OPTIMAL RELEASE POLICIES FOR SOFTWARE RELIABILITY GROWTH MODEL (SRGM) WITH MAINTENANCE COSTS

*Madhu Jain, Sandhya Maheshwari and Kriti Priya

**Department of Mathematics*
Institute of Basic Science, Khandari, Agra, 282002 India

E-mail: madhujain@sancharnet.in

ABSTRACT

A component based or module-based software system is considered in which the software reliability growth model (SRGM) follows a mixed distribution. We address the problem of determining the optimal testing time of software so that the total maintenance costs of the software could be minimized and a desired level of reliability could be achieved as well. The total maintenance costs of the software is obtained by assuming a warranty period in the operational phase of the software during which the cost incurred in the maintenance, is paid by the developer. The present value of the money is considered while calculating the total maintenance costs by including a discount rate. A technique for estimating the parameters of the SRGM is also suggested. Numerical illustrations are provided for testing the validity of the analytical results.

Key words: Software testing, Reliability estimation, Optimal testing time, Warranty cost, Maintenance cost.

1.0 INTRODUCTION

In the present scenario, computer systems are indispensable for society and their need and importance are increasing rapidly. To meet this increasing demand, the complexity of the software products to construct such computer systems, has enhanced to a considerable extent. During the development of such complex software systems, many software faults may occur. To reduce these faults, thorough testing of the software is required so that a highly reliable software system can be developed. The software is expected to be more reliable if the testing time is long, but it leads to a late delivery. On the other hand, shorter testing time gives the unreliable software to the customers as well as increase in the total maintenance costs during the development and operational phases. Thus it is important to determine the optimal time, when the testing should be stopped or when the software should be released to the customers such that the total software maintenance costs could be minimized and a desired level of reliability could be achieved.

The software release problems have attracted the attention of many researchers. Yamada *et al.* (1983) provided an S-shaped reliability growth model for software error detection. Yamada and Osaki (1986) presented an optimal software release policy for a non-homogeneous software error detection rate model. Yamada (1991) discussed software reliability measurement and assessment of various software reliability growth models and data analysis. He (1994) also studied the optimal release problems with the warranty period based on a software maintenance cost model. Kapur and Bhalla (1992) suggested optimal release policies for a flexible reliability growth model. Xia *et al.* (1993) discussed optimal release policies with a learning factor for imperfect debugging. Kapur *et al.* (1993) developed an exponential SRGM with a bound on the number of failures. Zeephongsekul (1996) studied reliability growth of a software model under imperfect debugging and generation of errors. Pham (1996) presented a software cost model with imperfect debugging, random life cycle and penalty cost. Chatterjee *et al.* (1997) studied the joint effect of test effort and learning factor on software reliability and optimal release policies. Pham and Zhang (1999) developed a software cost model with warranty and risk costs. Kimura et al. (1999) analyzed software release problems with warranty cost and reliability requirement. Yang and Xie (2000) studied two different approaches of software reliability i.e. operational reliability and testing reliability. Rattihalli and Zachariah (2002) presented NHPP models for reliability of software with imperfect debugging and testing effort. Tal *et al.* (2002) discussed optimal statistical testing policy for software reliability. More recently, Jain and Priya (2002) suggested optimal policies for software testing time by considering a variable failure detection rate.

While analyzing the reliability of software systems, the estimation of parameters involved in the mathematical models is of great importance. Hossain and Dahiya (1993) estimated the parameters of a non-homogeneous Poisson process for software reliability. Suresh and Babu (1997) estimated the reliability of a software system using a non-homogeneous Poisson process approach. The estimation of parameters for various software reliability growth models (SRGM) can be found in Pham (2000). Rallis and Lansdowne (2001) discussed reliability estimation for a software system.

In real time software systems, the component-based or module-based approach is followed in developing the software. In the reliability analysis of such software systems, the failure pattern depends on the individual modules of the software. Some research works have been done on the reliability analysis of module based software systems. Popstojanova and Trivedi (2001) presented the architecture based approach for quantitatively assessing the component based software systems. They estimated the software reliability analytically, by combining the architecture of the software with the failure behavior of the components and interfaces. Since the reliability of the whole software depends on the reliability of each of the modules, the SRGM should be such that it takes into consideration the behaviour of all the modules of the software.

In this paper, we obtain the optimal software testing time by assuming that the SRGM follows a mixed distribution i.e., Exponential and Rayleigh for different modules. A technique for estimating the parameters of the SRGM is also suggested. The rest of the paper is organized as follows: In section 2, the SRGM is described along with the assumptions and nomenclature. In section 3, the estimation of parameters by the maximum likelihood method is suggested. The maintenance cost analysis is presented in section 4. In section 5, numerical illustrations are provided to examine the optimal testing policies. The concluding remarks and directions for future research work are given in the last section 6.

