Software Reliability Growth Model with Bass Diffusion Test-Effort Function and Analysis of Software Release Policy

Shaik. Mohammad Rafi and Shaheda Akthar

Abstract—**Software reliability is one of the important factors of software quality. Many mathematical models are proposed in literature to predict the software quality and related reliability. Generally during testing many factors are considered like effort, time and resources. Testing effort can be better described by time, person hours and number of test cases. During testing many resources are being consumed. In this paper an analysis is done based on incorporating the Bass diffusion testing-effort function 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.**

Index Terms—**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 time t.
W (t): Cumulative testing effort consumption at time t.
W*(t): W (t)-W (0)
W (t): Cumulative testing effort consumption at time t.
W*(t) : W (t)-W (0)
A : Expected number of initial faults
W (t): Cumulative testing effort consumption at time
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 Bass diffusion TEF
r_2: Constant fault isolated rate in the Delayed S-shaped model with Bass diffusion TEF

Manuscript received March 3, 2011; revised September, 22, 2011.

Shaik.Mohammad Rafi is Assoc.Professor, Dept. of C.S.E Sri Mittapalli Institute of Technology for Women Affiliated to J.N.T.U , Kakinada, A.P, INDIA (Email: rafi527@gmail.com)

Shaheda Akthar is Assoc.Professor, Dept. of C.S.E Sri Mittapalli College of Engineering Affiliated to J.N.T.U , Kakinada, A.P, INDIA (Email: shaheda.akthar@yahoo.com)

I. INTRODUCTION

Software becomes crucial in daily life. Computers, communication 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 SRGM with NHPP models [4, 5, 8, 11, 120, 12, 20, 21]. All of them describe the tracking phenomenon with test expenditure. It is found that effort consumption rate is varied from time to time and no effort function fully describes effort consumed during the software development. For that we proposed a new effort function which is a combination of several effort-functions, better describes the effort expenditure.

This paper we used Bass diffusion testing-effort curve and incorporated in the SRGM. Result shows that the SRGM with Bass diffusion testing-effort function gives better results.

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.

II. 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)

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

The following are some of them

1) Exponential Testing effort function

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

$$W(t) = N \times (1 - e^{-b \times t}) \qquad t>0 \quad (2)$$

2) Rayleigh Testing effort curve:

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

$$W(t) = N \times (1 - e^{-b \times t^2}) \quad t>0 \quad (3)$$

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

3) Weibull Curve: the cumulative testing consumed is

$$W(t) = N \times (1 - e^{-b \times t^m}) \quad t>0 \quad (4).$$

A Weibull type curve can well fit the data often used in the field of software reliability modeling, it displays a peak phenomenon when the shape parameter is m>3.

4) Logistic Curve: the logistic testing-effort function was originally proposed by F.N Parr. It behaves in similarly as Rayleigh Curve, expect during early part of the project. The cumulative testing effort is given by

$$W(t) = \frac{N}{(1 + A \times e^{-\alpha \times t})} \quad t>0 \quad (5)$$

5) Bass diffusion TEF: Bass model for was introduced in the marketing research community to model production diffusion. [31]. Here the parameter p is a measure of intrinsic complexity of the code. If q=0 there is no debugging and each bug reveals if self independently with mean 1/p. The parameter q is a measure of learning by a particular debug team. Initially the testing team starts with no knowledge about the software under testing as the testing is progressed they learn about the software. So our bass TEF model could be used to describe the testing effort expenditure during testing, This function was interpreted in different way by P.K Kapur [32] where he describes faults are interdependent, while removing one leading fault can remove all dependent faults.

$$W(t) = \alpha \times \frac{(1 - e^{(-(p+q)\times t)})}{(1 + \frac{q}{p} \times e^{(-(p+q)\times t)})} \quad , \quad t>0 \quad (6)$$

And its current testing effort is

$$w(t) = \alpha \times \frac{[(p+q)^2 \times e^{-(p+q)\times t} \times q]}{(q + p \times e^{-(p+q)\times t})^2} \quad t>0 \quad (7)$$

The testing effort function reaches its maximum value at $t_{\max} = -\frac{\ln(q/p)}{(p+q)}$ α is the total expenditure, p and q positive complexity and learning parameters. There are two special cases of the Bass diffusion model.

- The first special case occurs when q=0, when the model reduces to the Exponential distribution.
- The second special case reduces to the logistic distribution, when p=0.

The Bass model is a special case of the Gamma/shifted Gompertz distribution (G/SG).

