Chapter 4

Two-Dimensional Problems in Software Reliability

The work in the area of software reliability done so far studies the effect of one factor like Testing-Time, Testing-Effort or Coverage, etc on fault removal process but in practical situations, several factors such as the running environment, testing strategy and resource allocation affect the process simultaneously. Once these factors are changed during testing phase, it could result in failure intensity function that increases or decreases non-monotonically and the time point corresponding to abrupt fluctuations is called change point.

Change-point is one of the interesting phenomenon observed during software development. Inclusion of change point in software reliability growth modeling enhances the predictive accuracy of the model. Many researchers have incorporated change point in software reliability growth modeling (See chapter 1 section 1.9.7).

Firstly Zhao (1993) incorporated change-point in software and hardware reliability. Huang (2005b) used change-point in software reliability growth modeling with testing effort functions. The imperfect debugging with change-point has been introduced in software reliability growth modeling by Shyur (2003). Kapur, Aggarwal and Kaur (2011a) introduced various testing effort function with change point in software reliability growth modeling. The multiple change-points in software reliability growth modeling for fielded software has been proposed by Kapur, et al. (2011e).

This Chapter is based on the following research papers entitled:


Testing Coverage (TC) is actually a structural testing technique in which the software performance is judged with respect to specification of the source code and the extent or the degree to which software is executed by the test cases. TC can help software developers to evaluate the quality of the tested software and determine how much additional effort is needed to improve the reliability of the software besides providing customers with a quantitative confidence criterion while planning to use a software product. Hence, safety critical system has a high coverage objective. The basic testing coverage measures are proposed by Inoue and Yamada (2004); Musa, Iannino, Okumoto (1987):

A testing coverage based SRGM was proposed by Malaiya et al. (2002). Inoue and Yamada (2004) developed SRGM to describe a time-dependent behavior of a testing-coverage attainment process with the testing-skill of test-case designer.

But all the models stated above in the chapter are either time dependent or are based on testing resource function that depends on time. These models do not take into account the simultaneous effect of time and resource or concurrent effect of time and coverage is not considered. Therefore, all such models can be termed as one-dimensional SRGM. So far in the thesis, we have work on such one-dimensional SRGM only. However, in order to capture the mutual effect of testing time and resources or simultaneous effect of testing time and coverage, two-dimensional SRGM is needed. In this chapter, we propose various two-dimensional SRGM.

In addition, multi release SRGM has been discussed in the chapter. The proposed model is also based on framework as described in chapter3, section2, i.e. the dependency on just previous release of the newer version of the software and effect of reported bugs in the FRP.

Importance of reliable software has escalated many folds. Notwithstanding its unassailable value, there is still no way to test whether it is completely fault free or not. It is owing to the prevailing paradox that software user’s requirements are conflicting with the developers. Software users demand faster deliveries, cheaper software and quality product whereas software developers aim at minimizing their development cost, maximizing the profit margins and meeting the competitive
requirements. The resulting situation calls for tradeoffs between conflicting objectives prevailing between software user’s requirements with the developers in turn driving the management to determine when to stop testing and release the software system to the user. Such a problem is known as “Software Release Time Optimization Problem”. If the release of the software is unduly delayed, the manufacturer (software developer) may suffer in terms of penalties and revenue loss, while a premature release may cost heavily in terms of fixes (removals) to be done after release, which consequently might harm the manufacturer’s reputation.

Later in the chapter, we propose a software cost model for multi up-gradation SRGM under two-dimensional framework. In order to determine optimal time and effort for the release of a new version of the software, multi attribute utility theory has been used. This technique attempts to identify relevant objectives for any given decision making problem, where a decision is typified by multiple objectives. It can be difficult to quantitatively compare these objectives one against another. In order to provide insight into this problem, a utility function is assessed for each of the relevant objectives. This allows for an appropriate multiple-objective utility function that is used to identify trade-offs and compare the various objectives in a consistent manner. The basic of utility theory and its underlying quantities axioms were initially established by Neumann and Morgenstern, (1947)

The chapter is divided in two sections. Section 4.1 develops a unified framework using hazard rate function to develop various two dimensional SRGMs using different failure distributions with change point. The models developed have been validated on real datasets. In Section 4.2, a two-dimensional multi release SRGM has been discussed. Numerical illustrations are given to justify the release time problems and lastly sensitivity analysis is given for few parameter based on time and effort, separately.

In this section, a general scheme is created to develop various existing and new software reliability growth models incorporating the effect of change point in two-dimensional modeling framework. The framework is developed using hazard rate function approach. The review of Change point in one-dimensional framework have been discuss in details in chapter1 section 1.9.7.