2.0 MODEL DESCRIPTION

Consider a SRGM in which fault detection is characterized by a mixture of an Exponential and Rayleigh distributions. This signifies that some modules of the software follow the exponential distribution and some other modules follow the Rayleigh distribution with their respective probabilities.

Following notations are being used in the mathematical formulation of the model:

C_0	Initial testing cost
C_t	Testing cost per unit time
C_w	Maintenance cost per fault during the warranty period

T	Release time of the software
T^*	Optimum software release time
T_w	Warranty period
b_1	Fault detection rate for Exponential distribution
b_2	Fault detection rate for Rayleigh distribution
a	Discount rate of the cost
a	Initial number of errors in the software
EC(T)	Expected total maintenance costs of software
$C_W(T)$	Maintenance cost during the warranty period

Following assumptions are made while describing the SRGM:

- The software consists of two types of modules, which follow different failure patterns.
- The failure detection rate is different for different types of modules.
- All faults in the software are mutually independent from the failure detection point of view.
- The number of failures that are detected at any time is proportional to the current number of faults in the software.
- Whenever an error occurs, it is immediately removed and at this instant, no other new errors are introduced in the software.
- There is a warranty period in the operational phase of the software where the maintenance cost is paid by the developer.
- Discounted maintenance cost is considered so as to take care of the present value of the money.

The expected number of failures detected at time t is given by

$$m(t) = 1[p(1\text{-}e^{-b_1 t}) + (1\text{-}p)(1\text{-}e^{-b_2 t^2})] \qquad \ldots(1)$$

where p and (1-p) are the respective probabilities of Exponential and Rayleigh distributions.

The failure intensity function is obtained as follows:

$$\lambda(t) = \frac{d}{dt}[m(t) = \frac{d}{dt}a[p(1\text{-}e^{-b_1 t}) + (1\text{-}p)(1\text{-}e^{-b_2 t2})] \qquad \ldots(2)$$
$$= a[pb1\text{-}e^{-b_1 t}) + 2(1\text{-}p)b_2 t e^{-b_2 t2})]$$

3.0 ESTIMATION OF PARAMETERS

For software reliability prediction, the estimation of parameters is essential. We suggest the maximum likelihood estimation (MLE) technique for estimating the parameters of the SRGM. There are four unknown parameters i.e. a, b_1, b_2, and p in the SRGM described in the previous section.

Let $t_1, t_2, \ldots, t_n$ be the random failure times of n items where $0<t_1<t_2 \ldots <t_n$. Then the log likelihood function (LLF) is given by

$$LLF = \sum_{i=1}^{n} (y_i - y_{i-1}) \log[m(t_i) - m(t_i - 1)] - m(t_n) \qquad \ldots(3)$$

where m(t) is given by (1) and y_i is the cumulative number of detected errors. Now the maximum of the *LLF* is determined by using the following system of equations:

$$0 = \sum_{i=1}^{n} \frac{\frac{\partial}{\partial \varphi} m(t_i) - \frac{\partial}{\partial \varphi} m(t_{i-1})}{m(t_i) - m(t_{i-1})} (y_i - y_{i-1}) - \frac{\partial}{\partial \varphi} m(t_n) \qquad \ldots(4)$$

where all unknown parameters can be put one by one in place of *y*. So we obtain the following set of equations by substituting *p*, b_1, b_2 and *a* respectively.

$$\sum_{i=1}^{n} \frac{(e^{-b_2 t_i^2} - e^{-b_1 t_i}) - (e^{-b_2 t^2_{i-1}} - e^{-b_1 t_{i-1}})(y_i - y_{i-1})}{D} - a(e^{-b_2 t^2 n} - e^{-b_1 t^n}) = 0 \qquad \ldots(5)$$

$$\sum_{i=1}^{n} \frac{p(t_i e^{-b_1 t_i} t_{i-1} - e^{-b_1 t_{i-1}}) - (y_i - y_{i-1})}{D} - apt_n e^{-b_1 t_n} = 0 \qquad \ldots(6)$$

$$\sum_{i=1}^{n} \frac{(1-p)(t_i^2 e^{-b_2 t_i^2} - t_{i-1}^2 e^{-b_2 t^2_{i-1}}) - (y_i - y_{i-1})}{D} - a(1-p) t_n^2 e^{-b_2 t^2_n} = 0 \qquad \ldots(7)$$

$$\sum_{i=1}^{n} (y_i - y_{i-1}) - [p(1 - e^{-b_1 t_n})] + (1-p)(1 - e^{-b_2 t^2_n}) = 0 \qquad \ldots(8)$$

$$\text{where } D = p(e^{-b_1 t_{i-1}} - e^{-b_1 t_i}) + (1-p)(e^{-b_2 t^2_{i-1}} - e^{-b_2 t_i^2}) \qquad \ldots(9)$$

We can easily solve the equations (5) - (8) and obtain the estimated values of unknown parameters i.e.*p*, b_1, b_2 and *a*.