III. Software Reliability Growth Models

A. Software reliability growth model with Bass diffusion TEF

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

1) The fault removal process follows the Non-Homogeneous Poisson process (NHPP)
2) The software system is subjected to failure at random time caused by faults remaining in the system.
3) 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.
4) The proportionality is constant over the time.
5) Consumption curve of testing effort is modeled by a Bass diffusion TEF.
6) Each time a failure occurs, the fault that caused it is immediately removed and no new faults are introduced.
7) We can describe the mathematical expression of a testing-effort based on following

$$\frac{dm(t)}{dt} \times \frac{1}{w(t)} = r \times (a - m(t)) \quad (8)$$

$$m(t) = a \times \left(1 - e^{-r \times (W(t) - W(0))}\right) \quad (9)$$

Substituting W(t) into Eq.(9), we get

$$m(t) = a \times (1 - e^{-r \times \alpha \times \frac{(1 - e^{(-(p+q)\times t)})}{(1 + \frac{q}{p} \times e^{(-(p+q)\times t)})}}) \quad (10)$$

This is an NHPP model with mean value function with the Bass diffusion testing-effort expenditure.

Now failure intensity is given by

$$\lambda(t) = \frac{dm(t)}{dt} = a \times r \times w(t) \times e^{-r \times W(t)} \quad (11)$$

$$\lambda(t) = a \times r \times \alpha \times \frac{[(p+q)^2 \times e^{-(p+q)\times t} \times q]}{(q + p \times e^{-(p+q)\times t})^2} \times e^{-r \times \alpha \times \frac{(1 - e^{(-(p+q)\times t)})}{(1 + \frac{q}{p} \times e^{(-(p+q)\times t)})}} \quad (12)$$

The expected number of errors detected eventually is

$$m(\infty) = a\left(1 - e^{-r\alpha}\right) \quad (13)$$

B. Yamada Delayed S-shaped model with Bass diffusion 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 error 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, and 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

$$\frac{dm_d(t)}{dt} \times \frac{1}{w(t)} = r_1 \times \left[a - m_d(t)\right] \tag{14}$$

And

$$\frac{dm_r(t)}{dt} \times \frac{1}{w(t)} = r_2 \times \left[a - m_r(t)\right] \tag{15}$$

We assume r2≠r1 by solving 2 and 3 boundary conditions md(t)=0 , we have

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

$$m_r(t) = a \times \left[1 - \frac{\left(r_1 \times e^{\left[-r_2 \times W*(t)\right]} - r_2 \times e^{\left[-r_1 \times W*(t)\right]}\right)}{r1 - r2}\right] \tag{16}$$

At this stage we assume r2≈ r1≈r , then using 'L' Hospitals rule the Delayed S-shaped model with TEF is given by

$$m(t) \cong m_r(t) = a \times \left(1 - (1 + r \times W*(t)) \times e^{\left[-r \times W*(t)\right]}\right) \tag{17}$$

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

$$\lambda(t) = a \times r^2 \times w(t) \times W*(t) \times e^{\left[-r \times W*(t)\right]} \tag{18}$$

IV. Evaluation Criteria

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

$$MSE = \sum_{i=1}^{k} \frac{\left[m\left(t_i\right) - m_i\right]^2}{k} \tag{19}$$

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

b) Coefficient of multiple determinations (R2) 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 provides the best fit to the data. The best model is that which proves higher R2. That is closer to 1.

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

$$RE = \frac{\left(m(t_q) - q\right)}{q} \tag{20}$$

SSE criteria: SSE can be calculated as :[17]

$$SSE = \sum_{i=1}^{n} \left[y_i - m(t_i)\right]^2 \tag{21}$$

where yi 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.

$$PE_i = Actual(observed)_i - Predicted(estimated)_i \tag{22}$$

$$Bias = \sum_{i=1}^{n} \frac{PE_i}{n} \tag{23}$$

$$Variation = \sqrt{\sum_{i=1}^{n} \frac{(PE_i - Bias)^2}{n-1}} \tag{24}$$

$$MRE = \frac{\left|M_{estimated} - M_{actual}\right|}{M_{actual}} \tag{25}$$

V. Model Performance Analysis

1) *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 MLE to estimate the parameters. Calculations are given in appendix A.

