World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:1, No:4, 2007

Analysis of Testing and Operational Software Reliability in SRGM based on NHPP

S. Thirumurugan, and D. R. Prince Williams

Open Science Index, Computer and Information Engineering Vol:1, No:4, 2007 publications.waset.org/2740.pdf

***Abstract*—**Software Reliability is one of the key factors in the software development process. Software Reliability is estimated using reliability models based on Non Homogenous Poisson Process. In most of the literature the Software Reliability is predicted only in testing phase. So it leads to wrong decision-making concept. In this paper, two Software Reliability concepts, testing and operational phase are studied in detail. Using S-Shaped Software Reliability Growth Model (SRGM) and Exponential SRGM, the testing and operational reliability values are obtained. Finally two reliability values are compared and optimal release time is investigated.

***Keywords*—**Error Detection Rate, Estimation of Parameters, Instantaneous Failure Rate, Mean Value Function, Non Homogenous Poisson Process (NHPP), Software Reliability.

I. Introduction

SOFTWARE reliability is one of the most important areas in the software industry. Software reliability is defined as the probability of failure-free operation of software for a specified time in a specified environment. Software reliability represents a customer-oriented view of software quality and it relates the practical operations rather than static. The objective of the study of software reliability is to increase the probability that a completed program will work as intended by the customer. Hence measuring and computing the reliability of a software system is very important. In the past few decades, many software reliability growth models (SRGM) are developed to evaluate the software reliability[2]–[9]. Most of these models are developed for the analysis of software failure data collected during the testing stage only and not on the operational stage. An important class of SRGMs that has been widely studied and used by practitioners is NHPP models. This class of model has a number of advantages in practice. However there are two different software reliability concepts, that is, the testing reliability which is the probability of no failure occurring during the testing phase and the operational reliability which is the probability of no failure occurring during the operational phase. In the software testing process identified faults are removed and similar failure will not occur again and the failure rate will depend on the testing time. But in the operational phase, fault removal is not considered, because the user will have an experience at a constant failure occurrence rate over the time. Hence these two concepts are different. Software testing is a very costly process. The software should reach the customer as quickly as possible with the desired level of reliability. The time at which the customer gets the software is called software release time. A software release time, which gives minimum cost spent for software testing and obtains maximum reliability, is called optimal release time. In most of the literature [11]–[13] optimal release time is calculated with respect to the testing phase, without considering the operational phase. In[10], Yang studied the optimal release time in operational phase using exponential SRGM, that is, he assumed that the software failure rate occurs in exponential form(strictly decreasing) with respect to the testing time *t*. But in real life situation the software failure rate occurs in S-Shaped form (first-increasing-then-decreasing) with respect to the testing time *t*. Hence it is necessary to investigate the optimal release time, considering operational phase, using S-Shaped SRGM.

In this paper, different reliability (testing phase, operational phase) values are obtained using exponential SRGM, S-Shaped SRGM and the importance of operational reliability concept in decision making are also explained.

Manuscript received March 27, 2007. This work was supported in part by Directorate of Technical Education, Ministry of Manpower, Sultanate of Oman.

S. Thirumurugan is with Department of Information Technology, Salalah College of Technology, PO No: 608 Salalah, Sultanate of Oman, PC 211 (e-mail: thirums@gmail.com).

D. R. Prince Williams is with Department of Information Technology, Salalah College of Technology, PO No: 608 Salalah, Sultanate of Oman, PC 211 (e-mail: princeshree1@gmail.com).

II. Testing and Operational Reliability

Software reliability should be defined as the probability of failure free operation for a specified period of time in a specified environment. As in most of the cases, the failure history of the software is known. Software reliability can be expressed in terms of conditional probability [1].

$$R(x/t) = P_r(x_k > x/t_{k-1} = t) \quad (1)$$

which represents the reliability during the next failure interval of *x* units given the failure history during *t* units.

The cumulative number of failure experienced upto time *t* is $\{N(t), t \geq 0\}$. It can be normally be modelled as an NHPP with mean value function of $m(t)$.

