Incorporating Generalized Modified Weibull TEF in to Software Reliability Growth Model and Analysis of Optimal Release Policy

Shaik.Mohammad Rafi
Sri Mittapalli College Institute of Technology for Women
NH-5, Tummalapalem, Guntur, A.P, India
Tel: 99-85-083-350 E-mail: rafi_527@rediffmail.com
K.Nageswara Rao
P.V.P.S.I.T, Vijayawada affiliated to J.N.T.U, Kakinada, India
Tel: 94-41-037-218 E-mail: hodcse@pvpsiddhartha.ac.in
Shaheda Akthar
Sri Mittapalli College of Engineering
NH-5, Tummalapalem, Guntur, A.P, India
Tel: 98-85-834-601 E-mail: shaheda.akthar@yahoo.com

Abstract

Software reliability is generally a key factor in software quality. Reliability is an essential ingredient in customer satisfaction. In software development process reliability conveys the information to managers to access the testing effort and time at which software release into the market. Large numbers of papers are published in this context. In this paper we proposed a software reliability growth model with generalized modified weibull testing effort. Performance application of proposed model is demonstrated through real datasets. The experimental results shown that the model gives an excellent performance compared to other models. We also discuss the optimal release time based on reliability requirement and cost criteria.

Keywords: Non homogeneous Poisson process, Mean value function, Optimal software release time, Software reliability growth model, Testing-effort function

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 Exponentiated Weibull TEF

r_2 : Constant fault isolated rate in the Delayed S-shaped model with Exponentiated Weibull TEF

1. Introduction

Reliability is one of the key factors in accessing the quality of the software. In past many papers are published in accessing the software quality through reliability.

The main objective of software industry is to prepare software which is much reliable and satisfy the customer needs. Software reliability represents a customer oriented view of software quality. Many NHPP software reliability growth models are proposed to access the software reliability.

Software reliability is defined as probability of failure free software over a period of time in a given environmental condition.[Lyu, Xie] .Before a software released into market an extensive test is conducted. Software with more errors when released into the market incurs high failure costs [Hoang Pham]. For that more sophisticated testing is needed to track the errors. During the software development many resources are consumed like manpower, test cases. TEF describes test expenditure in testing process. 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 by [Yamada 1986, Bokhari 2006, Kapur 1994 and Haung 1997]. All of them describe the tracking phenomenon with test expenditure.

This paper describes the time dependent behavior of testing – effort by a generalized modified weibull curve. Assuming that the error detection rate in software testing is proportional to the current error content and the proportionality depends on the current test effort, a flexible software reliability growth model based on non homogeneous Poisson process is developed and its applications are presented. Further an optimal release time is calculated based on reliability and cost. Section-2 proposed the test-effort function described by generalized modified weibull curve. In Section – 3 a software reliability growth model with the generalized modified weibull test effort function is discussed. Section -4 contains a model evaluation criteria . Section -5 includes model performance analysis. Section-6 presents the prediction of optimal release time based on the application of the model to software reliability management.

2. Generalized modified Weibull curve TEF

Generalized modified Weibull distribution with five parameters had 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. Another important characteristic of the distribution is that it contains special sub-models, the Weibull expoentiated exponential (Gupta and Kondu 1999, 2001) EW (Mudholkar et at 1995 1996), generalized rayleigh (Kondu and Rakesh 2005), MW (Lai 2003) and Some other distributions.

Current cumulative Testing effort

$$W(t) = \alpha \times \left(1 - e^{-\beta \times t^m \times e^{\lambda \times t}}\right)^{\theta} \tag{1}$$

where α>0,β>0,m>0,λ>0 and θ>0 at t>0 where α is the total effort expenditure ,β controls the scale of the distribution, m and θ are shape parameters. The parameter λ is a kind of accelerating factor in the imperfection time and it works as a factor of fragility in the survival of the individual when the time increases.

Current cumulative testing-effort

$$\frac{d}{dt} W(t) = w(t) = \frac{\alpha\, \theta\, \beta\, e^{\lambda t} \left(e^{-\beta t^m e^{\lambda t}} \left(e^{\beta t^m e^{\lambda t}} - 1\right)\right)^{\theta} \left(t^{m-1} m + t^m \lambda\right)}{e^{\beta t^m e^{\lambda t}} - 1} \tag{2}$$

Following are the some of the special cases;

1) Yamada Weibull curve : for λ=0 and θ=1 then

$$W(t) = \alpha \left(1 - e^{-\beta t^m}\right) \quad \text{Where } \alpha>0, \beta>0 \text{ and } m>0 \tag{3}$$

2) Generalized exponential curve (Quadri 2006):for m=1 and λ=0 there is a generalized exponential testing-effort function; the cumulative testing-effort consumed in time (0,t] is

$$W(t) = \alpha \left(1 - e^{-\beta t}\right)^{\theta} \quad \text{Where } \alpha>0, \beta>0 \text{ and } \theta>0 \tag{4}$$

3) Yamada exponential curve (Yamada 1986): for θ=1, m=1 and λ=0 there is an exponential testing effort function, and the cumulative testing effort consumed in time (0,t]:

$$W(t) = \alpha\left(1 - e^{-\beta t}\right) \quad \text{Where } \alpha>0, \beta>0 \tag{5}$$

4) Yamada Rayleigh curve (Yamada 1986): for θ=1,m=2 and λ=0 there is a rayleigh testing-effort function, the cumulative testing effort consumed in time (0,t] :

$$W(t) = \alpha\left(1 - e^{-\beta t^2}\right) \quad \text{Where } \alpha>0, \beta>0 \tag{6}$$

5) Burr type X curve : for m=2, and λ=0 there is Burr type testing-effort function and the cumulative testing-effort consumed in time (0,t]:

$$W(t) = \alpha\left(1 - e^{-\beta t^2}\right)^{\theta} \quad \text{Where } \alpha>0, \beta>0 \text{ and } \theta>0 \tag{7}$$