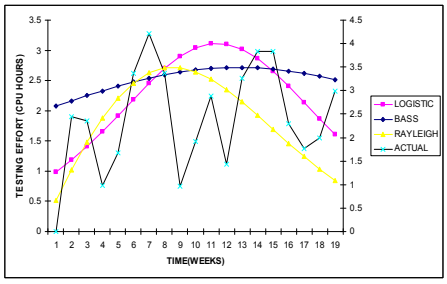


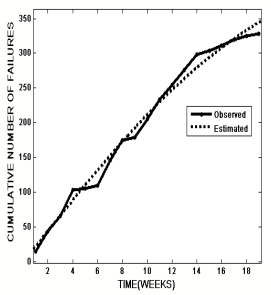
Fig. 1.Observed/estimated Bass diffusion, Logistic and Rayleigh TEF for DS1.

All parameters of other distribution are estimated through non linear least square estimation. The unknown parameters of Bass diffusion TEF are α=91.58(CPU hours), p=0.02164 and q=0.06848 correspondingly the estimated parameters of Logistic TEF N=54.84, A=13.03 and β=0.2263 and Rayleigh TEF N=49.32 and b=0.00684/week. Fig.1 plots the comparison between observed failure data and the data estimated by Bass diffusion TEF, Logistic TEF and Rayleigh TEF. The PE, Bias, Variation, MRE and RMS-PE for EW, Logistic and Rayleigh are listed in Table I. From the TABLE I we can see that Bass TEF has lower PE, Bias, Variation, MRE and RMS-PE than Logistic and Rayleigh TEF. We can say that our proposed model fits better than the other one. In the table II we havelisted 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 I: COMPARISION RESULT FOR DIFFERENT TEF APPLIED TO DS1

TEF	Bias	Variation	MRE
Bass Diffusion	0.035	0.9348	0.0025
Logistic	-0.09826	1.306	0.02
Rayleigh	*0.830337*	*2.169314*	*0.052676*



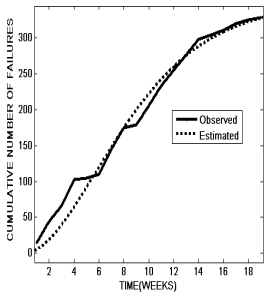


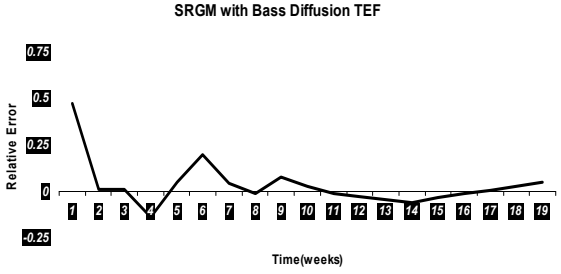
Fig. 2.SRGM with Bass diffusion TEFl for DS1 Fig 3. Delayed S shaped with Bass TEF for DS1

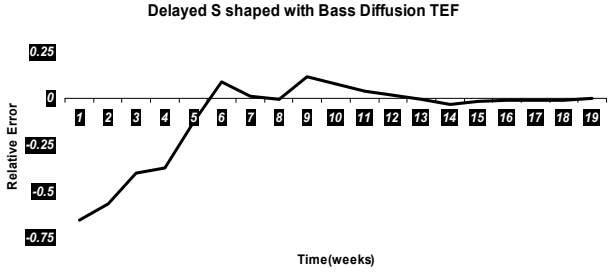
TABLE II: ESTIMATED PARAMETER VALUES AND MODEL COMPARISION FOR DS1

Models	a	r	SSE	R^2	MSE
SRGM with Bass diffusion TEF	564.1	0.01973	2034	0.9896	119.68
Delayed S shaped model with Bass diffusion TEF	353.4	0.08963	4138	0.9789	243.3
SRGM with Logistic TEF	395.6	0.0416	2167	0.989	127.46
Delayed S shaped model with Logistic TEF	319.3	0.1339	11060	0.9436	650.25
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 SELECTEDMODELS (DS1)

Models	a		r	
	Lower	Upper	Lower	Upper
SRGM with Bass diffusion TEF	449.2	679	0.014	0.02546
SRGM with Rayleigh TEF	348.6	569.6	0.01651	0.03817
SRGM with Logistic TEF	358	433.2	0.03399	0.04928
Yamada Delayed S shaped Model with Bass diffusion TEF	326.1	380.6	0.07862	0.1006
Yamada Delayed S shaped Model with Logistic TEF	291	347.5	0.1088	0.1589
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