A. The Testing Reliability

During the testing stage the software is improving. At this stage, the process follows the NHPP and we have

Open Science Index, Computer and Information Engineering Vol:1, No:4, 2007 publications.waset.org/2740.pdf

TABLE I
NOTATION

Symbol	Meaning
m(t)	Expected number of observed failures during the time interval [0,t)
T_o	The inflection point of *m(t)*
$\lambda(t)$	Instantaneous failure rate at time *t*
λ_r	The failure intensity at the time when the software is released
X_k	The time interval between $(k-1)$ and k^{th} failure, k =1,2,...
S_k	k^{th} failure occurrence time
$R(x/t)$	Software Reliability
$R_{te}(x/t)$	Testing Reliability
$R_{op}(x/t)$	Operational Reliability
a	Expected number of faults in the software when the testing begins
b	Fault detection rate per remaining fault in the software
C_1	Expected cost of removing a fault during the testing
C_2	Expected cost of removing a fault during the operation
C_3	Expected cost per unit time of testing
T	Release time of the software
$C(T)$	Total expected cost of the software development
*	Superscript which denote the optimal solution

$$P_r\{[N(t+x) - N(t)] = k\} = \frac{[m(t+x)-m(t)]^k}{k!}\exp\left(-[m(t+x) - m(t)]\right), \quad k = 0,1,2,...$$

Hence the reliability of the software is

$$R_{te}(x/t) = \exp\{-[m(t+x) - m(t)]\} \quad (2)$$

B. The Operational Reliability

If the software has been tested for t units of time and then released to customers, then the time to next failure will follow exponential distribution with parameter $\lambda_r(t)$, where $\lambda_r(t)$ is the failure intensity function of the original NHPP calculated at time *t*. Then the reliability function is given by

$$R_{op}(x/t) = \exp\{-\lambda_r(t)\, x\} \quad (3)$$

Equation (3) is called operational reliability which measures the software reliability in the operational phase, see Fig. 1 (Exponential SRGM) and Fig. 2 (S-Shaped SRGM). Clearly (2) and (3) are not same and hence different estimates will be obtained. Hence it is clear that operational and testing reliability concepts are different. It is important to study the actual difference and compare their properties. During the testing phase the mean value function *m(t)* is usually either exponential[1] or S-shaped[13]. In this paper, exponential failure rate model, exponential SRGM and S-Shaped failure rate model, S-Shaped SRGM is used to obtain the two reliability values.

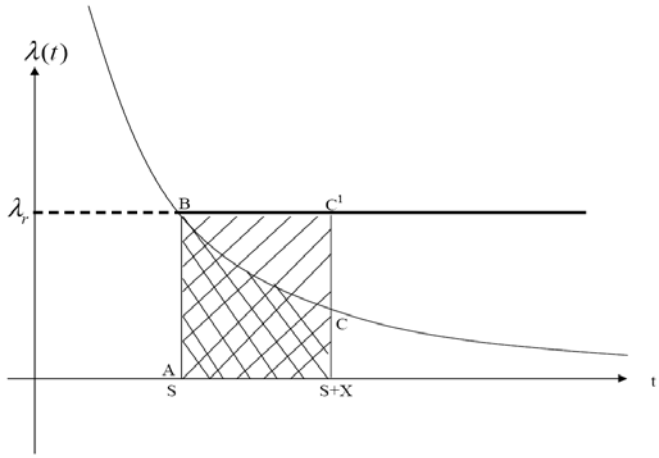

Fig. 1 The testing reliability and the operational reliability (when λ (t) to strictly decreasing for t ≥ 0)

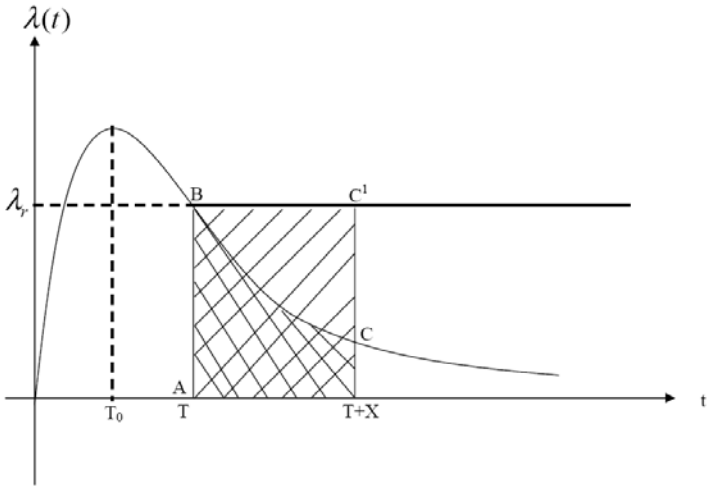

Fig. 2 The testing reliability and the operational reliability (when λ (t) to first-increasing –then-decreasing. $T_0 > T$ and $T_1 < T + x$

III. SOFTWARE RELIABILITY GROWTH MODELS

A. Exponential SRGM

The general assumption in this model is that existing defects do not introduce any new faults during the development and correction process. Hence this ideal process of perfect debugging allows the reliability to increase throughout the testing process. Exponential SRGM and proposed SRGM fall in this category.