4.0 MAINTENANCE COST ANALYSIS

Here we wish to determine the optimal testing time of a software system so that the total expected maintenance costs of the software can be minimized. For this purpose, we construct a cost model for the software by assuming that there are three types of costs i.e. an initial testing cost, testing cost per unit time and the maintenance cost during the warranty period. Hence, the total expected software maintenance costs are given by

$$EC(T) = C_0 + Ct\int_0^T e^{-\alpha t}dt + C_W(T) \qquad \ldots(10)$$

For calculating the maintenance costs during the warranty period i.e. $C_W(T)$, we consider the following two cases depending on the warranty period:

Case 1: When the warranty cost is of constant length and the software reliability growth is not assumed to occur after the testing phase. Then

$$C_W(T) = C_W \int_0^{T+T_W} \lambda(T)e^{-\alpha t}dt \qquad \ldots(11)$$

Case 2: When the warranty cost is of constant length and the software reliability growth occurs after the testing time. Then

$$C_W(T) = C_W \int_0^{T+T_W} \lambda(t)e^{-\alpha t}dt \qquad \ldots(12)$$

Substituting the value of $C_W(T)$ from equations (11) & (12) in equation (10), we obtain expected total maintenance costs as follows:

In case 1, we obtain $EC(T) = C_0 + C_t\int_0^T e^{-\alpha t}dt + C_W\int_0^{T+T_W} \lambda(T)e^{-\alpha t}dt$ $\quad\ldots(13)$

For case 2, $EC(T) = C_0 + C_t\int_0^T e^{-\alpha t}dt + C_W\int_0^{T+T_W} \lambda(t)e^{-\alpha t}dt$ $\quad\ldots(14)$

Differentiating equations (13) & (14) with respect to T and equating them to zero, we get the optimal testing time T^* as follows:

For case 1, $T^* = T_1$ is given by

$$A = pb_1(b_1+\alpha)e^{-b_1T} + 2b_2(1-p)e^{-b_2T^2}[T(2b_2T+\alpha)-1] \qquad \ldots(15)$$

where $A = \dfrac{C_t \alpha_{T+T_W}}{aC_W(1-e^{-\alpha T_W})}$

and for case 2, $T^* = T_2$ can be determined by solving the following equation

$$e^{-\alpha T}[C_t - C_W apb_1 e^{-b_1 T}(1-e^{-(b_1+\alpha)TW})] + 2ab_2C_W(1-p)\frac{d}{dt}\int_T^{T+T_W}(te^{-(b_2t^2+\alpha t)})dt = 0 \quad \ldots(16)$$

We can see that $\dfrac{d^2EC(T)}{dT^2} > 0 \mid T = T_1$ and $\dfrac{d^2EC(T)}{dT^2} > 0 \mid T = T_2$

Hence $EC(T)$ gives a minimum value at $T^* = T_1$ and $T^* = T_2$ for case 1 and case 2 respectively.

The **Optimal Testing Policy 1** for case 1 is stated as follows:

OTP 1.1 $T^* = T_1$ when $\lambda(0) > \lambda(T_1)$

OTP 1.2 $T^* = 0$ when $\lambda(0) \leq \lambda(T_2)$

and for case 2, the **Optimal Testing Policy 2** is given by

OTP 1.1 $T^* = T_2$ when $\lambda(0) > \lambda(T_2)$

OTP 1.2 $T^* = 0$ when $\lambda(0) \leq \lambda(T_2)$

Now, we find the optimal testing time T_R by imposing the reliability constraint. The software reliability function $R(x|T)$ can be defined as the probability that a software failure does not occur during the time interval $(T, T+x)$. Hence the software reliability is given by

$$R(x|T)=exp[-\{m(T+x)-m(T)\} \quad \ldots(17)$$

From equation (1), we have

$$R(x|T)= exp[a\{pe^{-b_1T}(e^{-b_1x}-1)+(1-p)e^{-b_2T^2}(e^{-b_2(2Tx+x^2)}-1)\}] \quad \ldots(18)$$