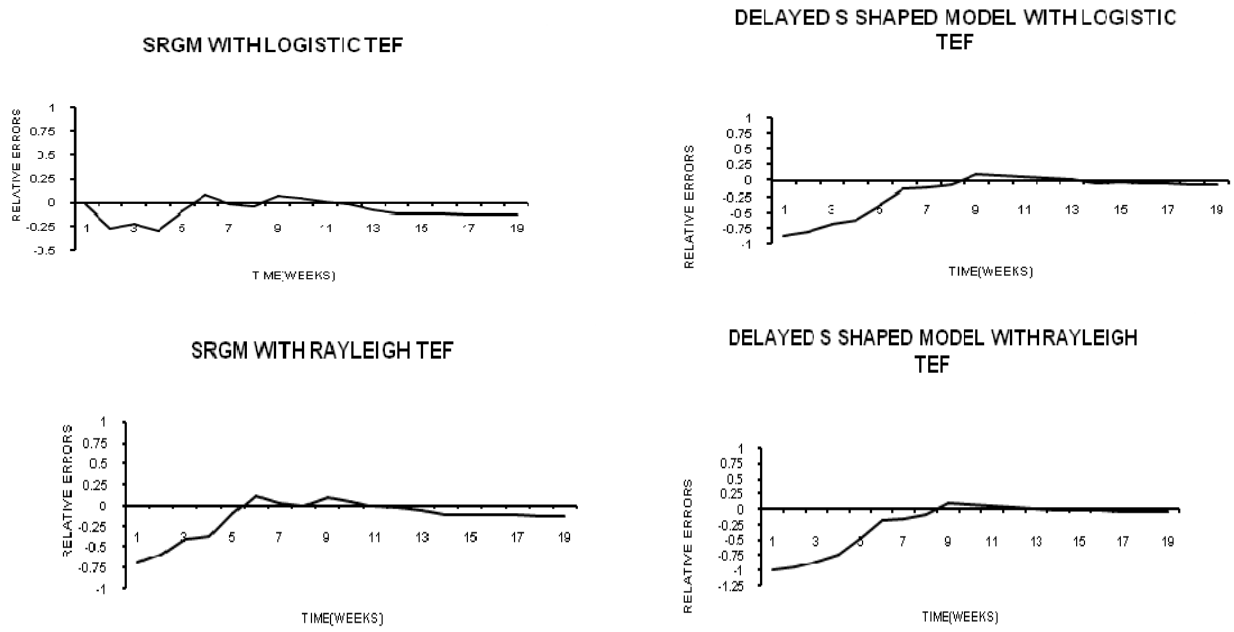


Fig. 4. RE curves of selected models compared with actual failure data(DS1)

2) *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 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. Estimates of the parameters for Bass diffusion TEF in the case of DS2 are α=10850(CPU hours), p=0.03567, and q=0.1624 correspondingly the estimated parameters of Logistic TEF N=9974, A==13.22 and β=0.2881 and Rayleigh TEF N=9669 and b=0.009472/week. The computed Bias, Variation, MRE, and RMS-PE for Bass diffusion TEF, Logistic TEF and Rayleigh TEF are listed in the table IV ,fig 5 graphically illustrate the comparisons between the observed failure data, and the data estimated by the Bass diffusion, Logistic TEF and Rayleigh TEF. From the figure 5 we can observe the Bass diffusion 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 Bass diffusion TEF have less MSE than other models.

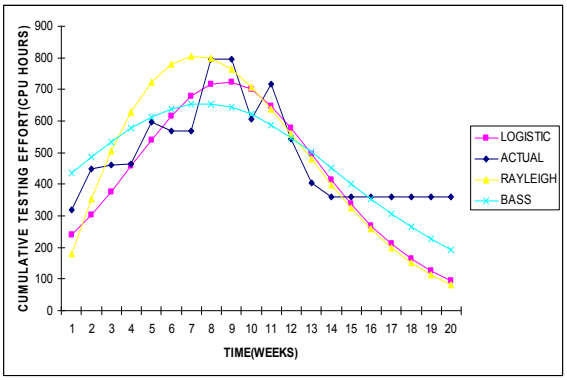


Fig . 5.Observed/estimated Bass diffusion TEF ,Logistic and Rayleigh TEF for DS2.