The exponential SRGM is given by Goel and Okumoto[1]. It has a mean value function

$$m(t) = a[1-e^{-bt}],\ a > 0,\ b > 0 \quad (4)$$

where

a is the initial error content in the software.

b is the error detection rate per error at time *t*.

B. S-Shaped SRGM

The general assumption in this model is that all defects during the development and correction process do not

Open Science Index, Computer and Information Engineering Vol:1, No:4, 2007 publications.waset.org/2740.pdf

introduce any new defects. Hence this ideal process of perfect debugging allows the reliability to increase throughout the testing process. The S-Shaped SRGM falls in this category.

The S-Shaped SRGM is given by Yamada *et al* [11]. It has a mean value function

$$m(t) = a[1-(1+bt)e^{-bt}],\ a>0,\ b>0 \qquad (5)$$

where

a is the initial error content in the software.

b is the error detection rate per error at time *t*.

C. Software Failure Data

This data is originally from the U.S. Navy Fleet Computer Programming center and consist of the errors in the development of software of the real time, multi computer complex which forms the core of the Naval Tactical Data System (NTDS). The NTDS software consisted of some 38 different modules. The data are trouble reports or 'software anomaly reports' for one of the large models. This software failure data is taken from [1]. The time (days) between software failures is given in Table II.

TABLE II
SOFTWARE FAILURE DATA TABLE

Error No.	Time Between Errors (Days)	Cumulative Time (Days)
1	9	9
2	12	21
3	11	32
4	4	36
5	7	43
6	2	45
7	5	50
8	8	58
9	5	63
10	7	70
11	1	71
12	6	77
13	1	78
14	9	87
15	4	91
16	1	92
17	3	95
18	3	98
19	6	104
20	1	105
21	11	116
22	33	149
23	7	156
24	91	47
25	2	249
26	1	250

D. Parameters Estimation

The number of failures observed up to time *t* is denoted by *m(t)*, which is a random quantity. Assuming a Poisson distribution, the probability that *m*(*t*) has the value *z* is given by

$$P_r[m(t)=z] = \frac{[m(t)]^z}{z!}e^{-m(t)}$$

where $m(t)$ is a mean value function of the models.

Suppose that z_i number of failures have been observed upto t_i and z_{i-1} number of failures have been observed upto t_{i-1}, where $t_i > t_{i-1}$ and $z_i > z_{i-1}$, then conditional probability of $m(t_i)=z_i$ given $m(t_{i-1})=z_{i-1}$ is given by

$$P_r[m(t_i)=z_i / m(t_{i-1})=z_{i-1}] = P_r[m(t_i)-m(t_{i-1})=z_i-z_{i-1}]$$
$$= \frac{[m(t_i)-m(t_{i-1})]^{z_i-z_{i-1}}}{(z_i-z_{i-1})!}e^{-[m(t_i)-m(t_{i-1})]}$$

The model parameter *a* and *b* are estimated as follows. Assume that the data are available in the form of pairs (t_i, z_i), $i = 1,2,...n$, where z_i is the cumulative number of error detected upto time t_i. The joint probabilities for the pairs of data (t_i, z_i), $i = 1,2,...n$, are observed. They are

$$P_r(m(t_0)=0, m(t_1)=z_1,...,m(t_n)=z_n) = \prod_{i=1}^{n}\frac{[m(t_i)-m(t_{i-1})]^{(z_i-z_{i-1})}}{(z_i-z_{i-1})!}e^{-[m(t_i)-m(t_{i-1})]}$$

where $m(t_i)$ is mean value function of the model with time t_i, i=1,2,...n and $m(t_0)=0$.

