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.


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. Popstojanov 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
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

$$m(t) = 1[p(1-e^{b_1}) + (1-p)(1-e^{b_2})]$$  \hspace{1cm} \text{...(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-e^{b_1}) + (1-p)(1-e^{b_2}))]\]$$

$$= a[p(1-e^{b_1}) + 2(1-p)b_2te^{b_2}]$$  \hspace{1cm} \text{...(2)}$$
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 \( 0 < t_1 < t_2 \cdots < t_n \). Then the log likelihood function (LLF) is given by

\[
LLF = \sum_{i=1}^{n} (y_i - Y_i) \log[\lambda(t_i) - \lambda(t_{i-1})] - \lambda(t_n)
\]

where \( y_i \) is the cumulative number of detected errors. Now the maximum of the LLF is determined by using the following system of equations:

\[
0 = \sum_{i=1}^{n} \frac{\partial}{\partial \phi} \frac{\lambda(t_i) - \lambda(t_{i-1})}{\lambda(t_i)} (y_i - Y_i) - \frac{\partial}{\partial \phi} \lambda(t_n)
\]

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

\[ EC(T) = C_o + C_t \int_0^T e^{-\alpha t} dt + C_w(T) \]  \hspace{1cm} \text{...(10)}

For calculating the maintenance costs during the warranty period i.e. evil), 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

\[ C_w(T) = C_w \int_0^{T_w} \lambda(t)e^{-\alpha t} dt \]  \hspace{1cm} \text{...(11)}

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

\[ C_w(T) = C_w \int_0^{T_w} \lambda(t)e^{-\alpha t} dt \]  \hspace{1cm} \text{...(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

\[ EC(T) = C_o + C_t \int_0^T e^{-\alpha t} dt + C_w \int_0^{T_w} \lambda(t)e^{-\alpha t} dt \]  \hspace{1cm} \text{...(13)}

**For case 2,** \( EC(T) = C_o + C_t \int_0^T e^{-\alpha t} dt + C_w \int_0^{T_w} \lambda(t)e^{-\alpha t} dt \)  \hspace{1cm} \text{...(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

\[ A = pb_1(e^{b_1 T} + \alpha) + 2b_2(1-p)e^{b_2 T} + [T(2b_2 T + \alpha - 1)] \]  \hspace{1cm} \text{...(15)}
Now, we find the optimal testing time $T^*$ by imposing the reliability constraint. The software reliability function $R(x)$ 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

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

...(17)

From equation (1), we have

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

...(18)

Let $R_0$ be the minimum required software reliability ($0 < R_0 \leq 1$). Then the optimal software release problem is stated as
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-OTP4 are tested for \( p = 1 \) and are tabulated in Tables 1-4. The optimal testing times of the respective policies are obtained by varying \( c \) 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 \( c \). Moreover, we notice that for a particular combination of the values of \( c \) and \( T_w \), the optimal testing time in table 1 is greater than that in table 2 which implies that \( T_1^* \) (1* in case 1) > \( T_2^* \) (1* 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

\[
\begin{align*}
\text{Minimize} & \quad EC(T) \\
\text{subject to} & \quad R(x|T) \geq R_0 
\end{align*}
\]

Let \( T_0 \) denote the optimal testing time satisfying the above constraint. Putting \( R(x|T) = R_0 \) in equation (19), we get,

\[
R = \exp [-\{m(T+x)-m(T)\}] 
\]

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 \( T^* = \max\{T_p, T_k\} \).
- **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 \( T^* = T_k \).
- **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 \( T^* = \max\{T_2, T_k\} \).
- **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 \( T^* = T_k \).
- **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-OTP4 are tested for \( p = 1 \) and are tabulated in Tables 1-4. The optimal testing times of the respective policies are obtained by varying \( c \) 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 \( c \). Moreover, we notice that for a particular combination of the values of \( c \) and \( T_w \), the optimal testing time in table 1 is greater than that in table 2 which implies that \( T_1^* \) (1* in case 1) > \( T_2^* \) (1* in case 2) i.e. lesser time is required for testing if the software reliability growth is assumed to occur in the warranty period.
4. \( T^* = \max \{ T_1(T_2), T_g \} = 195.64 \) 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**  
\( a=.001, a=150, b_i=.03, C_w=10 \)

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

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

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

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

<table>
<thead>
<tr>
<th>ct</th>
<th>Tw</th>
<th>5</th>
<th>10</th>
<th>15</th>
<th>20</th>
<th>25</th>
<th>30</th>
<th>35</th>
<th>40</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>195.64</td>
<td>195.64</td>
<td>195.64</td>
<td>195.64</td>
<td>195.64</td>
<td>195.64</td>
<td>195.64</td>
<td>195.64</td>
<td>195.64</td>
</tr>
<tr>
<td>10</td>
<td>195.64</td>
<td>195.64</td>
<td>195.64</td>
<td>195.64</td>
<td>195.64</td>
<td>195.64</td>
<td>195.64</td>
<td>195.64</td>
<td>195.64</td>
</tr>
<tr>
<td>15</td>
<td>195.64</td>
<td>195.64</td>
<td>195.64</td>
<td>195.64</td>
<td>195.64</td>
<td>195.64</td>
<td>195.64</td>
<td>195.64</td>
<td>195.64</td>
</tr>
<tr>
<td>20</td>
<td>195.64</td>
<td>195.64</td>
<td>195.64</td>
<td>195.64</td>
<td>195.64</td>
<td>195.64</td>
<td>195.64</td>
<td>195.64</td>
<td>195.64</td>
</tr>
<tr>
<td>25</td>
<td>195.64</td>
<td>195.64</td>
<td>195.64</td>
<td>195.64</td>
<td>195.64</td>
<td>195.64</td>
<td>195.64</td>
<td>195.64</td>
<td>195.64</td>
</tr>
<tr>
<td>30</td>
<td>195.64</td>
<td>195.64</td>
<td>195.64</td>
<td>195.64</td>
<td>195.64</td>
<td>195.64</td>
<td>195.64</td>
<td>195.64</td>
<td>195.64</td>
</tr>
</tbody>
</table>
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.
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)

\[
C_0=150, a=150, C_t=20, C_w=10, \ p=.2, \\
b_2=.05, b_1=.03, \alpha=0.001, T_w=10
\]

**Figure 2a:** Expected total maintenance costs by varying \(a\) (for case 2)

\[
C_0=150, \alpha=.001, C_t=20, \\
C_w=10, p=.2, b_1=.03, b_2=.05, T_w=10
\]

**Figure 2b:** Expected total maintenance costs by varying \(b_2\) (for case 2)

\[
C_0=150, \alpha=.001, C_t=20, C_w=10, \\
p=.2, a=150, b_1=.03, T_w=10
\]
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 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 $T^*=T_1=10$ 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 $R_o$ to be achieved is 0.99, then $T^*$ comes out to be $T^*=T_2=24$ 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.

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