6) Extreme value distribution (log-gamma): for m=0 and θ=1 the cumulative testing-effort function

$$W(t) = \alpha\left(1 - e^{-\beta e^{\lambda t}}\right) \quad \text{Where } \alpha>0, \beta>0 \text{ and } \lambda>0 \tag{8}$$

7) Exponentiated weibull curve(Mudholkar 1995,1996): for λ=0 the cumulative testing-effort given by

$$W(t) = \alpha\left(1 - e^{-\beta t^m}\right)^{\theta} \quad \text{Where } \alpha>0, \beta>0, m>0 \text{ and } \theta>0 \tag{9}$$

8) Modified weibull distribution(Lai 2003) : for θ=1 the cumulative testing-effort function

$$W(t) = \alpha\left(1 - e^{-\beta t^m e^{\lambda t}}\right) \quad \text{Where } \alpha>0, \beta>0, m>0 \text{ and } \lambda>0 \tag{10}$$

3. Software Reliability growth model and testing effort functions

3.1 SRGM with Generalized Modified Weibull testing-effort function

The following assumptions are made for software reliability growth modeling (Yamada and Osaki 1985 Yamada 1986, 1993, Kapur 1999, Kuo 2001 Haung and Kuo 2002, Haung 2005)

(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 generalized modified Weibull 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

$$\frac{dm(t)}{dt} \times \frac{1}{w(t)} = r \times [a - m(t)] \tag{11}$$

$$m(t) = a \times \left(1 - e^{-r \times (W(t) - W(0))}\right) \tag{12}$$

Substituting W(t) from eq.(1), we get

$$m(t) = a \times \left(1 - e^{-r \times \left(\alpha \times \left(1 - e^{-\beta \times t^m \times e^{\lambda \times t}}\right)^{\theta}\right)}\right) \tag{13}$$

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

Now failure intensity is given by

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

$$\lambda(t) = \frac{a\, r\, \alpha\, \theta\, \beta\, e^{-r\,\alpha\,\left(1-e^{-\beta\, t^m\, e^{\lambda\, t}}\right)^{\theta}+\lambda\, t}\left(1-e^{-\beta\, t^m\, e^{\lambda\, t}}\right)^{\theta}\left(t^{m-1}\, m + t^m\, \lambda\right)}{e^{\beta\, t^m\, e^{\lambda\, t}}-1} \quad (14)$$

The expected number of errors detected eventually is

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

3.2 Yamada Delayed S-shaped model with Generalized modified Weibull testing-effort function

The delayed 'S' shaped model originally proposed by Yamada [Yamada] 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 follow 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 [Goel 1985, M.Ohba 1984, M.R.Lyu 1996].

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 [Yamada 1983] is modeled by

$$\frac{dm_d(t)}{dt} \times \frac{1}{w(t)} = r_1 \times \left[a - m_d(t)\right] \quad (15)$$

and $\frac{dm_r(t)}{dt} \times \frac{1}{w(t)} = r_2 \times \left[a - m_r(t)\right]$ (16)

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

$$m_r(t) = a \times \left[1 - \frac{\left(r_1 \times e^{\left[-r_2 \times W*(t)\right]} - r_2 \times e^{\left[-r_1 \times W*(t)\right]}\right)}{r1 - r2}\right] \quad (17)$$

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

$$m(t) \cong m_r(t) = a \times \left(1 - \left(1 + r \times W*(t)\right) \times e^{\left[-r \times W*(t)\right]}\right) \quad (18)$$

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

$$\lambda(t) = a \times r^2 \times w(t) \times W*(t) \times e^{\left[-r \times W*(t)\right]} \quad (19)$$

4. Evaluation Criteria

a) The goodness of fit technique

Here we used MSE [M.Xie 1991, C.Y Huang& Kuo 2007, H.Pham 2000] which gives real measure of the difference between actual and predicted values. The MSE defined as

$$MSE = \sum_{i=1}^{k} \frac{\left[m(t_i) - m_i\right]^2}{k} \quad (20)$$

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 (J.Dmusa 1987], can be represented by computing RE for a data set

$$RE = \frac{\left(m\left(t_q\right) - q\right)}{q} \tag{21}$$

In order to check the performance of the Generalized Modified Weibull testing _effort and make a comparison criteria for our evaluations [M.Shepperd and C.Schofield 1997,K. Srinivasan and D.Fisher1995].

d) SSE criteria: SSE can be calculated as :[Hoang Pham 2000]

$$SSE = \sum_{i=1}^{n} \left[y_i - m\left(t_i\right) \right]^2 \tag{22}$$

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.

$$PE_i = Actual(observed)_i - Predicted(estimated)_i \tag{23}$$

$$Bias = \sum_{i=1}^{n} \frac{PE_i}{n} \tag{24}$$

$$Variation = \sqrt{\sum_{i=1}^{n} \frac{\left(PE_i - Bias\right)^2}{n-1}} \tag{25}$$

$$MRE = \frac{\left|M_{estimated} - M_{actual}\right|}{M_{actual}} \tag{26}$$

5. Model Performance Analysis

DS1: the first set of actual data is from the study by Ohba(1984).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) to estimate the parameters. Calculations are given in appendix A. All parameters of other distribution are estimated through LSE. The unknown parameters of GMW are α=52.99(CPU hours), β=0.000031 / week, m=2.933, λ=0.09971, and θ=0.3033. Correspondingly the estimated parameters of Logistic TEF are N=54.84(CPU hours), A=13.03 and b=0.2263/week 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 Generalized modified Weibull TEF, Logistic TEF and Rayleigh TEF. The PE, Bias, Variation, MRE and RMS-PE for GMW, Logistic

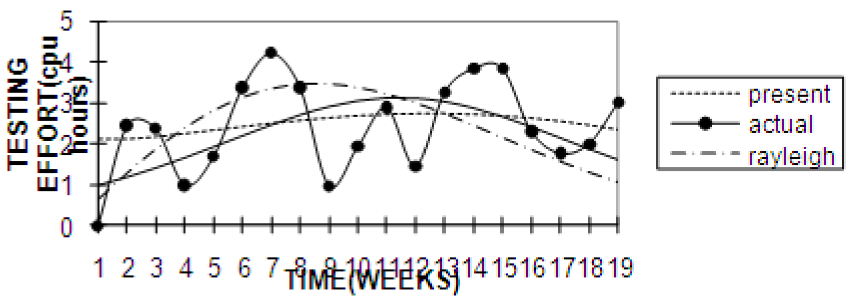

Figure 1. Observed/estimated GMW, Logistic and Rayleigh TEF for DS1

and Rayleigh are listed in Table I. From the TABLE I we can see that GMW 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 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.