Let R_0 be the minimum required software reliability $(0 < R_0 \leq 1)$. Then the optimal software release problem is stated as

Minimize $EC(T)$

subject to $R(x|T) \geq R_0$ ...(19)

Let T_R denote the optimal testing time satisfying the above constraint. Putting $R(x|T) = R_0$ in equation (19), we get,

$$R = exp\,[-\{m(T+x)-m(T)\}] \quad ...(20)$$

Now, we give the optimum release policies for both the cases as follows:

For Case 1: Optimum Testing Policy 3

OTP 3.1 If $\lambda(0) > \lambda(T_1)$ and $R(x|0) < R_0$, then $T^* = \max\{T_I, T_R\}$.

OTP 3.2 If $\lambda(0) > \lambda(T_1)$ and $R(x|0) \geq R_0$, then $T^* = T_1$.

OTP 3.3 If $\lambda(0) \leq \lambda(T_1)$ and $R(x|0) < R_0$, then $T^* = T_R$.

OTP 3.4 *If* $\lambda(0) \leq \lambda(T_1)$ and $R(x|0) \geq R_0$, then $T^* = 0$.

For Case 2: Optimum Testing Policy 4

OTP 4.1 If $\lambda(0) > \lambda(T_2)$ and $R(x|0) < R_0$, then $T^* = \max\{T_2, T_R\}$.

OTP 4.2 If $\lambda(0) > \lambda(T_2)$ and $R(x|0) \geq R_0$, then $T^* = T_2$.

OTP 4.3 If $\lambda(0) \leq \lambda(T_2)$ and $R(x|0) < R_0$, then $T^* = T_R$.

OTP 4.4 *If* $\lambda(0) \leq \lambda(T_2)$ and $R(x|0) \geq R_0$, then $T^* = 0$.

5.0 NUMERICAL ILLUSTRATIONS

In this section, we present numerical illustrations to check the validity of the analytical results obtained in the previous sections. The suggested policies OTP 1-OTP 4 are tested for p=1 and are tabulated in Tables 1-4. The optimal testing times of the respective policies are obtained by varying c_t and T_w. Evidently, the optimal testing time increases with the increase in the warranty period T_w and decreases with the testing cost per unit time c_t. Moreover, we notice that for a particular combination of the values of c_t and T_w, the optimal testing time in table 1 is greater than that in table 2 which implies that T_1 (T^* in case 1) > T_2 (T^* in case 2) i.e. lesser time is required for testing if the software reliability growth is assumed to occur in the warranty period. Further we see that in tables 3 and

4, $T^* = \max\{T_1(T_2), T_R\} = 195.64$ except for some values of T_w where $T^* = T_1(T_2)$. This validates the policies OTP 3 and OTP 4.

Table 1: Optimal Testing Time (T^*) For OTP 1 By Taking a=.001,a=150,b_1=.03,C_w=10

Tw / ct	5	10	15	20	25	30	35	40
5	126.81	149.83	163.26	172.77	180.12	186.12	191.17	195.54
10	103.70	126.72	140.15	149.66	157.02	163.01	168.07	172.43
15	90.19	113.21	126.64	136.15	143.50	149.49	154.55	158.92
20	80.60	103.62	117.05	126.56	133.91	139.91	144.96	149.33
25	73.16	96.18	109.61	119.12	126.47	132.47	137.52	141.89
30	67.08	90.10	103.53	113.04	120.40	126.39	131.45	135.81

Table 2: Optimal Testing Time (T^*) For OTP 1 By Taking a=.001,a=150,b_1=.03,C_w=10

Tw / ct	5	10	15	20	25	30	35	40
5	126.81	149.83	163.26	172.77	180.12	186.12	191.17	195.54
10	103.70	126.72	140.15	149.66	157.02	163.01	168.07	172.43
15	90.19	113.21	126.64	136.15	143.50	149.49	154.55	158.92
20	80.60	103.62	117.05	126.56	133.91	139.91	144.96	149.33
25	73.16	96.18	109.61	119.12	126.47	132.47	137.52	141.89
30	67.08	90.10	103.53	113.04	120.40	126.39	131.45	135.81

Table 3: Optimal Testing Time (T^*) For OTP 3 By Taking a=.001, a=150, b_1=.03, C_w=10, R_0=.99, x=.8