TABLE IV: COMPARISION RESULT FOR DIFFERENT TEF APPLIED TO DS2

TEF	Bias	Variation	MRE
Bass diffusion	6.27	117.1	0.02
Logistic	-17.99	198.76	0.042
Rayleigh	122.94	321.7	0.05

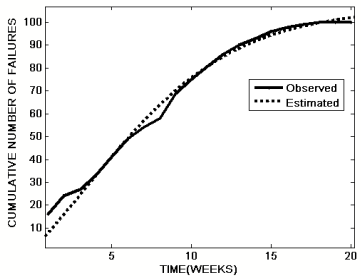


Fig. 6.cumulative errors for SRGM with Bass diffusion TEF(DS2)

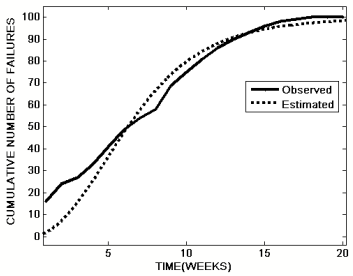


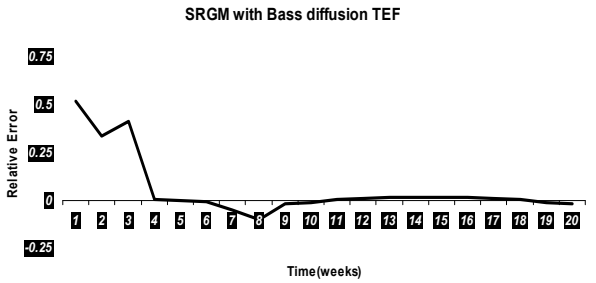
Fig. 7.Cumulative errors for delayed S shaped model with Bass diffusion TEF for DS2

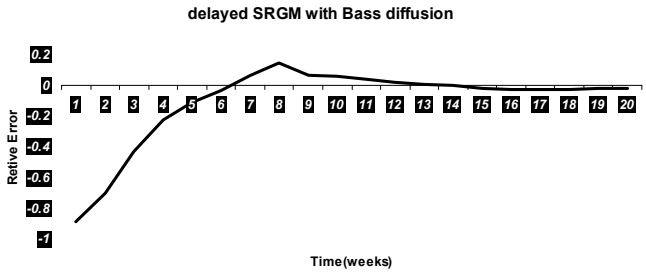
TABLE V: ESTIMATED PARAMETER VALUES AND MODEL COMPARISION FOR DS2

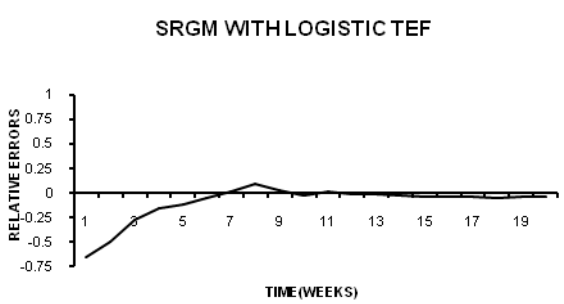
Models	a	r	SSE	R^2	MSE
SRGM with Bass diffusion TEF	135.3	0.0001431	200.5	0.9877	11.14
Delayed S shaped model with Bass diffusion TEF	103.1	0.000492	871	0.9464	48.38
SRGM with Logistic TEF	112.3	0.0002399	433.1	0.9734	24.059
Delayed S shaped model with Logistic TEF	96.88	0.0006853	1577	0.903	87.61
SRGM 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

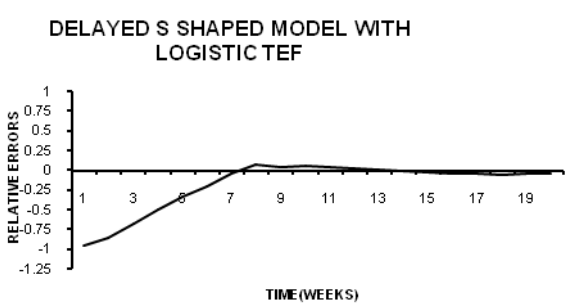
TABLE VI: 95% CONFIDENCE LIMIT FOR DIFFERENT SELECTED MODELS(DS2)

Models	a		R	
	Lower	Upper	Lower	Upper
SRGM with Bass diffusion TEF	119.7	150.9	0.000114	0.0001722
SRGM with Logistic TEF	101.4	123.1	0.000186	0.0002938
SRGM with Rayleigh TEF	98.4	143	0.0001122	0.0002461
Yamada Delayed S shaped Model with Bass diffusion TEF	91.5	111.7	0.0004091	0.0005749
Yamada Delayed S shaped Model with Logistic TEF	88.64	105.1	0.0005346	0.0008359
Yamada Delayed S shaped Model with Rayleigh TEF	88.24	110.6	0.0003991	0.0006877