DS2: the dataset used here presented by wood 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. Similarly the least square estimates of the parameters for Generalized modified weibull TEF in the

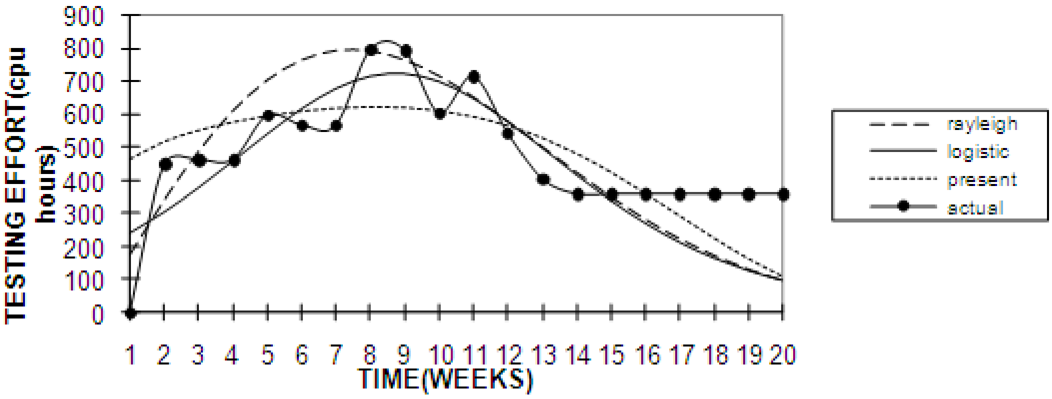

Figure 5. Observed/estimated GMW, Logistic and Rayleigh TEF for DS2

Case of DS2 are α=10310(CPU hours), β=0.00031 / week, m=2.948, λ=0.00065,and θ=0.4147. Correspondingly the estimated parameters of Logistic TEF are N=9974(CPU hours), A=13.22 and b=0.2881/week and Rayleigh TEFN=9669 and b=0.009472/week. The computed Bias, Variation, MRE, and RMS-PE for GMW 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 GMW TEF, Logistic TEF and Rayleigh TEF. From the figure 5 we can observe the GMW 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 Generalized modified Weibull TEF have less MSE than other models.

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 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

$$R(t+\Delta t/t) = \exp(-[m(t+\Delta t/t)-m(t)]) = \exp(-m(\Delta t) \times \exp(-r \times W*(t))) \quad (27)$$

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

$$\ln R = -m(\Delta t) \times \exp(-r \times W*(t)) \quad (28)$$

thus

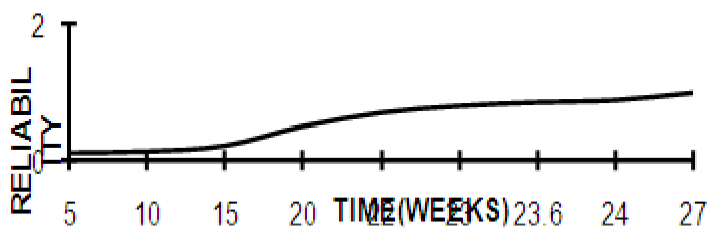

Figure 9. plot for reliability of first dataset at Δt=0.1

By solving the eq (28) and eq(12) we can calculate that the testing time needed to reach the desired reliability. α=52.99(CPU hours), β=0.000031 / week, m=2.933, λ=0.09971, θ=0.3033 and a=567.9and r=0.01954 this software has been run for operational time until it reaches its reliability level 0.80($\Delta t=0.1$) is t=20.1 weeks. To reach the reliability level at 0.85 is t=23.5 weeks. In the way for the dataset2 $\alpha=10310$(CPU hours), $\beta=0.00031$ / week, m=2.948, $\lambda=0.00065$, and $\theta=0.4147$,a=136 and r=0.0001418, software has been run for operational time until it reaches its reliability level 0.85($\Delta t=0.1$) is t=17.4 , its reliability level 0.92($\Delta t=0.1$) is t=20.3, its reliability level 0.960($\Delta t=0.1$) is t=23.4.

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 [2, 16, 17,20]:

$$C(T) = C_1\, m(T) + C_2\left[m(T_{LC}) - m(T)\right] + C_3\left(\int_0^T w(x)\,dx\right) \tag{30}$$

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

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

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

$$\frac{d}{dT}C(T) = C_1\left(\frac{d}{dT}m(T)\right) + C_2\left[-\left(\frac{d}{dT}m(T)\right)\right] + C_3\,w(T) = 0 \tag{32}$$

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

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

$$a\,r\,e^{-r\,W(t)} = r\,(a - m(T)) \tag{33}$$

When T=0 then m(0)=0 and $\frac{\lambda(T)}{w(T)} = a\,r$ when T->∞, then $m(\infty) = a\left(1 - e^{-r\alpha}\right)$

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. (37) 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

$$T_0^{m} \times e^{\lambda \times T_0} = \left\{ -\frac{1}{\beta} \times \log\left[1 - \left(\frac{1}{r \times \alpha} \log\left[a \times r \times \frac{(c2 - c1)}{c3} \right] \right)^{\frac{1}{\theta}} \right] \right\} \quad (34)$$

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 $T_0 \le T$ 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 $0< R_0 <1$[Yamada 1985,Huang 1999]

T^* =opimal software release time or total testing time =max$\{T_0, T_1\}$.