Tw / ct	5	10	15	20	25	30	35	40
5	195.64	195.64	195.64	172.77	180.12	186.12	191.17	195.54
10	195.64	195.64	195.64	195.64	195.64	195.64	168.07	172.43
15	195.64	195.64	195.64	195.64	195.64	195.64	195.64	195.64
20	195.64	195.64	195.64	195.64	195.64	195.64	195.64	195.64
25	195.64	195.64	195.64	195.64	195.64	195.64	195.64	195.64
30	195.64	195.64	195.64	195.64	195.64	195.64	195.64	195.64

Table 4: Optimal Testing Time (T^*) For OTP 4 By Taking a=.001, a=150, b_1=.03, C_w=10, R_0=.99, x=.8

Tw / ct	5	10	15	20	25	30	35	40
5	195.64	195.64	195.64	195.64	168.45	172.31	175.29	177.65
10	195.64	195.64	195.64	195.64	195.64	195.64	195.64	195.64
15	195.64	195.64	195.64	195.64	195.64	195.64	195.64	195.64
20	195.64	195.64	195.64	195.64	195.64	195.64	195.64	195.64
25	195.64	195.64	195.64	195.64	195.64	195.64	195.64	195.64
30	195.64	195.64	195.64	195.64	195.64	195.64	195.64	195.64

Figures 1(*a*)-1(*d*) and 2(*a*)-2(*d*) depict the variation of the total maintenance costs *ECT* with the testing time *T* for various parameters, for case 1 and case 2 respectively. It can be inferred that *ECT* first decreases and then shows the increasing trend with *T* for all the parameters for both the cases. The reason behind the high value of *ECT* in the beginning is the initial testing cost, which is taken in determining *ECT*. Also the total maintenance cost is comparatively higher in case 2 than in case 1, for all the parameters which signifies that more money is required for the maintenance if the software reliability growth occurs

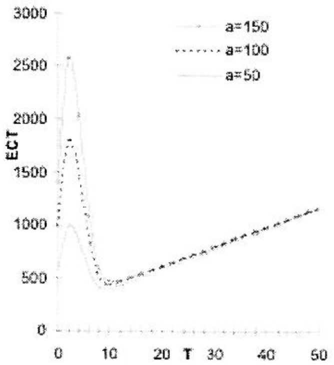

Figure (1a): Expected total maintenance costs by varying a (for case 1)
C_0=150, alpha=.001,C_1=20, C_w=10,p=.2, b_2=.05,b_1=.03, T_w=10

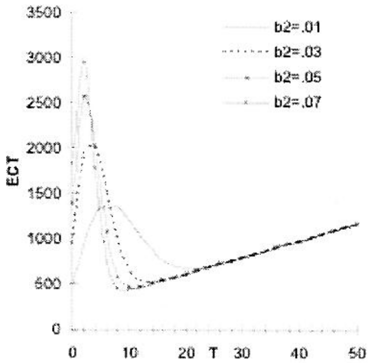

Figure (1b): Expected total maintenance costs by varying b_2 (for case 1)
C_0=150, a=150, C_1=20,C_w=10 p=.2, b_1=.03, alpha=.001,T_w=10

in the operational phase also. Further from figures 1(*a*), 2(*a*), 1(*b*) and 2(*b*), we notice that in both the cases, *ECT* increases with the initial number of errors, *a* and decreases with the failure detection rate b_2. Figures 1(*c-d*) and 2(*c-d*) show that *ECT* increases with the discount rate *a* and with the parameter *p* as well.

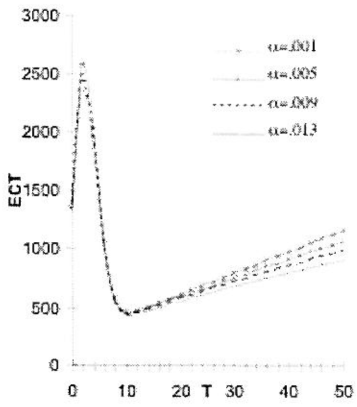

Figure 1c: Expected total maintenance costs by varying a (for case 1)
C_0=150, a=150, C_t=20, C_w=10, p=.2, b_2=.05,b_1=.03, T_w=10

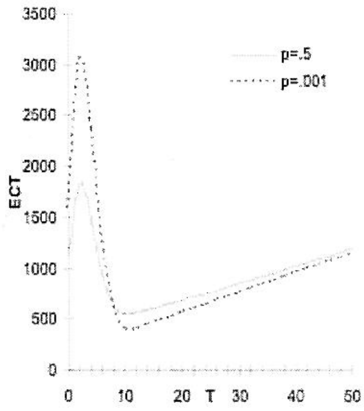

Figure 1d: Expected total maintenance costs by varying a (for case 1)
C_0=150,a=150, C_t=20, C_w= 10, b_2=.05,b_1=.03,alpha=.001,T_w=10