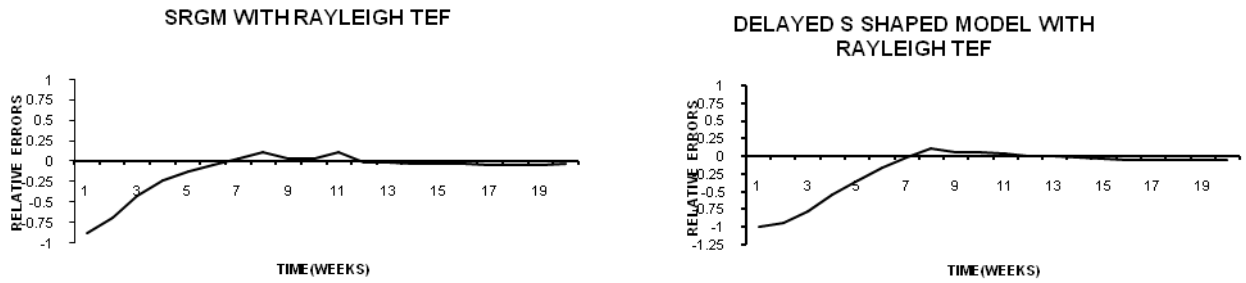


Fig. 8. RE curves of selected models compared with actual failure data (DS2)

3) DS-3: the system T1 data of the Rome Air Development center (RADC) projets and cited from Musa et.al (1987). The number if object instructions for the system T1 which is used for a real time command and control. In this case size of the software is approximately 21,700 object instructions. The software tested for the 21 week. During the testing phase about 25.3 CPU hours were used and 136 faults were discovered.

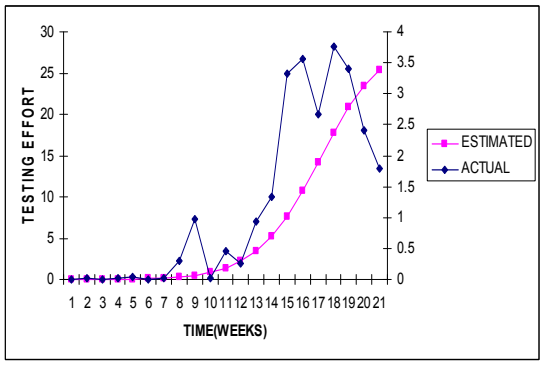


Fig. 9.Observed/Estimated Current Testing effort function Vs Time

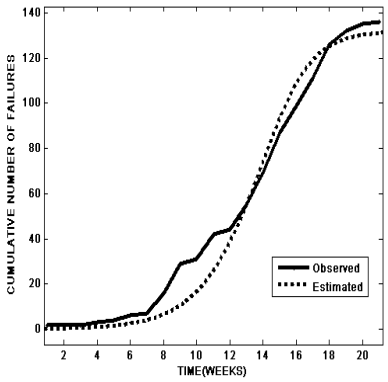


Fig. 10.Cumulative Number of Failures for SRGM with Bass Diffusion TEF

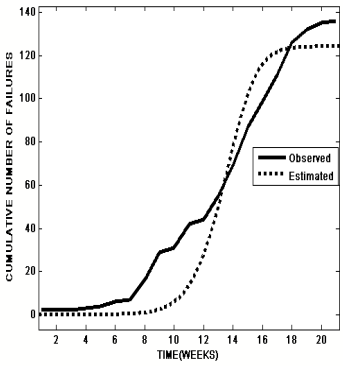


Fig. 11.Cumulative Number of Failures for Delayed S SRGM with Bass Diffusion TEF

Estimates of the parameters for Bass diffusion TEF in the case of DS2 are α=29.1(CPU hours), p=0.0001072, and q=0.4931

TABLE V.