Where T_0 =finite and unique solution T satisfying eq.(30)

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

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

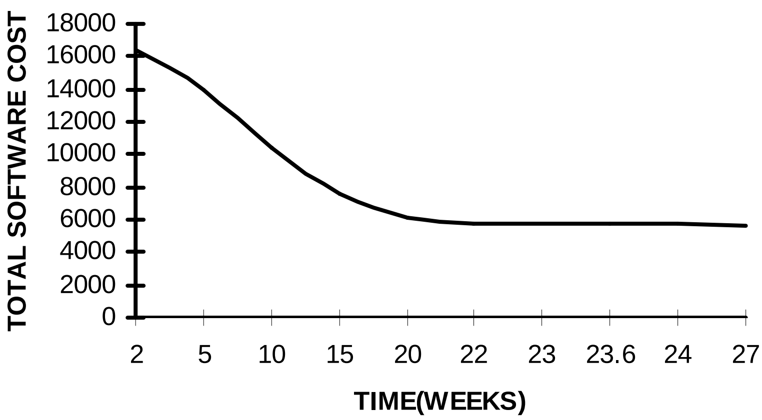

Figure 10. Total software cost for the first dataset vs Time

From the dataset one estimated values of SRGM with GMW TEF α=52.99(CPU hours), β=0.000031 / week, m=2.933, λ=0.09971, θ=0.3033, a=567.9 and r=0.01954 when $\Delta t=0.1$ $R_0=0.85$ and we let $C_1=1$, $C2=50$, $C_3=100$ and $T_{LC}=100$ the estimated time $T_1=23.6$ weeks and release time from eq 30 $T_0=12.35$ weeks. Now optimal Release Time max(12.35,23.6) is T*=23.6 weeks. Fig 10 shows the change in software cost during the time span. Now total cost of the software at optimal time 5713.

Table 1.

TEF	Bias	Variation	MRE	RMS-PE
GMW	0.003061	0.928220	0.007765	0.928215
Logistic	-0.098262	1.306677	0.022246	1.302977
Rayleigh	0.830337	2.169314	0.052676	2.004112

Comparison result for different TEF applied to DS1

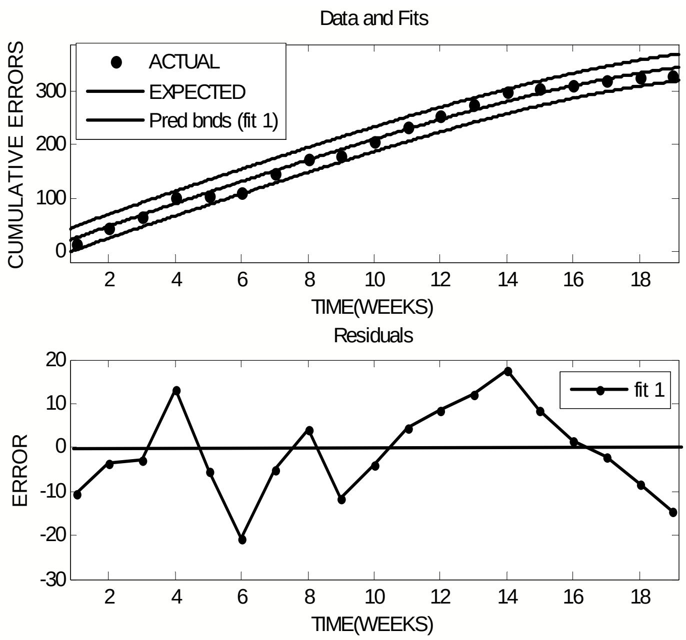

Figure 2. Cumulative and residual error for SRGM with GMW for DS1

Table 2.

Models	a	r	SSE	R^2	MSE
SRGM with GMW TEF	567.9	0.01954	1864	0.9905	109.62
Delayed S shaped model with GMW	352.3	0.1339	3498	0.9822	205.63
SRGM with Logistic TEF	395.6	0.04164	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

Estimated parameter values and model comparison for DS1

Table 3.

Models	a		R	
	Lower	Upper	Lower	Upper
SRGM with Generalized Modified Weibull TEF	456.2	679.6	0.01408	0.025
SRGM with Logistic TEF	358	433.2	0.03399	0.04928
SRGM with Rayleigh TEF	348.6	569.6	0.01651	0.03817
Yamada Delayed S shaped Model with Generalized Modified Weibull TEF	327.7	376.8	0.08019	0.1004
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

95% confidence limit for different selected models(DS1)