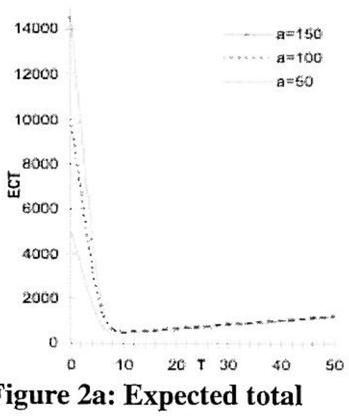

Figure 2a: Expected total maintenance costs by varying a (for case 2)
C_0=150, alpha=.001, C_t=20, Cw=10, p=.2,b1=.03, b2=.05, Tw=10

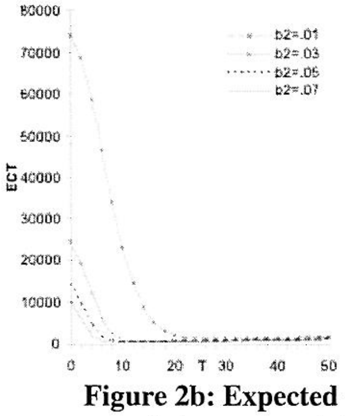

Figure 2b: Expected total maintenance costs by varying b_2 (for case 2)
C_0=150, alpha=.001,C_t=20, C_w=10, p=.2, a=150,b_1=.03, T_w=10

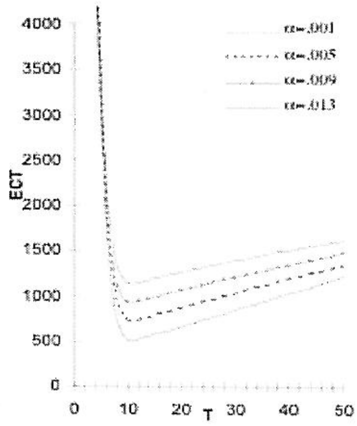

Figure 2c: Expected total maintenance costs by varying
(for case 2)
C_0=150,p=.2, C_t=20,C_w=10, a=150, b_1=.03, b_2=.05, T_w=10

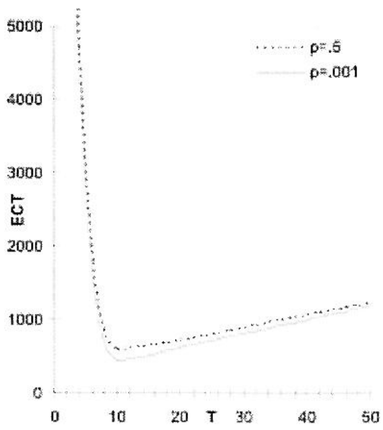

Figure 2d: Expected total maintenance costs by varying p
(for case 2)
C_0=150,alpha=.001,C_t=20,C_w=10, a=150,b_1=.03,b_2=.05, T_w=10

Figures 3(*a-c*) demonstrate the variation of the reliability *R* of the software with *T* for the parameters *a, p* and b_2 respectively. It is clear that the reliability first increases (decreases) with b_2 (*a*) and then it becomes constant.

