\ln(W_{k1})}{\alpha} + \frac{2\,k \ln\left(e^{\lambda t_{k1}} - 1\right)^k}{\alpha\left(1 + \left(e^{\lambda t_{k1}} - 1\right)^k\right)} \right) \quad (33)$$

$$\alpha = e^{\frac{\sum_{k1=1}^{n} \ln(W_{k1}) - k \ln\left(\sum_{k1=1}^{n} \frac{\left(e^{\lambda t_{k1}} - 1\right)^k}{1 + \left(e^{\lambda t_{k1}} - 1\right)^k} \right)}{n}} \quad (34)$$

$$\frac{\partial S}{\partial \lambda} =$$

$$0 = 2\,k^2 \ln\Bigg(\sum_{k1=1}^{n} \frac{1}{\left(1 + \left(e^{\lambda t_{k1}} - 1\right)^k\right)^3} \Big(t_{k1}\, e^{\lambda t_{k1}} \Big(-\ln(W_{k1}) \left(e^{\lambda t_{k1}} - 1\right)^{k-1} - \ln(W_{k1}) \left(e^{\lambda t_{k1}} - 1\right)^{2k-1} + \left(e^{\lambda t_{k1}} - 1\right)^{k-1} \ln(\alpha) + \left(e^{\lambda t_{k1}} - 1\right)^{2k-1} \ln(\alpha) + k \left(e^{\lambda t_{k1}} - 1\right)^{2k-1} \ln \Big)\Big)\Bigg)$$

$$\frac{\partial S}{\partial k} =$$

$$2 \ln\Bigg(\sum_{k1=1}^{n} \frac{1}{\left(1 + \left(e^{\lambda t_{k1}} - 1\right)^k\right)^3} \Big(-\ln(W_{k1}) \left(e^{\lambda t_{k1}} - 1\right)^k - 2\ln(W_{k1}) \left(e^{\lambda t_{k1}} - 1\right)^{2k} - \ln(W_{k1}) \left(e^{\lambda t_{k1}} - 1\right)^{3k} - \ln(W_{k1})\, k \left(e^{\lambda t_{k1}} - 1\right)^k \ln\left(e^{\lambda t_{k1}} - 1\right) - \ln(W_{k1})\, k \left(e^{\lambda t_{k1}} - 1\right)^{2k} \ln\left(e^{\lambda t_{k1}} - 1\right) + \left(e^{\lambda t_{k1}} - 1\right)^k \ln(\alpha) + 2\left(e^{\lambda t_{k1}} - 1\right)^{2k} \ln(\alpha) + \left(e^{\lambda t_{k1}} - 1\right)^{3k} \ln(\alpha) + \ln(\alpha)\, k \left(e^{\lambda t_{k1}} - 1\right)^k \ln\left(e^{\lambda t_{k1}} - 1\right) + \ln(\alpha)\, k \left(e^{\lambda t_{k1}} - 1\right)^{2k} \ln\left(e^{\lambda t_{k1}} - 1\right) + k \ln\left(e^{\lambda t_{k1}} - 1\right)^{3k} + k \left(e^{\lambda t_{k1}} - 1\right)^{2k} \ln + k^2 \ln\left(e^{\lambda t_{k1}} - 1\right)^{2k} \ln\left(e^{\lambda t_{k1}} - 1\right) \Big)\Bigg)$$

S shaped model

$$\frac{dm_d(t)}{dt} \times \frac{1}{w(t)} = r1 \times \left[a - m_d(t)\right] \quad (35)$$

$$m_d(t) = a \times \left(1 - e^{-r \times W(t)}\right) \quad (36)$$

$$\frac{dm_r(t)}{dt} \times \frac{1}{w(t)} = r2 \times \left(m_d(t) - m_r(t)\right) \quad (37)$$

$$\frac{dm_r(t)}{dt} + r2 \times w(t) \times m_r(t) = r2 \times w(t) \times m_d(t) \quad (38)$$

$$I.F\ of\ above\ equation\ e^{\int_0^t r2 \times w(t)\, dt}$$

$$m_r(t) \times e^{r2 \times W(t)} = \int_0^t r2 \times w(t) \times m_d(t) \times e^{r2 \times W(t)}\, dt \quad (39)$$

solving above equation substituting $m_d(t)$

$$m_r(t) = a\left(1 - \left(\frac{\left(r1 \times e^{-r2 \times W(t)} - r2 \times e^{-r1 \times W(t)}\right)}{r1 - r2}\right)\right) \quad (40)$$