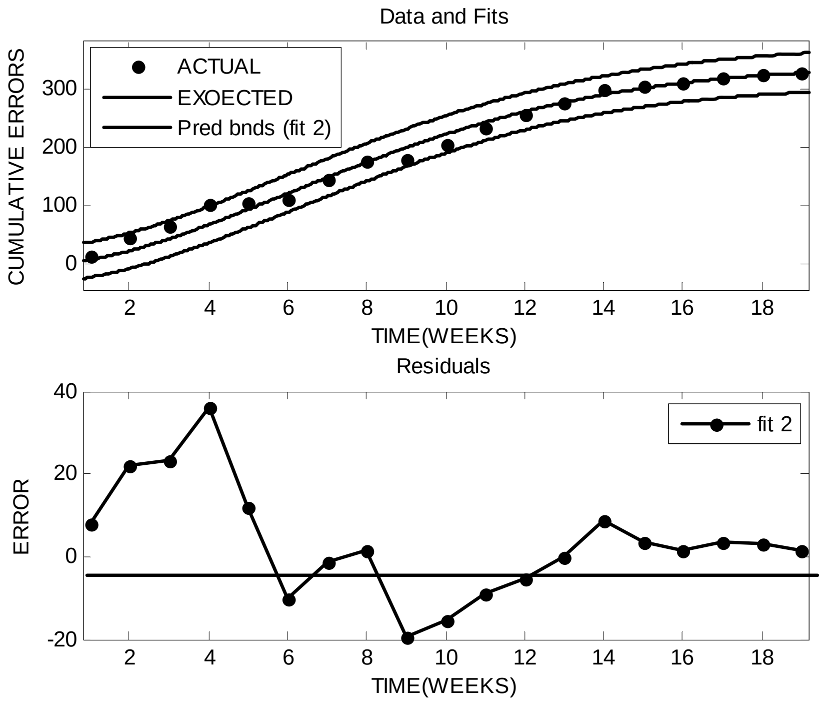

Figure 3. Cumulative and residual error for delayed S shaped model with GMW for DS1

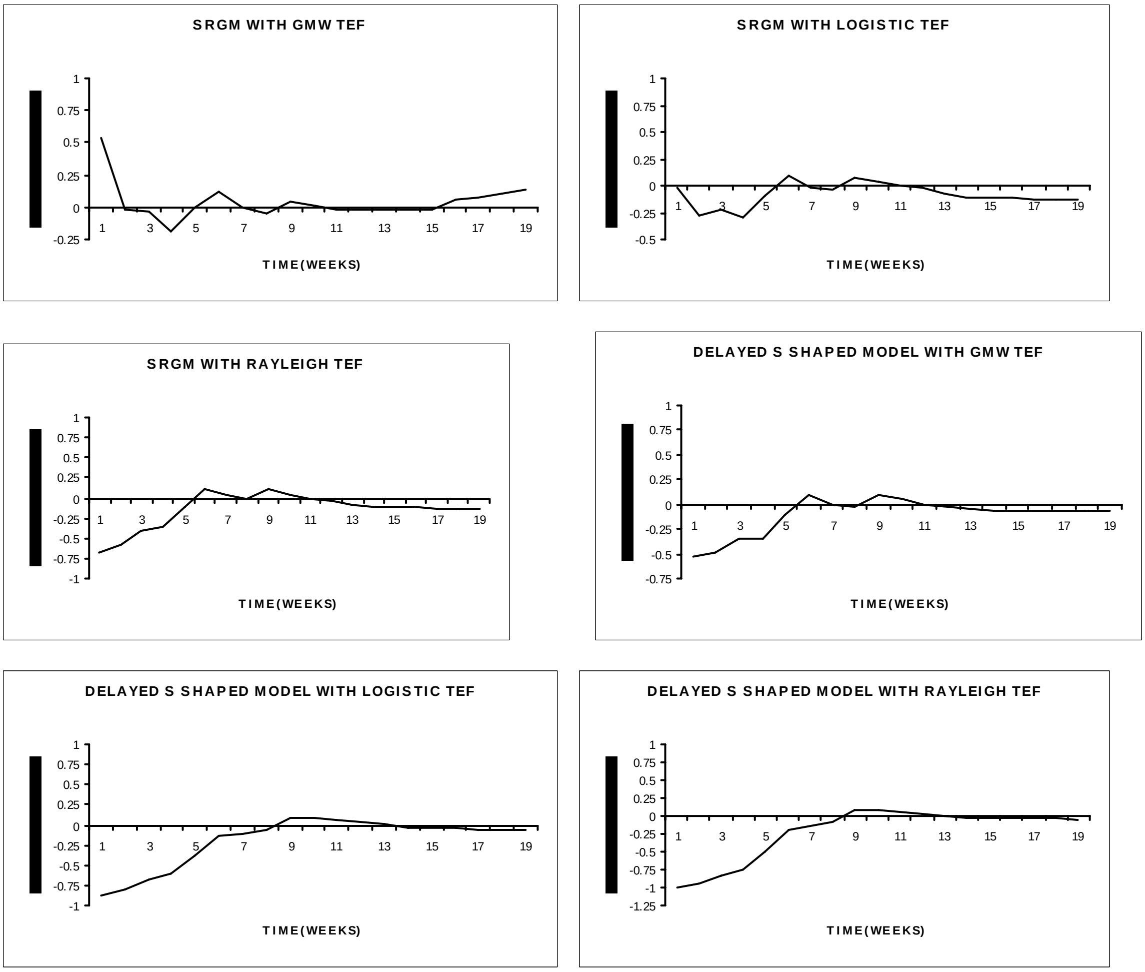

Figure 4. RE curves of selected models compared with actual failure data(DS1)

Table 4.

TEF	Bias	Variation	MRE	RMS-PE
GMW	6.47	137.74	0.020	137.57
Logistic	-19.345	198.44	0.026	197.5
Rayleigh	121.61	322	0.055	298.23

Comparison result for different TEF applied to DS2

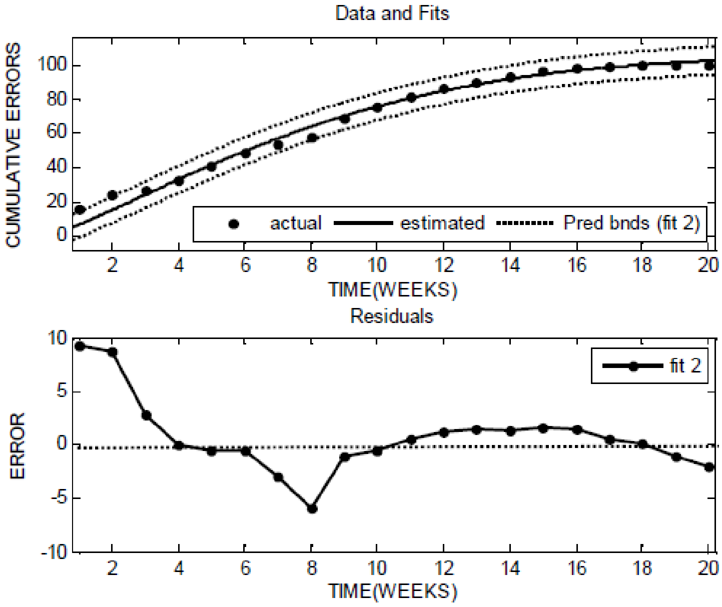

Fig 6. Cumulative and residual error for SRGM with GMW for DS2

Table 5.