This joint probability function may be used as the likelihood function for estimating the model parameters. The estimates can be found by maximizing the log likelihood L.

$$L = \sum_{i=1}^{n}(z_i-z_{i-1})\ \ln\left(m(t_i)-m(t_{i-1})\right) - \sum_{i=1}^{n}\ln\left((z_i-z_{i-1})!\right) - m(t_n) \qquad (6)$$

In (6) substitute $m(t_i) = a[1-e^{-bt_i}]$.

Taking the derivatives of *L* with respect to *a* and *b* and setting them equal to zero, we obtain

$$\hat{a} = \frac{z_n}{1-e^{-bt_n}} \qquad (7)$$

$$\frac{t_n e^{-bt_n}}{1-e^{-bt_n}} = \sum_{i=1}^{n}(z_i-z_{i-1})\left[\frac{t_i e^{-bt_i}-t_{i-1}e^{-bt_{i-1}}}{e^{-bt_{i-1}}-e^{-bt_i}}\right] \qquad (8)$$

Solving (7) and (8) numerically we get estimated values of a and b. Using the data set of Table II the estimated values are $\hat{a} = 33.99$, $\hat{b} = 0.00579$. The graphical representation of actual data and estimated values are shown in Fig. 3.

In (6) substitute $m(t_i) = a[1-(1+bt_i)e^{-bt_i}]$. Taking the derivatives of *L* with respect to *a* and *b* and setting them equal to zero, we obtain

$$\hat{a} = \frac{z_n}{1-(1+bt_n)e^{-bt_n}} \qquad (9)$$

$$\frac{zt_n^2 e^{-bt_n}}{1-(1+bt_n)e^{-bt_n}} = \sum_{i=1}^{n}(z_i-z_{i-1})\left[\frac{t_i^2 e^{-bt_i}-t_{i-1}^2 e^{-bt_{i-1}}}{(1+bt_{i-1})e^{-bt_{i-1}}-(1+bt_i)e^{-bt_i}}\right] \qquad (10)$$

Solving (9) and (10) numerically, we get estimated values of *a* and *b*. Using the data set of Table II the estimated values

Open Science Index, Computer and Information Engineering Vol:1, No:4, 2007 publications.waset.org/2740.pdf

are $\hat{a} = 27.0553$,$\hat{b} = 0.0202$ .The graphical representation of actual data and estimated values are shown in Fig. 4.

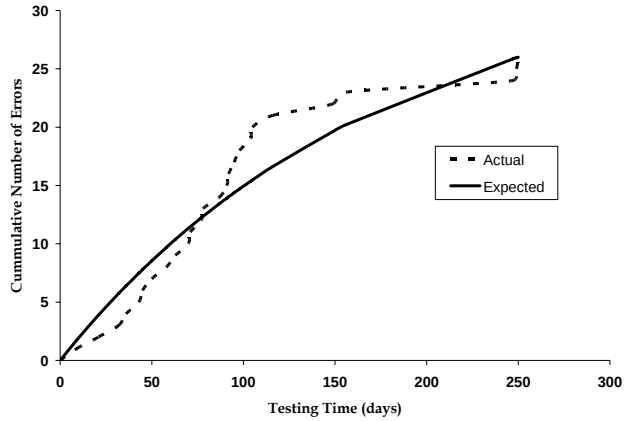

Fig. 3 Goodness of Fit (Exponential SRGM)

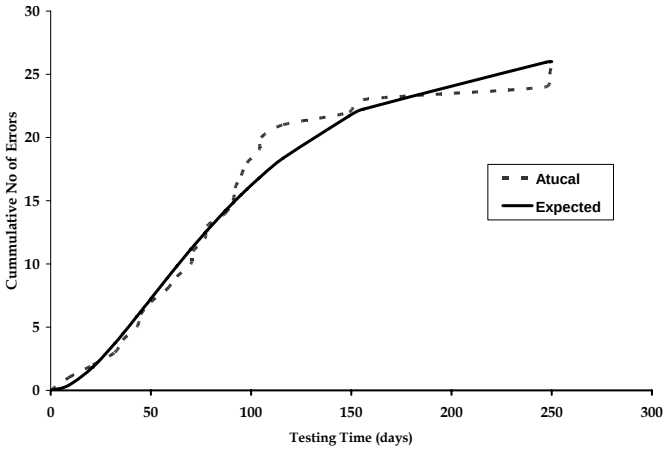

Fig. 4 Goodness of Fit (S Shaped SRGM)

IV. The Testing and Operational Reliability concept in Optimal Release Time

The determination of the software release time is an important aspect in the software development process. Since the minimum reliability level is an essential requirement to be achieved, the usage of different software reliability definitions will certainly have an impact on the optimal release time obtained.