Above equation approaches to infinity so we apply the L' Hospitals Rule by letting

$$f(r2) = \left(r1 \times e^{-r2 \times W(t)} - r2 \times e^{-r1 \times W(t)}\right)$$

$$g(r2) = r1 - r2$$

$$\lim_{r2 \to r1} \frac{f(r2)}{g(r2)} = \lim_{r2 \to r1} \frac{(f(r2) - f(r1))}{(g(r2) - g(r1))}$$

$$f'(r1) = -r1 \times W(t) \times e^{[-r1 \times W(t)]} - e^{[-r1 \times W(t)]} \quad (41)$$

$$g'(r1) = -1$$

And $\frac{f'(r1)}{g'(r1)} = (1 + r1 \times W(t)) \times e^{[-r1 \times W(t)]}$ (42)

Appendix -B

Using the estimated parameters α, λ and k above, we estimate the reliability growth parameters a and r in Eq(8). Suppose that the data on the cumulative number of detected errors y_k in a given time interval $(0, t_k]$ (k = 1, 2,..., n) are observed. Then, the joint probability mass function, i.e. the likelihood function for the observed data, is given by

$$L \equiv \{Pr(N(t_1) = y_1, N(t_2) = y_2 \ldots\ldots N(t_n)) = y_n\} \quad (43)$$

$$\prod_{k=1}^{n} \frac{\left[m(t_k) - m(t_{k-1})\right]^{y_k - y_{k-1}}}{(y_k - y_{k-1})} \times \exp(-m(t_n)) \quad (44)$$

From Eq :13 $\frac{\partial}{\partial a}\ln(L) = 0$

$$0 = \sum_{k=1}^{n} \frac{y_k - y_{k-1}}{a} - 1 + (1 + r\, t_n)\, e^{-r W(t_n)} \quad (45)$$

$$a = \frac{y_n}{1 - (1 + r\, W(t_n))\, e^{-r W(t_n)}} \quad (46)$$

$$\frac{\partial}{\partial r}\ln(L) = 0$$

$$a \times (W(t_n))^2 \times \exp(-r \times W(t_n)) = \sum_{k=1}^{n} (y_k - y_{k-1}) \times \Big\{ \Big((W(t_k))^2 \times \exp(-r \times W(t_k)) - (W(t_{k-1}))^2 \times \exp(-r \times W(t_{k-1})) \Big) \Big/ \Big((1 + r \times W(t_{k-1})) \times \exp(-r \times W(t_{k-1})) - (1 + r \times W(t_k)) \times \exp(-r \times W(t_k)) \Big) \Big\} \quad \textbf{(47)}$$

Acknowledgement: I would like to thank to all the authors mentioned in the reference

REFERENCES

[1] A.L. Goel and K. Okumoto, A time dependent error detection rate model for a large scale software system, *Proc. 3rd USA-Japan Computer Conference*, pp. 3540, San Francisco, CA (1978).
[2] A.Wood, Predicting software reliability, IEEE computers 11 (1996) 69–77.
[3] Bokhari, M.U. and Ahmad, N. (2005), "Software reliability growth modeling for exponentiated Weibull functions with actual software failures data", in Proceedings of 3rd International Conference on Innovative Applications of Information Technology for Developing World (AACC'2005), Nepal.
[4] Bokhari, M.U. and Ahmad, N. (2006), "Analysis of a software reliability growth models: the case of log-logistic test-effort function", in Proceedings of the 17th International Conference on Modelling and Simulation (MS'2006), Montreal, Canada, pp. 540-5.
[5] C.-Y. Huang, S.-Y. Kuo, J.Y. Chen, Analysis of a software reliability growth model with logistic testing effort function proceeding of Eighth International Symposium on Software Reliability Engineering, 1997, pp. 378–388.
[6] Goel, A.L., "Software reliability models: Assumptions, limitations, and applicability", *IEEE Transactions on Software Engineering* SE-11 (1985) 1411-1423.
[7] Huang, C.Y. and Kuo, S.Y. (2002), "Analysis of incorporating logistic testing-effort function into software reliability modeling", IEEE Transactions on Reliability, Vol. 51 No. 3, pp. 261-70.
[8] Huang, C.Y., Kuo, S.Y. and Chen, I.Y. (1997), "Analysis of software reliability growth model with logistic testing-effort function", in Proceeding of 8th International Symposium on Software Reliability Engineering (ISSRE'1997), Albuquerque, New Mexico, pp. 378-88.
[9] Huang, C.Y., Kuo, S.Y. and Lyu, M.R. (1999), "Optimal software release policy based on cost, reliability and testing efficiency", in Proceedings of the 23rd IEEE Annual International
[10] Huang, C.Y., Kuo, S.Y. and Lyu, M.R. (2000), "Effort-index based software reliability growth models and performance assessment", in Proceedings of the 24th IEEE Annual International Computer Software and Applications Conference (COMPSAC'2000), pp. 454-9.