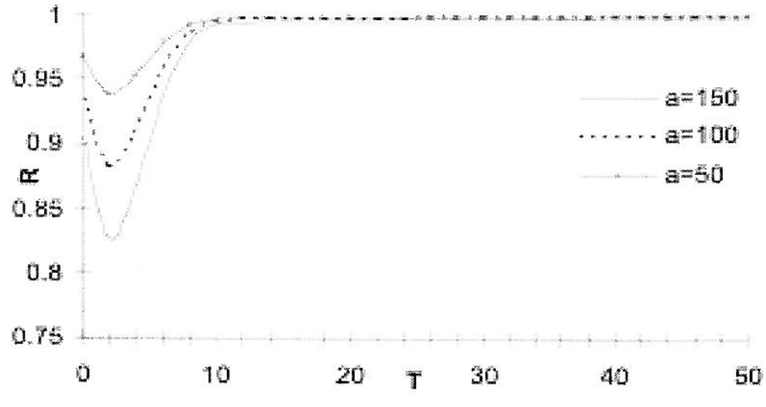

Figure 3a: Software reliability Vs T for different a

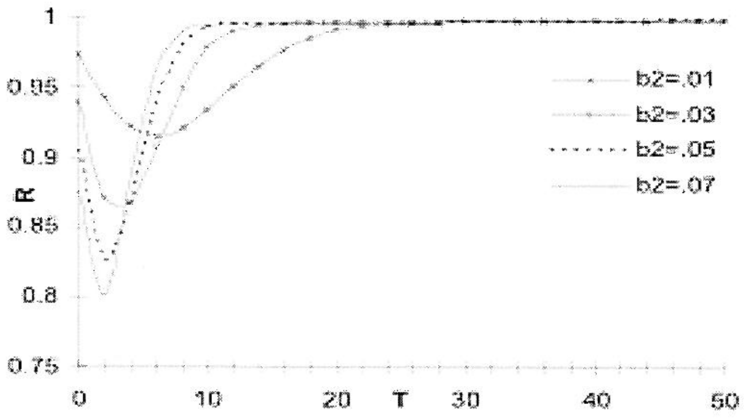

Figure 3b: Software reliability Vs. T by varying b_2

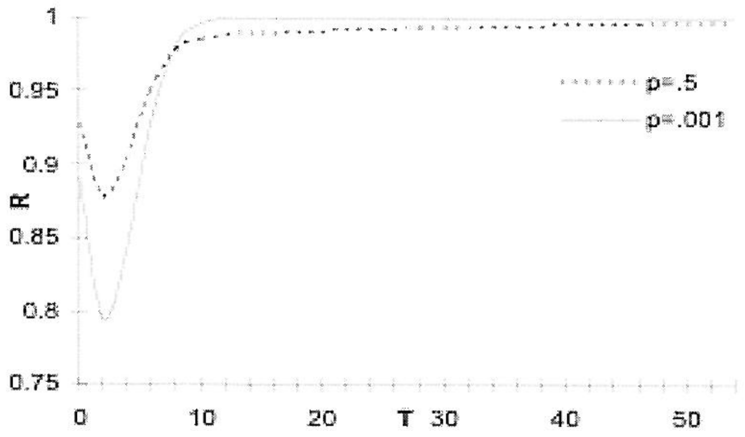

Figure 3c: Software reliability Vs. T for different probabilities

Figure 4 illustrates the effect of imposing the reliability constraint on the total maintenance costs of the software. It can be observed that without the reliability constraint, the optimal testing time comes out to be $T^{*}=T1=10$ which only minimizes ECT which is 463.101, and the corresponding reliability achieved is equal to 0.91. But if the desired level of reliability R_0 to be achieved is 0.99, then T^{*} comes out to be $T^{*}=T_R=24$ which gives ET=685.056. Hence we can conclude that more time is required for testing if a pre-decided level of reliability has to be achieved thereby minimizing the total maintenance costs of the software.

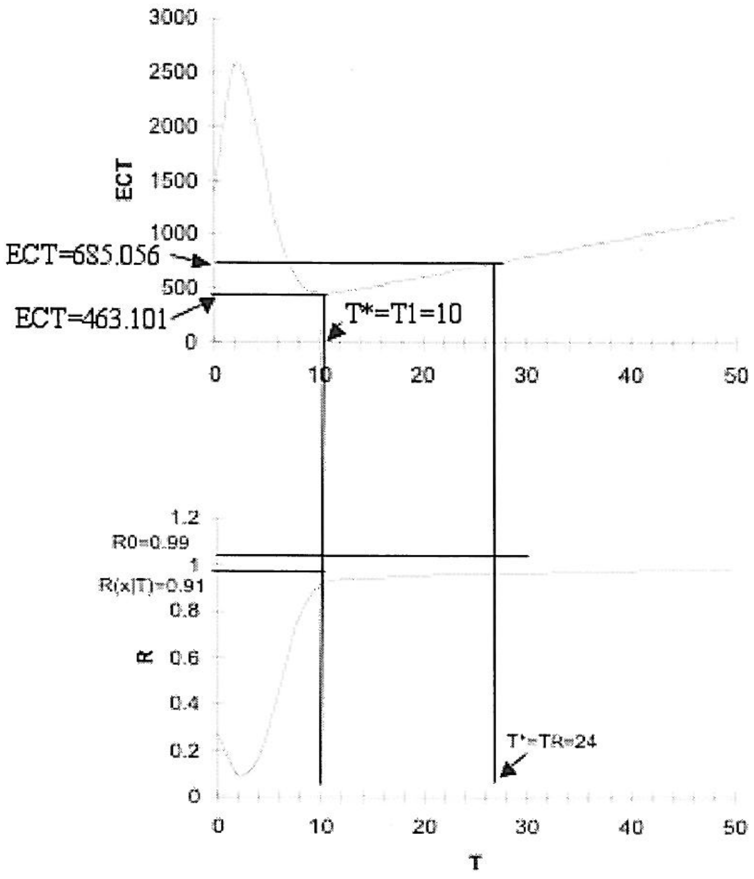

Figure 4: The optimum release time with costs and reliability requirement

6.0 CONCLUSION

It can be concluded from the analysis of the maintenance costs of the software that more time is required for testing the software if the reliability constraint is considered along with the minimization of the maintenance costs of the software. Furthermore, the optimal time required for testing is comparatively lesser if the software reliability growth is assumed to occur in the warranty period.