The optimal release time problem is commonly formulated as

$$\min \quad C(T) \tag{11}$$

subject to

$$R_{te}(x/t) \geq R_0 \tag{12}$$

In most of the literature [2]–[5], [9] the testing reliability of the software is given by

$$\exp\{-[\, m(t+x) - m(t)]\} \geq R_0 \tag{13}$$

The software is tested T unit of time and then it is released to customers, there will be no reliability growth during the operational phase, thus it is more appropriate to state the reliability as the operational reliability requirement.

$$R_{op}(-\lambda(T)\, x) \geq R_0 \tag{14}$$

The optimal release time problem formulated in (11) and (13) are hereafter referred as P_1 and (11), (14) are referred as P_2 .

Define

T_R^1 : The minimum value that satisfies (13) $T_R^1 \geq 0$.

T_R^2 : The minimum value that satisfies (14), $T_R^2 \geq 0$.

T_C : The one that minimizes $C(T)$, $T_C \geq 0$.

$T_{P_2}^*$: The optimal solution to P_1.

$T_{P_2}^*$: The optimal solution to P_2.

Theorem 4.1[10]: When $\lambda(t)$ is strictly decreasing for $t \geq 0$ then

Case 1: If $R_{op}(x/0) \geq R_0$ then $T_{P_1}^* = T_{P_2}^* = T_C$

Case 2: If $R_{te}(x/0) \geq R_0 \geq R_{op}(x/0)$ then $T_{P_1}^* = T_C$ and $T_{P_2}^* = \max(T_C, T_R^2)$

(a) If $T_C \geq T_R^2$ then $T_{P_1}^* = T_{P_2}^* = T_C$

(b) If $T_C < T_R^2$ then $T_{P_1}^* = T_C < T_{P_2}^* = T_R^2$

Case 3: If $R_{te}(x/0) < R_0$ then $T_{P_1}^* = \max(T_C, T_R^1)$ and $T_{P_2}^* = \max(T_C, T_R^2)$

(a) If $T_C \geq T_R^2$ then $T_{P_1}^* = T_{P_2}^* = T_C$.

(b) If $T_R^1 < T_C < T_R^2$ then $T_{P_1}^* = T_C < T_{P_2}^* = T_R^2$

(c) If $T_C < T_R^1$ then $T_{P_1}^* = T_R^1 < T_{P_2}^* = T_R^2$.

Theorem 4.2: When $\lambda(t)$ is first-increasing-then-decreasing and T_0 the inflection point of $m(t)$, then

Case 1: If $T_0 \leq T$ then $R_{op}(x/T) < R_{te}(x/T)$

Case 2: If If $T_0 > T$ then let T_1 be the solution to $\lambda(T) = \lambda(T_1), T_1 > T$, we have

(a) If $T_1 \geq T + x$ then $R_{op}(x/T) > R_{te}(x/T)$

(b) If $T_1 < T + x$ then

$$\begin{cases} R_{op}(x/T) > R_{te}(x/T), & \text{if } M < 0 \\ R_{op}(x/T) = R_{te}(x/T), & \text{if } M = 0 \\ R_{op}(x/T) < R_{te}(x/T), & \text{if } M > 0 \end{cases}$$

where $M \equiv \lambda_r x - m(T+x) + m(T)$.

The proof is simple and straight forward.

Theorem 4.3: If $\lambda(t)$ is first-increasing-then-decreasing and $T_0 \leq T$, then

Case 1: If $R_{op}(x/T_0) \geq R_0$ then $T_{P_1}^* = T_{P_2}^* = \max(T_C, T_0)$.