Models	a	r	SSE	R^2	MSE
SRGM with GMW TEF	136	0.0001418	229.3	0.9859	13.39
Delayed S shaped model with GMW	103.5	0.0004866	881.5	0.9458	48.97
SRGM with Logistic TEF	112.3	0.0002399	433.1	0.9734	24.06
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

Estimated parameter values and model comparison for DS2

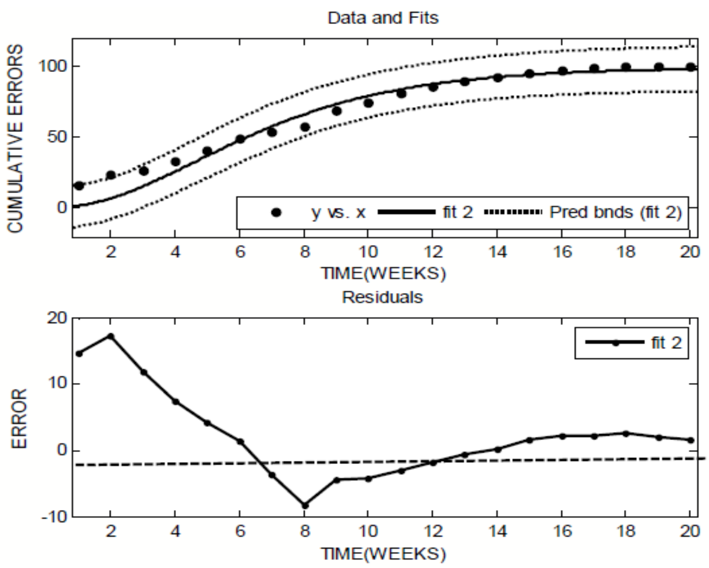

Fig 7. Cumulative and residual error for delayed S shaped model with GMW for DS2

Table 6.

Models	a		R	
	Lower	Upper	Lower	Upper
SRGM with Generalized Modified Weibull TEF	119	153	0.0001108	0.0001728
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 Generalized Modified Weibull TEF	94.74	112.4	0.0004041	0.0005629
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

95% confidence limit for different selected models (DS2)

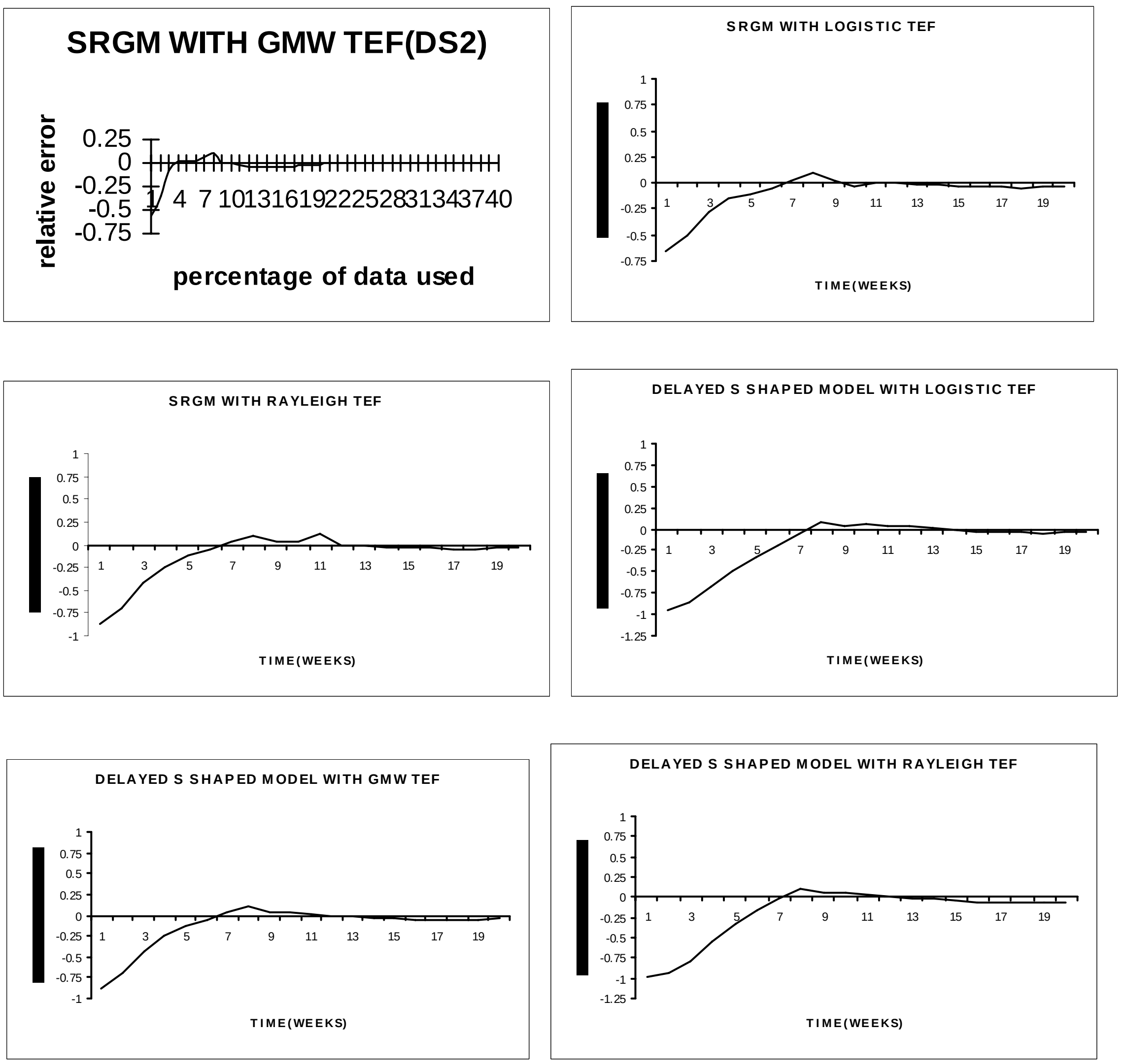

Figure 8. RE curves of selected models compared with actual failure data(DS2)

