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

S. Thirumurugan, and D. R. Prince Williams

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. But in real life situation the software failure rate occurs in S-Shaped form (first-increasing-then-decreasing) with respect to the testing time. 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.

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].

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

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
Hence the reliability of the software is

\[ R_{op}(x/t) = \exp\left\{-\lambda_e(t)x\right\} \]  

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.

### 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

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

where

- \( a \) is the initial error content in the software.
- \( b \) is the error detection rate per fault at time \( t \).
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

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

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>
<thead>
<tr>
<th>Error No.</th>
<th>Time Between Errors (Days)</th>
<th>Cumulative Time (Days)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>9</td>
<td>9</td>
</tr>
<tr>
<td>2</td>
<td>12</td>
<td>21</td>
</tr>
<tr>
<td>3</td>
<td>11</td>
<td>32</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>36</td>
</tr>
<tr>
<td>5</td>
<td>7</td>
<td>43</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>45</td>
</tr>
<tr>
<td>7</td>
<td>5</td>
<td>50</td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>58</td>
</tr>
<tr>
<td>9</td>
<td>5</td>
<td>63</td>
</tr>
<tr>
<td>10</td>
<td>7</td>
<td>70</td>
</tr>
<tr>
<td>11</td>
<td>1</td>
<td>71</td>
</tr>
<tr>
<td>12</td>
<td>6</td>
<td>77</td>
</tr>
<tr>
<td>13</td>
<td>1</td>
<td>87</td>
</tr>
<tr>
<td>14</td>
<td>9</td>
<td>97</td>
</tr>
<tr>
<td>15</td>
<td>4</td>
<td>91</td>
</tr>
<tr>
<td>16</td>
<td>1</td>
<td>92</td>
</tr>
<tr>
<td>17</td>
<td>3</td>
<td>95</td>
</tr>
<tr>
<td>18</td>
<td>3</td>
<td>98</td>
</tr>
<tr>
<td>19</td>
<td>6</td>
<td>104</td>
</tr>
<tr>
<td>20</td>
<td>1</td>
<td>105</td>
</tr>
<tr>
<td>21</td>
<td>11</td>
<td>116</td>
</tr>
<tr>
<td>22</td>
<td>33</td>
<td>149</td>
</tr>
<tr>
<td>23</td>
<td>7</td>
<td>156</td>
</tr>
<tr>
<td>24</td>
<td>7</td>
<td>157</td>
</tr>
<tr>
<td>25</td>
<td>91</td>
<td>249</td>
</tr>
<tr>
<td>26</td>
<td>2</td>
<td>250</td>
</tr>
</tbody>
</table>

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

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

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

\[ P_z[m(t_i) = z_i | m(t_{i-1}) = z_{i-1}] = P_z[m(t_i) - m(t_{i-1}) = z_i - z_{i-1} | m(t_{i-1}) = z_{i-1}] = \frac{(m(t_i) - m(t_{i-1}))^z}{z!} e^{-(m(t_i) - m(t_{i-1}))} \]

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 up to time \( t_i \). The joint probabilities for the pairs of data \( (t_i, z_i), \ i = 1,2,..n \) are observed. They are

\[ P_z(m(t_i) = z_i, m(t_j) = z_j) = \frac{(m(t_i) - m(t_j))^z}{z!} e^{-(m(t_i) - m(t_j))} \]

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 \( L \).

\[ L = \sum_{i=1}^{n} \ln(m(t_i) - m(t_{i-1})) - \sum_{i=1}^{n} \ln((z_i - z_{i-1})!) - m(t_0) \]

In (6) substitute \( m(t_1) = a[1 - e^{-bt_1}] \).

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

\[ \hat{a} = \frac{z_n}{1 - e^{-bt_n}} \]

\[ \hat{b} = \frac{\sum_{i=1}^{n} (z_i - z_{i-1}) \left[ \frac{e^{bt_i} - 1}{e^{bt_i} - 1} \right]}{1 - e^{-bt_n}} \]

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

\[ \hat{a} = \frac{z_n}{1 - (1 + bt_n)e^{-bt_n}} \]

\[ \frac{z_n^2 e^{-bt_n}}{1 - (1 + bt_n)e^{-bt_n}} = \sum_{i=1}^{n} (z_i - z_{i-1}) \left[ \frac{t_i^2 e^{-bt_i} - t_i t_i e^{-bt_i}}{1 - (1 + bt_i)e^{-bt_i}} \right] \]

Solving (9) and (10) numerically, we get estimated values of \( a \) and \( b \). Using the data set of Table II the estimated values
are \( \hat{a} = 27.0553, \hat{b} = 0.0202 \). The graphical representation of actual data and estimated values are shown in Fig. 4.