Case 2: If $R_{te}(x/T_0) \geq R_0 > R_{op}(x/T_0)$ then

$T_{P_1}^* = \max(T_C, T_0)$, $T_{P_2}^* = \max(T_C, T_R^2)$ and

Open Science Index, Computer and Information Engineering Vol:1, No:4, 2007 publications.waset.org/2740.pdf

(a) If $T_C \geq T_R^2$ then $T_{P_1}^* = T_{P_2}^* = T_C$

(b) If $T_0 < T_C < T_R^2$, then $T_{P_1}^* = T_C < T_{P_2}^* = T_R^2$

(c) If $T_C \leq T_0$, then $T_{P_1}^* = T_0 < T_{P_2}^* = T_R^2$

Case 3: If $R_{te}(x/T_0) < R_0$ then $T_{P_1}^* = \max(T_C, T_R^1)$,

$T_{P_2}^* = \max(T_C, T_R^2)$ and

(a) If $T_C \geq T_R^2$ then $T_{P_1}^* = T_{P_2}^* = T_C$

(b) If $T_R^1 < T_C < T_R^2$, then $T_{P_1}^* = T_C < T_{P_2}^* = T_R^2$

(c) If $T_C < T_R^1$ then $T_{P_1}^* = T_R^1 < T_{P_2}^* = T_R^2$.

The proof is simple and straight forward.

From Theorem 4.1 and 4.3 it can be clearly seen that $T_{P_1}^* \leq T_{P_2}^*$, which always holds for both exponential SRGM and S-Shaped SRGM. That is, for the optimal release time problem if the testing reliability is adopted, then a shorter T^* will be obtained. This leads to an over-optimistic estimation of the required testing time and the software will be incorrectly released before it reaches the required reliability level. In fact, the operational reliability is meaningful to the customers. Thus the optimal release problem should be formulated as P_2 rather than P_1.

V. Numerical Illustration

For illustrative purposes, we consider a numerical example of the optimal release time problem. The testing process is modeled by (4) or (5) and adopted cost model is given in [13]

$$C(T) = C_1 m(T) + C_2[m(\infty) - m(T)] + C_3 T \qquad (15)$$

A. Using Exponential SRGM

Using the data set of Table II, the maximum likelihood estimates are found to be $\hat{a} = 33.99$ and $\hat{b} = 0.00579$

Assume $x = 100, R_0 = 0.8, C_1 = 200, C_2 = 1500$ and $C_3 = 10$ then

$$T_C = \frac{1}{b}\ln\left[\frac{ab(C_3 - C_1)}{C_3}\right] = 560$$

$$T_R^1 = \frac{1}{b}\ln\left[\frac{a(1-\exp(-bx))}{\ln(1/R_0)}\right] = 727$$

$$T_R^2 = \frac{1}{b}\ln\left[\frac{abx}{\ln(1/R_0)}\right] = 774$$

This case, if $R_{te}(x/0) < R_0$ then $T_{P_1}^* = 727$ and $T_{P_2}^* = 774$ from Theorem 4.1 of case (3). If $T_{P_1}^* = 727$ testing reliability of the software under this solution is 0.746 which does not satisfy the reliability requirement of 0.8. But $T_{P_2}^* = 774$ operational reliability gives the desired reliability level see Fig. 5.

B. Using S-Shaped SRGM

Using the data set of Table II, the maximum likelihood estimates are found to be $\hat{a} = 27.055262$ and $\hat{b} = 0.020172$.

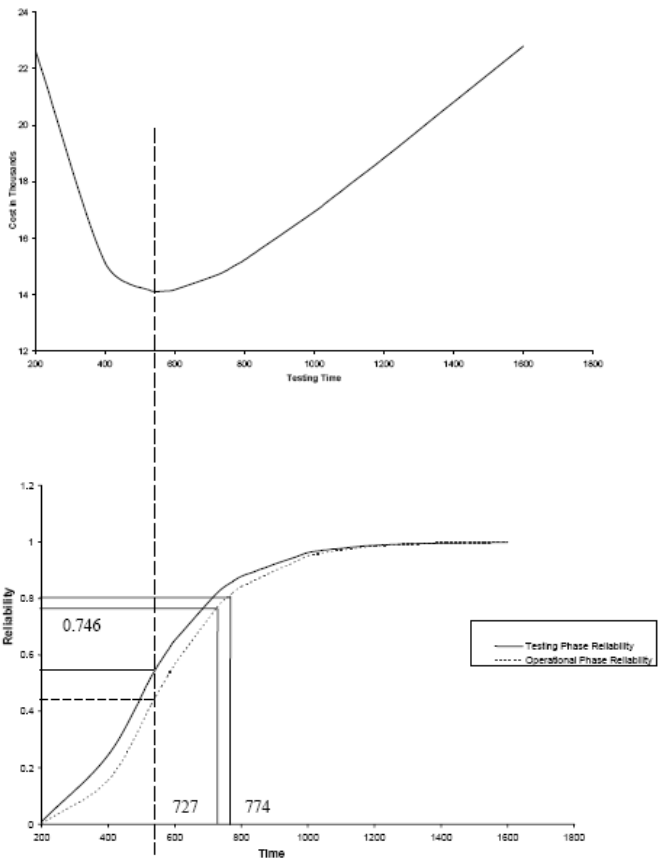

Fig. 5 Reliability Comparison Curve (Exponential SRGM)