Appendix A

$$S(\alpha, \beta, m, \lambda, \theta) = \sum_{k=1}^{n} \left(\ln(W_k) - \ln(\alpha) - \theta \times \ln\left(1 - e^{-\beta \times t_k^{m} \times e^{\lambda \times t_k}}\right)\right)^2$$

$$\frac{\partial}{\partial \alpha} S(\alpha, \beta, m, \lambda, \theta) = 0$$

$$\frac{\partial}{\partial \alpha} S(\alpha, \beta, m, \lambda, \theta) = \frac{2\, n\, \ln(\alpha)}{\alpha} + \sum_{k=1}^{n} \left(-\frac{2\, \ln(W_k)}{\alpha} + \frac{2\, \theta\, \ln\left(1 - e^{-\beta\, t_k^m\, e^{\lambda\, t_k}}\right)}{\alpha} \right)$$

$$\frac{2 \left(n \ln(\alpha) + \sum_{k=1}^{n} \left(-\ln(W_k) + \theta \ln\left(1 - e^{-\beta\, t_k^m\, e^{\lambda\, t_k}}\right) \right) \right)}{\alpha} = 0$$

$$\alpha = e^{-\frac{-\left(\sum_{k=1}^{n} \ln(W_k) \right) + \theta \left(\sum_{k=1}^{n} \ln\left(1 - e^{-\beta\, t_k^m\, e^{\lambda\, t_k}}\right) \right)}{n}}$$

$$\frac{\partial}{\partial \beta} S(\alpha, \beta, m, \lambda, \theta) = 0$$

$$\sum_{k=1}^{n} \left(-\frac{2 \ln(W_k)\, \theta\, t_k^m\, e^{\lambda\, t_k}\, e^{-\beta\, t_k^m\, e^{\lambda\, t_k}}}{1 - e^{-\beta\, t_k^m\, e^{\lambda\, t_k}}} + \frac{2 \ln(\alpha)\, \theta\, t_k^m\, e^{\lambda\, t_k}\, e^{-\beta\, t_k^m\, e^{\lambda\, t_k}}}{1 - e^{-\beta\, t_k^m\, e^{\lambda\, t_k}}} + \frac{2\, \theta^2 \ln\left(1 - e^{-\beta\, t_k^m\, e^{\lambda\, t_k}}\right) t_k^m\, e^{\lambda\, t_k}\, e^{-\beta\, t_k^m\, e^{\lambda\, t_k}}}{1 - e^{-\beta\, t_k^m\, e^{\lambda\, t_k}}} \right)$$

$$0 = -2\, \theta \left(\sum_{k=1}^{n} \frac{e^{\lambda\, t_k}\, t_k^m \left(\ln(W_k) - \ln(\alpha) - \theta \ln\left(1 - e^{-\beta\, t_k^m\, e^{\lambda\, t_k}}\right) \right)}{e^{\beta\, t_k^m\, e^{\lambda\, t_k}} - 1} \right)$$

$$\frac{\partial}{\partial m} S(\alpha, \beta, m, \lambda, \theta) = 0$$

$$\sum_{k=1}^{n} \left(-\frac{2 \ln(W_k)\, \theta\, \beta\, t_k^m \ln(t_k)\, e^{\lambda\, t_k}\, e^{-\beta\, t_k^m\, e^{\lambda\, t_k}}}{1 - e^{-\beta\, t_k^m\, e^{\lambda\, t_k}}} + \frac{2 \ln(\alpha)\, \theta\, \beta\, t_k^m \ln(t_k)\, e^{\lambda\, t_k}\, e^{-\beta\, t_k^m\, e^{\lambda\, t_k}}}{1 - e^{-\beta\, t_k^m\, e^{\lambda\, t_k}}} + \frac{2\, \theta^2 \ln\left(1 - e^{-\beta\, t_k^m\, e^{\lambda\, t_k}}\right) \beta\, t_k^m \ln(t_k)\, e^{\lambda\, t_k}\, e^{-\beta\, t_k^m\, e^{\lambda\, t_k}}}{1 - e^{-\beta\, t_k^m\, e^{\lambda\, t_k}}} \right)$$

$$0 = -2\,\theta\,\beta\left(\sum_{k=1}^{n}\frac{\ln(t_k)\,e^{\lambda t_k}\,t_k^m\left(\ln(W_k)-\ln(\alpha)-\theta\ln\left(1-e^{-\beta t_k^m e^{\lambda t_k}}\right)\right)}{e^{\beta t_k^m e^{\lambda t_k}}-1}\right)$$

$$\frac{\partial}{\partial\lambda}S(\alpha,\beta,m,\lambda,\theta)=0$$

$$\sum_{k=1}^{n}\left(-\frac{2\ln(W_k)\,\theta\,\beta\,t_k^m\,t_k\,e^{\lambda t_k}\,e^{-\beta t_k^m e^{\lambda t_k}}}{1-e^{-\beta t_k^m e^{\lambda t_k}}}+\frac{2\ln(\alpha)\,\theta\,\beta\,t_k^m\,t_k\,e^{\lambda t_k}\,e^{-\beta t_k^m e^{\lambda t_k}}}{1-e^{-\beta t_k^m e^{\lambda t_k}}}+\frac{2\theta^2\ln\left(1-e^{-\beta t_k^m e^{\lambda t_k}}\right)\beta\,t_k^m\,t_k\,e^{\lambda t_k}\,e^{-\beta t_k^m e^{\lambda t_k}}}{1-e^{-\beta t_k^m e^{\lambda t_k}}}\right)$$