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
\[
T_R^1 : \text{The minimum value that satisfies (13)} \quad T_R^1 \geq 0.
\]
\[
T_R^2 : \text{The minimum value that satisfies (14)} \quad T_R^2 \geq 0.
\]
\[
T_C : \text{The one that minimizes } C(T), \quad T_C \geq 0.
\]
\[
T^*_R : \text{The optimal solution to } P_1.
\]
\[
T^*_R : \text{The optimal solution to } P_2.
\]

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

Case 1: If \( R_{op}(x/T) \geq R_0 \) then \( T_R^* = T_R^1 = T_C \)

Case 2: If \( R_{op}(x/T) > R_0 \) then let \( T^* \) be the solution to
\[
T^* = \text{the solution to } \lambda(T) > 0.
\]

\[
(a) \quad \text{If } T_C \geq T_R^2 \quad \text{then } T^*_R = T^*_R = T_C.
\]
\[
(b) \quad \text{If } T_C < T_R^2 \quad \text{then } T^*_R = T_C < T^*_R = T_R^2.
\]

Case 3: If \( R_{op}(x/T) < R_0 \) then \( T^*_R = \text{max}(T_C, T_R^1) \) and \( T^*_R = \text{max}(T_C, T_R^2) \)

\[
(a) \quad \text{If } T_C \geq T_R^2 \quad \text{then } T^*_R = T^*_R = T_C.
\]
\[
(b) \quad \text{If } T_C < T_R^2 \quad \text{then } T^*_R = T_C < T^*_R = T_R^2.
\]

**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 \( R_{op}(x/T) < R_{op}(x/T) \)

Case 2: If \( T_0 > T \) then let \( T_1 \) be the solution to
\[
\lambda(T) = \lambda(T_1) > T, \quad \text{we have}
\]
\[
(a) \quad \text{If } T_1 \geq T + x \quad \text{then } R_{op}(x/T) > R_{op}(x/T).
\]
\[
(b) \quad \text{If } T_1 < T + x \quad \text{then}
\]
\[
R_{op}(x/T) > R_{op}(x/T), \quad \text{if } M < 0
\]
\[
R_{op}(x/T) = R_{op}(x/T), \quad \text{if } M = 0
\]
\[
R_{op}(x/T) < R_{op}(x/T), \quad \text{if } M > 0.
\]

where \( M = \lambda(t) - m(T + x) + m(T) \).

The proof is simple and straightforward.

**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 \( T^*_R = T^*_R = \text{max}(T_C, T_0) \).

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

\[
T^*_R = \text{max}(T_C, T_0) \quad \text{and} \quad T^*_R = \text{max}(T_C, T_R^2) \]
The optimal release problem should be formulated as $2P$ rather than $1P$. If the operational reliability is meaningful to the customers, the software will be incorrectly released before it reaches the required reliability level. In fact, the required testing time and the software will be incorrectly estimated. 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]

$$C(T) = C_1 m(T) + C_2 [m(e) - m(T)] + C_3 T$$

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

$$T_C = \frac{1}{\hat{b}} \ln \left( \frac{\hat{a} R_0 (C_1 - C_2)}{C_3} \right) = 560$$

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

$$T_R^2 = \frac{1}{\hat{b}} \ln \left( \frac{\hat{a} x}{\ln(1/R_0)} \right) = 774$$

This case, if $R_{x/R_0}(x/R_0) < R_0$, then $T_R^* = 727$ and $T_R^* = 774$ from Theorem 4.1 of Case (3). If $T_R^* = 727$ testing reliability of the software under this solution is 0.746 which does not satisfy the reliability requirement of 0.8. But $T_R^* = 774$ 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 $\hat{b} = 0.020172$.

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

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

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

$$[1 + b(T_C^2 + x)] e^{-bT_C^2} - [1 + bT_C^2] e^{-bT_C^2} = \ln \left( \frac{R_0}{a} \right)$$

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

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

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

From the above results $R_{x/R_0}(x/R_0) < R_0$ and the mean value function $m(t)$ is S-shaped, from Theorem 4.3 it corresponds to (c) of case 3. Therefore $T_R^1 = 329.611$ and $T_R^* = 372.659$. Hence from Theorem 4.3 of (c) the optimum release time of the problem $P_1$ is $T_R^* = 329.611$. 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 $P_1$ is $T_R^* = 372.659$. If the reliability requirement is used as testing reliability requirement, it will lead to inadequate
software testing time and the required software reliability will not be reached.

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 is 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