MODEL	a	r	MSE
SRGM with Bass diffusion TEF	133.4	0.1575	66.52
G-O model (ohba)	142.32	0.1246	2438.3
SRGM with Rayleigh TEF	866.9	0.00962	89.239
Delayed S-Shaped model	237.196	0.0963	245.24

VI. Optimal Software Release Policy

A. 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+Δt/t)=exp(-[m(t+ Δt/t)-m(t)])

$$= \exp\left(-m(\Delta t) \times \exp(-r \times W*(t))\right) \quad (26)$$

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

$$\ln R = -m(\Delta t) \times \exp(-r \times W*(t)) \quad (27)$$

Thus

$$W*(t) = \frac{1}{r}\left[\ln m(\Delta t) - \ln \ln\left(\frac{1}{R}\right)\right] \quad (28)$$

By solving the Eq.26 we can reach the desired reliability level. For DS1 Δt=0.1 R=0.91 at T=25.1weeks

B. 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, and 25]

$$C(T) = C_1\, m(T) + C_2\left[m\left(T_{LC}\right) - m(T)\right] + C_3\left(\int_0^T w(x)\,\mathrm{d}x\right) \tag{29}$$

Where C1 the cost of correcting an error during testing, C2 is the cost of correcting an error during the operation, C2 > C1, C3 is the cost of testing per unit testing effort expenditure and TLC is the software life-cycle length.

From reliability criteria, we can obtain the required testing time needed to reach the reliability objective R0. 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+Δt/t)≥ R0 for C2 > C1, C3 >0, Δt>0, 0 < R0 <1.

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

$$\frac{\mathrm{d}}{\mathrm{d}T}C(T) = C_1\left(\frac{\mathrm{d}}{\mathrm{d}T}m(T)\right) + C_2\left[\frac{\partial}{\partial T}m\left(T_{LC}\right) - \left(\frac{\mathrm{d}}{\mathrm{d}T}m(T)\right)\right] + C_3\,w(T) \tag{30}$$

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

$$\frac{\mathrm{d}}{\mathrm{d}T}C(T) = C_1\left(\frac{\mathrm{d}}{\mathrm{d}T}m(T)\right) + C_2\left[-\left(\frac{\mathrm{d}}{\mathrm{d}T}m(T)\right)\right] + C_3\,w(T) = 0 \tag{31}$$

$$\frac{\mathrm{d}}{\mathrm{d}T}m(T) = \lambda(T),\ \frac{\lambda(T)}{w(T)} = \frac{c3}{c2 - c1}$$

$$a\,r\,\mathrm{e}^{-r\,W(t)} = r\,(a - m(T)) \tag{32}$$

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

When T->∞, then $m(\infty) = a\left(1 - \mathrm{e}^{-r\,\alpha}\right)$

And $\frac{\lambda(T)}{w(T)} = a \times r \times \mathrm{e}^{-r\times\alpha}$ 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 $\frac{\lambda(0)}{w(0)} = a \times r \le \frac{C3}{C2 - C1}$, then $\frac{\lambda(T)}{w(T)} \le \frac{C3}{C2 - C1}$ for 0<T<TLC it can be obtained at dC(T)/dT>0 for 0<T<TLC and the minimal value can found at C(T) can be found at T=0.

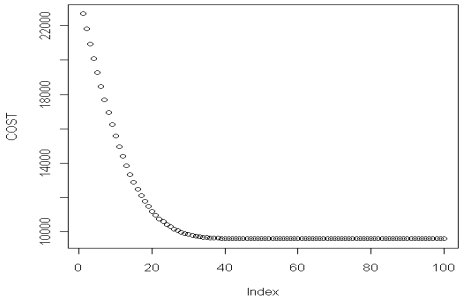
if $\frac{\lambda(0)}{w(0)} = a \times r > \frac{C3}{C2 - C1} > \frac{\lambda(T)}{w(T)} = a \times r \times \mathrm{e}^{-r\times\alpha}$ there can be found a finite and unique real number

$$T_0 = -\frac{\ln\left(-\dfrac{q\left(r\,\alpha + \ln\left(-\dfrac{C3}{a\,r\,(-C2 + C1)}\right)\right)}{-r\,\alpha\,q + \ln\left(-\dfrac{C3}{a\,r\,(-C2 + C1)}\right)p}\right)}{p + q} \tag{33}$$

because dC(T)/dT<0 for 0<T<T0 and dC(T)/dT>0 for T> T0 , the minimum of C(T) is at T=T0 for T0 ≤ T. we can easily get the required testing time needed to reach the reliability objective R0 . 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+Δt/t)≥ R0 where 0< R0 <1 [9,25]

T* =optimal software release time or total testing time =max {T0, T1}.Where T0 =finite and unique solution T satisfying Eq.(31) T1 =finite and unique T satisfying R(t+Δt/t)=R0

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<R0 <1. Let T*be the optimal software release time