$$0 = -2\,\beta\,\theta\left(\sum_{k=1}^{n}\frac{t_k\,e^{\lambda t_k}\,t_k^m\left(\ln(W_k)-\ln(\alpha)-\theta\ln\left(1-e^{-\beta t_k^m e^{\lambda t_k}}\right)\right)}{e^{\beta t_k^m e^{\lambda t_k}}-1}\right)$$

$$\frac{\partial}{\partial\theta}S(\alpha,\beta,m,\lambda,\theta)=0$$

$$\sum_{k=1}^{n}\left(-2\ln(W_k)\ln\left(1-e^{-\beta t_k^m e^{\lambda t_k}}\right)+2\ln(\alpha)\ln\left(1-e^{-\beta t_k^m e^{\lambda t_k}}\right)+2\,\theta\ln\left(1-e^{-\beta t_k^m e^{\lambda t_k}}\right)^2\right)$$

$$0 = -2\left(\sum_{k=1}^{n}\left(\ln(W_k)-\ln(\alpha)-\theta\ln\left(1-e^{-\beta t_k^m e^{\lambda t_k}}\right)\right)\ln\left(1-e^{-\beta t_k^m e^{\lambda t_k}}\right)\right)$$

$$\theta = -\frac{-\left(\sum_{k=1}^{n}\ln(W_k)\ln\left(1-e^{-\beta t_k^m e^{\lambda t_k}}\right)\right)+\ln(\alpha)\left(\sum_{k=1}^{n}\ln\left(1-e^{-\beta t_k^m e^{\lambda t_k}}\right)\right)}{\sum_{k=1}^{n}\ln\left(1-e^{-\beta t_k^m e^{\lambda t_k}}\right)^2}$$

Appendix -B

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

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

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

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

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

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

solving above equation substituting $m_d(t)$

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

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

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

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

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

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

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

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

Appendix -C

Using the estimated parameters α, β, m, and θ above, we estimate the reliability growth parameters a and r in (13). 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

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

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

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

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

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

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

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

7. Conclusions & Future work

In this paper, we proposed a SRGM incorporating the Generalized modified Weibull testing effort function that is completely different from the Weibull type Curve. We observed that most of software failure data is time dependent. By incorporating testing effort in to SRGM we can make realistic assumptions about the software failure. The experimental results indicate that our proposed model fits fairly well compared to other models. In future we study the present testing-effort in imperfect debugging environment.

References

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.

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.

C.-Y. Huang, S.-Y. Kuo, J.Y. Chen. (1997). 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.

Goel, A.L. (1985). "Software reliability models: Assumptions, limitations, and applicability", *IEEE Transactions on Software Engineering* SE-11 (1985) 1411-1423.

Hoang Pham and Chin-Diew Lai. (2007). “On Recent Generalizations of the Weibull Distribution” IEEE TRANSACTIONS ON RELIABILITY, VOL. 56, NO. 3, SEPTEMBER 2007.

Huang and S. Y. Kuo. (2002). “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.

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.

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.

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.

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.

Huang, Lyu and Kuo. (2007). “An Assesment of testing effort dependent software reliability Growth model”. *IEEE transactions on Reliability* Vol 56, No: 2, June 2007

K. Pillai and V. S. Sukumaran Nair. (1997). “A model for software development effort and cost estimation,” *IEEE Trans. Software Engineering*, vol. 23, no. 8, August 1997.

K. Srinivasan and D. Fisher. (1995). “Machine learning approaches to estimating software development effort,” *IEEE Trans. Software Engineering*, vol. 21, no. 2, pp. 126–136, 1995.

Kapur, P.K. and Younes, S. (1994). “Modeling an imperfect debugging phenomenon with testing effort”, in Proceedings of 5th International Symposium on Software Reliability Engineering (ISSRE’1994), pp. 178-83.

L. Goel and K. Okumoto. (1978). 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).

Lai, Min Xie, D. N. P. Murthy. (2003). “A Modified Weibull Distribution *“IEEE TRANSACTIONS ON RELIABILITY,* VOL. 52, NO. 1, MARCH 2003.

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

M.R. Lyu. (1996). *Handbook of Software Reliability Engineering,* Mcgraw Hill, 1996.

Mudholkar, G.S. and Srivastava, D.K. (1993). “Exponentiated Weibull family for analyzing bathtub failure-rate data”, *IEEE Transactions on Reliability*, Vol. 42, pp. 299-302.

P. K. Kapur and S. Younes. (1995). “Software reliability growth model with error dependency,” *Microelectronics and Reliability*, vol. 35, no. 2, pp. 273–278, 1995.

Pham, H. (2000). *Software Reliability*, Springer-Verlag, New York, NY.

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

Rameshwar D. Gupta and Debasis Kundu. (2000). “generalized exponential distribution: different method of estimations” *j. statist. comput. simul.*, 2000, vol. 00, pp. 1 – 22 14 November 2000.

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

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

S.Yarnada, S.Osalci. (1985). "Software reliability growth modeling: models and applications", *IEEE Trans. Software Engineering,* vol.l I, no.12, p.1431-1437, December 1985.

Wood. (1996). Predicting software reliability, *IEEE computers* 11 (1996) 69–77.

Xie, M. (1991). *Software Reliability Modeling*, World Scientific Publication, Singapore.

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.

Yamada, S., Ohba, M., Osaki, S. (1983). S-shaped reliability growth modeling for software error detection. *IEEE Trans. Reliability.* 12, 475–484.