Assume $x = 100, R_0 = 0.8, C_1 = 200, C_2 = 1500$ and $C_3 = 10$. In this case T_C is obtained as

$$T_C \exp[-bT_C] = \frac{1}{ab^2}\ln\left(\frac{C_3}{C_2 - C_1}\right) \qquad (16)$$

Solving (16), we get $T_C = 300$.

$$[1 + b(T_R^1 + x)]\, e^{-b(T_R^1 + x)} - [1 + bT_R^1]\, e^{-bT_R^1} = \ln\left(\frac{R_0}{a}\right) \qquad (17)$$

Solving (17), we get $T_R^1 = 329.611$

$$T_R^2 \exp[-bT_R^2] = \frac{1}{ab^2 x}\ln\left(\frac{1}{R_0}\right) \qquad (18)$$

Solving (18), we get $T_R^2 = 372.659$ and $T_0 = \frac{1}{b} = 49.57$

From the above results $R_{te}(x/T_0) < R_0$ and the mean value function m(t) is s-shaped, from Theorem 4.3 it corresponds to(c) of case 3. Therefore $T_{P_1}^* = 329.611$ and $T_{P_2}^* = 372.659$. Hence from Theorem 4.3 of (c) the optimum release time of the problem P_1 is $T_{P_1}^* = 329.611$. However the operational reliability of the software under the solution is 0.63, which does not satisfy the reliability requirement of 0.8 see Fig. 6.

The solution to the optimal release time problem P_2 is $T_{P_2}^* = 372.659$. If the reliability requirement is used as testing reliability requirement, it will lead to inadequate

Open Science Index, Computer and Information Engineering Vol:1, No:4, 2007 publications.waset.org/2740.pdf

software testing time and the required software reliability will not be reached.

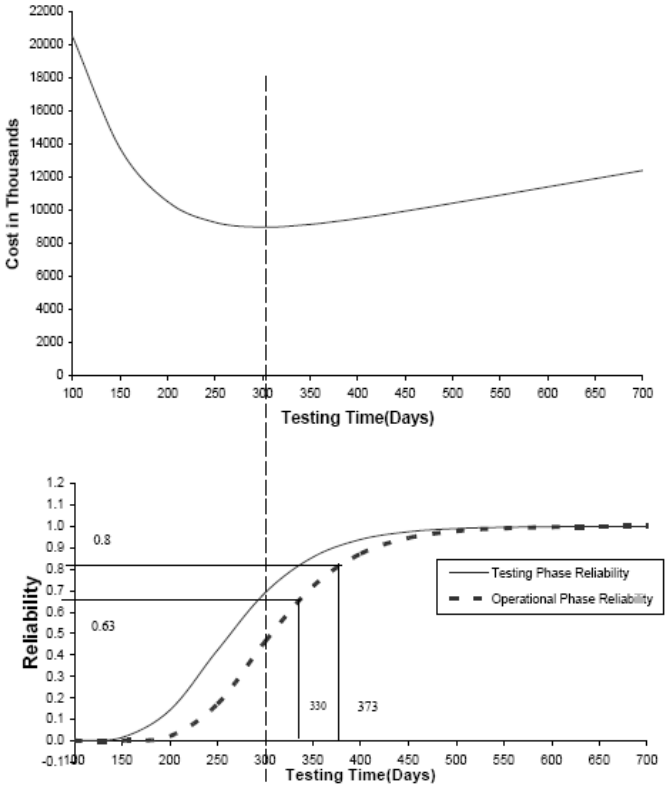

Fig. 6 Reliability Comparison Curve(S-Shaped SRGM)

VI. Conclusion

Decision-Making is one of the most important factors of Software Reliability. Any software should reach the customer only after the software reaches the desired level of reliability. In this paper two software reliability concepts of testing and operational phase reliability are presented. These two concepts are different and they should be used carefully in decision-making. If the testing reliability concept is used, it will give incorrect results and thus mislead in the decision-making. We have used exponential and S-Shaped SRG models as illustrated and proved that Operational Reliability lesser than testing reliability at any given time for these two models. Hence it is recommended that the operational reliability concept may be adopted for the software release time problem and in other related decision-making process.