[11] Huang, Lyu and Kuo "An Assesment of testing effort dependent software reliability Growth model". IEEE transactions on Reliability Vol 56, No: 2, June 2007

[12] Huang and S. Y. Kuo, "Analysis and assessment of incorporating logistic testing effort function into software reliability modeling," *IEEE Trans. Reliability*, vol. 51, no. 3, pp. 261–270, Sept. 2002.

[13] K. Pillai and V. S. Sukumaran Nair, "A model for software development effort and cost estimation," *IEEE Trans. Software Engineering*, vol. 23, no. 8, August 1997.

[14] K. Srinivasan and D. Fisher, "Machine learning approaches to estimating software development effort," *IEEE Trans. Software Engineering*, vol. 21, no. 2, pp. 126–136, 1995.

[15] M. Ohba, Software reliability analysis models, IBM J. Res. Dev. 28 (1984) 428–443.

[16] M.R. Lyu, Handbook of Software Reliability Engineering, Mcgraw Hill, 1996.

[17] Pham, H. (2000), Software Reliability, Springer-Verlag, New York, NY.

[18] Quadri, S.M.K., Ahmad, N., Peer, M.A. and Kumar, M. (2006), "Nonhomogeneous Poisson process software reliability growth model with generalized exponential testing effort function", RAU Journal of Research, Vol. 16 Nos 1-2, pp. 159-63.

[19] Rameshwar D. Gupta and Debasis Kundu "generalized exponential distribution: different method of estimations" j. statist. comput. simul., 2000, vol. 00, pp. 1 – 22 14 november 2000.

[20] S. Yamada, H. Ohtera and R. Narihisa, "Software Reliability Growth Models with Testing-Effort," IEEE Trans. Reliability, Vol. R-35, pp. 19-23 (1986).

[21] S. Yamada, H. Ohtera, Software reliability growth model for testing effort control, Eur. J. Oper. Res. 46 (1990) 343–349.

[22] S.Yamada, S.Osaki, "Software reliability growth modeling: models and applications", *IEEE Trans. Software Engineering,* vol.l I, no.12, p.1431-1437, December 1985.

[23] Xie, M. (1991), Software Reliability Modeling, World Scientific Publication, Singapore.

[24] Yamada, S., Ohba, M., Osaki, S., 1983. S-shaped reliability growth modeling for software error detection. IEEE Trans. Reliab. 12, 475–484.

[25] Yamada, S. and Osaki, S. (1985b), "Cost-reliability optimal release policies for software systems", IEEE Transactions on Reliability, Vol. R-34 No. 5, pp. 422-4.

[26] J.D. Musa, A. Iannino, and K. Okumoto, Software Reliability:Measurement, Prediction, Application, McGraw-Hill NewYork, 1987.

[27] Y. Lan, and L. Leemis, (Aug. 2007) "The Logistic-Exponential Survival Distribution," *Naval Research Logistics (NRL)* volume 55, number 3, pp. 252-264.

Sk.MD.Rafi received B.Tech (comp) from Jawaharlal Nehru Technological University, M.Tech (comp) from Acharya Nagarjuna University. Pursuing PhD from Jawaharlal Nehru Technological University. Presently working as Associate. Professor in Sri Mittapalli Institute of Technology for women, affiliated to J.N.T.U, Kakinada. My area of interest is Software Reliability, Software Architecture Recovery, Network Security, and Software Engineering.

Dr.K.Nageswara Rao received B.Tech(electronics) from Karnataka University, M.Tech(comp) from Andhra University and PhD from Andhra University. Presently working as Professor& H.O.D in P.V.P.S.I.T, Vijayawada affiliated to J.N.T.U, Kakinada. My area of interest is Robotics, Software Engineering, Algorithms, and Software Reliability.

Shaheda Akthar received Bachelor of computer science from Acharya Nagarjuna University, M.S (Software Systems) from BITS, Pilani, pursuing PhD from Acharya Nagarjuna University. Presently working as Asscociate .Professor in Sri Mittapalli College of engineering, affiliated to J.N.T.U, Kakinada. My area of interest is Software Reliability, Software Architecture Recovery, Network Security, and Software Engineering