The optimal testing time of module-based software is determined by minimizing the total maintenance costs of the software subject to the reliability constraint. The *SRGM* developed can be successfully employed for analyzing the real time software systems, which consist of two types of modules following different failure patterns. The suggested *MLE* technique can also be used for estimating the parameters of other *SRGMs*.

The warranty cost taken in constructing the cost model can be further improved by considering a hybrid warranty policy in which case some costs are paid by the customer also. The suggested policies can be further extended to software consisting of more than two types of modules in which direction of the work is going on.

7.0 REFERENCES

Chatterjee, S., Misra, R. B. & Alam, S. S. (1997). Joint effect of test effort and learning factor on software reliability and optimal release policy. *International Journal of Systems Sciences, 28*, 391-396.

Hossain, S. A. & Dahiya (1993). Estimating the parameters of a non-homogeneous Poisson process model for software reliability. *IEEE Trans. Reliab., 42,* No. 2, 604-612.

Jain, M. & Priya, K. (2002). Optimal policies for software testing time. *Journal of the CSI, 32*, No. 3, 25-30.

Kapur, P. K. & Bhalla, V. K. (1992). Optimal release policies for a flexible software reliability growth model. *Reliab. Engg. and System Safety, 35*, 49-54.

Kapur, P. K., Bhushan, S. & Younes, S.(1993). "An exponential SRGM with a bound on the number of failures". *Microelectronics and Reliability, 33*, 1245-1249.

Kimura. M., Toyota, T. & Yamada, S. (1999). Economic analysis of software release problems with warranty cost and reliability requirement. *Reliab. Engg. and System Safety, 66*, 49-55.

Pham, H. (1996). A software cost model with imperfect debugging, random life cycle and penalty cost. *International Journal of Systems Science, 27*, 455-463.

Pham, H. & Zhang, X. M. (1999). A software cost model with warranty and risk costs. *IEEE Transactions on Computers, 48*, 71-75.

Pham, H. (2000). *Software Reliability* (pp. 106-115). Singapore: Springer-Verlag Pte. Ltd.

Popstojanova, K. G. & Trivedi, K. S. (2001). Architecture-based approach to reliability assessment of software systems, *Performance Evaluation, 45*, 179-204.

Rallis, N. E & Lansdowne, Z. F. (2001). Reliability estimation for a software system with sequential independent reviews, *IEEE Trans. on Software Engg., 27*, No.12, 1057-1061.

Rattihalli, R. N. & Zachariah, B. (2002). NHPP models for reliability of software with imperfect debugging and testing effort, *OPSEARCH, 39*, No. 3 & 4, *215-229.*

Suresh, N. & Babu, A. J. G. (1997). Software reliability estimation and optimization, a non-homogeneous Poisson process approach, *International Journal of Quality and Reliability. Management, 14*, 287-300.

Tal, U., Collin, M. C. & Ben dell, A. (2002). An optimal statistical testing policy for software reliability, *Demonstration of Safety Critical Systems, 137*, No. 3, 544-557.

Xia, G., Zeephongsekul, P. & Kumar, S. (1993). Optimal software release policy with a learning factor for imperfect debugging, *Microelectron. and Reliab., 33*, 81-86.

Yamada, S., Ohba, M. & Osaki, S. (1983). S-shaped reliability growth modeling for software error detection, *IEEE Trans. on Reliab., R-32*, 475-478.

Yamada, S. & Osaki, S. (1986). Optimal software release policy for a non-homogeneous software error detection rate model, *Microelectron. and Reliab., 26*, 691-702.

Yamada, S. (1991). Software quality/reliability measurement and assessment software reliability growth models and data analysis, *J. Info. Process, 14*, No. 3, 254-266.

Yamada, S. (1994). Optimal release problems with warranty period based on a software maintenance cost model, *Trans. IPS Japan, 35*, No.10, 2197-2202 in Japanese.

Yang, B. & Xie, M. (2000). A study of operational and testing reliability in software reliability analysis, *Reliab. Engg. and System Safety, 70,* 323-329.

Zeephongsekul, P. (1996). "Reliability growth of a software model under imperfect debugging and generation of errors". *Microelectronics and Reliability, 36, 1475-1482.*