References

[1] A.L. Goel and K. Okumoto, "Time Dependent Error Detection Rate Model for Software Reliability and other performance measure," *IEEE Trans. Reliab.,* vol. R-28, no. 3, pp.206-211, 1979.

[2] N. Kareer, P.K. Kapur, P.S. Grover, An S-Shaped Software Reliability Growth Models with Two types of Errors, *Micro. Reliab.,* vol. 30, no. 6, pp. 1085-1090, 1990.

[3] J. Musa and K.Okumoto, Software Reliability: Measurement, Predication, Application ,McGraw-Hill, Inc., New York, 1987

[4] M.Obha, "Software Reliability Analysis Models," *IBM. J. Res. Deve.*, vol. 28, no. 4, pp.428-443, 1984.

[5] D. R. Prince Williams and P. Vivekanandan, "Truncated Software Reliability Growth Model," *Korean J. of Computational and Applied Mathematics*, vol. 9, no. 2, pp. 591-599, May. 2002.

[6] A.K.Sheikh, J.K.Boah and M. Younas, "Truncated Extreme value model for pipeline Reliability," *Reliab. Engineering and System Safety*, vol. 25, pp. 1–14, 1989.

[7] M. Xie, "Practical Software Reliability Analysis," *Comp. Sci. and Informatics*, vol. 25, no. 3, pp. 17-25, 1995.

[8] S.Yamada, J. Hishitani and S.Osaka, "Software Reliability Measurement and Assessment based on Non-homogenous Poisson Process Models: A Survey," *Micro. Reliab.*, vol. 32, no. 12, pp. 1763–1773, 1992.

[9] B.Yang and M.Xie, "Optimal testing-time allocation for modular systems," *I. J. of Quali. and Reliab. Mange.,* vol. 18, no. 8, pp. 854-863, 2000.

[10] B.Yang and M.Xie, "A Study of Operational and Testing reliability Analysis," *Reliab. and System Safety*, vol. 70, no. 3, pp. 323-329, 2000.

[11] S.Yamada, M.Obha and S.Osaki, "S-Shaped reliability growth modelling for software error dedection," *IEEE Trans. on Reliab.,* vol. R32, pp. 475-478, 1987.

[12] S.Yamada and S.Osaki, "Optimal software release policy for a nonhomogeneous software error detection rate model," *IEEE Trans. on Reliab*., vol. R-34, pp. 422-424, 1985.

[13] S.Yamada and S.Osaki, "Cost-reliability optimal release policies for software systems," *Reliab. Engineering and System Safety*, vol. 26, pp. 691-702, 1986.

S. Thirumurugan graduated from Scared Heart College, University of Madras, Chennai and obtained Post Graduate degree and Ph.D from College of Engineering Guindy, Anna University, Chennai, India. From June 2001 to July 2004 he worked as Mathematics Faculty in College of Engineering Guindy, Anna University, Chennai, India. From September 2004, he is working as Mathematics Faculty in Department of Information Technology, Salalah College of Technology, Salalah, Sultanate of Oman. His area of specialization is Network Reliability, Software Reliability and its applications. He has so far published several research papers and some papers have been communicated. He has also participated and presented technical papers in National/International Conferences. He also received a best paper award from Indian Institute of Management, Kozhikode, India for the paper presentation in the Annual Conference.

D. R. Prince Williams, received his Masters degree in Mathematics (1991) and Master of Philosophy in Mathematics (1992) from Pachiyappa's College, University of Madras, Chennai, India. He received his Ph.D (1999-2004) from Department of Mathematics, Anna University, Chennai, India. From 1992 to 2004 he worked as Mathematics Faculty in various Engineering Colleges in Chennai, India. From 2005, he is working as Mathematics Faculty in Department of Information Technology, Salalah College of Technology, Salalah, Sultanate of Oman. He has published many papers in national and international journals. His research interests are in the areas of Fuzzy Algebraic Structures, Software Reliability and Mathematical Modeling.