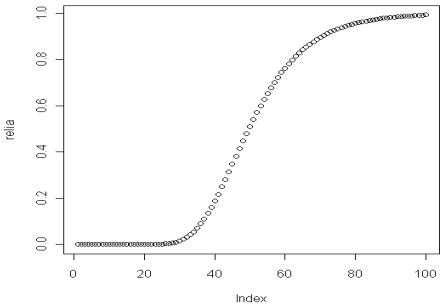


Fig. 12. Reliability and Total Cost curve (DS1)

a) if $\frac{\lambda(0)}{w(0)} > \frac{C3}{C2 - C1}$ and

$\frac{\lambda(T)}{w(T)} = a \times r \times \mathrm{e}^{-r\times\alpha} \le \frac{C3}{C2 - C1}$ 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) $\text{if } \frac{\lambda(0)}{w(0)} \geq \frac{C3}{C2-C1} \text{ then } T^* \geq \begin{cases} T_1 & \text{for } R\left(\frac{\Delta t}{0}\right) < R_0 < 1 \\ 0 & \text{for } 0 < R_0 < R\left(\frac{\Delta t}{0}\right) \end{cases}$

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

From the first dataset estimated values of SRGM with Bass diffusion TEF α=91.58(CPU hours), p=0.02168, and q=0.6848, a=564.1 and r=0.01973 when Δt=1 R_0=0.85 and we let C_1=1, C2 =50, C_3 =100 and T_{LC} =100 the estimated time T_1=65.8 weeks and release time from Eq 31 T_0=46.29 weeks. Now optimal Release Time max[46.3,65.8] is T*=65.8 weeks. Fig 10 shows the change in software cost during the time span. Now total cost of the software at optimal time 9595.

VII. Conclusions

In this paper, we proposed a SRGM incorporating the Exponentiated Weibull testing-effort function that is completely different from the Weibull 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.

Delay

$$\frac{dm_d(t)}{dt} \times \frac{1}{w(t)} = r1 \times \left[a - m_d(t)\right] \tag{34}$$

$$m_d(t) = a \times \left(1 - e^{-r \times W(t)}\right) \tag{35}$$

$$\frac{dm_r(t)}{dt} \times \frac{1}{w(t)} = r2 \times \left(m_d(t) - m_r(t)\right) \tag{36}$$

$$\frac{dm_r(t)}{dt} + r2 \times w(t) \times m_r(t) = r2 \times w(t) \times m_d(t) \tag{37}$$

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

$$m_r(t) \times e^{r2 \times W(t)} = \int_0^t r2 \times w(t) \times m_d(t) \times e^{r2 \times W(t)}\, dt \tag{38}$$

Solving above equation substituting $m_d(t)$

$$m_r(t) = a\left(1 - \left(\frac{\left(r1 \times e^{-r2 \times W(t)} - r2 \times e^{-r1 \times W(t)}\right)}{r1 - r2}\right)\right) \tag{39}$$

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

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

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

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

$$f'(r1) = -r1 \times W(t) \times e^{[-r1 \times W(t)]} - e^{[-r1 \times W(t)]} \tag{40}$$

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

$$\text{And } \frac{f'(r1)}{g'(r1)} = (1 + r1 \times W(t)) \times e^{[-r1 \times W(t)]} \tag{41}$$

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

$$L \equiv \left\{Pr\left(N(t_1) = y_1, N(t_2) = y_2 \ldots . N(t_n)\right) = y_n\right\} \tag{42}$$

$$\prod_{k=1}^{n} \frac{\left[m(t_k) - m(t_{k-1})\right]^{y_k - y_{k-1}}}{(y_k - y_{k-1})} \times \exp\left(-m(t_n)\right) \tag{43}$$

$$\ln(L) = \sum_{k=1}^{n} (y_k - y_{k-1}) \ln\left(m(t_k) - m(t_{k-1})\right) - m(t_n) - \left(\sum_{k=1}^{n} (y_k - y_{k-1})!\right) \tag{44}$$

from Equation13, $\frac{\partial}{\partial a} \ln(L) = 0$

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

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

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

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. Reliability. 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.
[28] Sumantra Chakravarty Adapting BASS-NIU model for product diffusion to software reliability, e-journal "Reliability: Theory & Applications" N0 1 (Vol 2)
[29] P.K.Kapoor, A.Gupta, V.S.S Yadavalli and S.J. Claasen, "Flexible Software Reliability Growth Models", SA Journal of Industrial Engineering Nov 2006 Vol 17 (2) : 109-125

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. Published So many research papers in various International Journals

Shaheda Akthar received Bachelor of computer science and Master 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. Published So many research papers in various International Journals.