4.1.1. Assumptions:

The proposed modeling scheme is based upon the following basic assumptions:

1. Failure/ fault removal phenomenon is modeled by NHPP.
2. Software is subject to failures during execution caused by faults remaining in the software.
3. Failure rate is equally affected by all the faults remaining in the software.
4. Fault detection / removal rate may change at any time moment, known as change point.
5. During the fault detection /removal, no new fault is introduced into the software.
6. The number of faults in the beginning of the testing phase is finite.

4.1.2. Notations

\( F(\cdot) \), \( f(\cdot) \) : Probability distribution function and Probability density function for fault removal process.

\( \alpha \) : Initial number of faults in software.

\( s, u \) : Testing –time and Testing –Coverage.

\( \alpha \) : Time Elasticity of fault removal process and is constant.

\( m(s,u) \) : Cumulative number of faults removed by time \( s \) and with coverage \( u \)

\( \psi_{s,u} \) : \( s^\alpha u^{1-\alpha} \quad 0 \leq \alpha \leq 1. \)

\( \psi_\tau \) : Change Point and it is defined as \( s_\tau^\alpha u_\tau^{1-\alpha} \quad 0 \leq \alpha \leq 1. \)
4.1.3. Modeling of the Two-Dimensional SRGM

Here in this section, we formulate probability distribution function based software reliability growth models incorporating change-point. The proposed model incorporates testing-time and testing-coverage factors, simultaneously. For representation of the concurrent effect of these factors on the software reliability growth process, we have used the Cobb-Douglas function (Inoue and Yamada, 2008; Inoue et al., 2010) as

\[ \psi_{s,u} = s^\alpha u^{1-\alpha} \quad 0 \leq \alpha \leq 1 \]

(4.1)

Where \( s,u \) represent testing-time and coverage, respectively and \( \alpha \) is the degree of the impact to the software reliability growth process. Inoue and Yamada (2008); Inoue, Fukuma et al. (2010) have proposed a two dimensional NHPP to characterize stochastic behavior of two-dimensional software reliability growth process (see chapter 1 section 10).

Let \( \{N(s,u), s \geq 0, u \geq 0\} \) be a two-dimensional stochastic process which represent the cumulative number of software failures by testing-time \( s \) and testing-coverage \( u \). A two-dimensional NHPP with a mean value function \( m(s,u) \) is given by:

\[ \Pr(N(s,u) = n) = \frac{(m(s,u))^n}{n!} \exp(-m(s,u)) ; n = 0,1,2… \]

(4.2)
Under above assumptions the differential equation representing the rate of change of cumulative number of faults removed based on testing time and coverage is given as:

\[
m'(\psi_{s,u}) = b(\psi_{s,u})(a - m(\psi_{s,u}))
\]  
(4.3)

Now, Equation (4.3) may be written as:

\[
\frac{dm(\psi_{s,u})}{d\psi_{s,u}} = \frac{f(\psi_{s,u})}{1 - F(\psi_{s,u})} \left[ a - m(\psi_{s,u}) \right]
\]  
(4.4)

Here \( h(\psi_{s,u}) \) represents hazard rate function corresponding two-dimensional SRGM.

### 4.1.4. Modeling of the Two-Dimensional SRGM with Change-Point

Our modeling framework with change-point is based on a unified framework proposed by Kapur, Pham, Anand and Yadav (2011). Let us assumed that the hazard rate \( h(\psi_{s,u}) \) i.e. fault detection/correction rate per remaining fault of the software, before and after some specified time point does not have same value. This change may be caused by shift in testing strategy, effort, testing technique etc. Let \( h(\psi_{s,u}) \) may be written as

\[
h(\psi_{s,u}) = \begin{cases} 
\frac{f_1(\psi_{s,u})}{1 - F_1(\psi_{s,u})} & \psi_{s,u} \leq \psi_e \\
\frac{f_2(\psi_{s,u})}{1 - F_2(\psi_{s,u})} & \psi_{s,u} > \psi_e
\end{cases}
\]  
(4.5)

Further, after incorporating the change-point concept on the model, For \( \psi_{s,u} \leq \psi_e \), Equation (4.4) becomes:

\[
\frac{dm(\psi_{s,u})}{d\psi_{s,u}} = \frac{f_1(\psi_{s,u})}{1 - F_1(\psi_{s,u})} \left[ a - m(\psi_{s,u}) \right]
\]  
(4.6)
By solving the above equation with initial condition \( m(\psi_{0.0} = 0) = 0 \), we get:

\[
m(\psi_{s,u}) = a \ F_1(\psi_{s,u})
\]  \hspace{1cm} (4.7)

Also for \( \psi_{s,u} > \psi_{\tau} \) the equation (4.4) may be rewritten as:

\[
\frac{dm(\psi_{s,u})}{d\psi_{s,u}} = \frac{f_2(\psi_{s,u})}{1 - F_2(\psi_{s,u})} \left[ a - m(\psi_{s,u}) \right]
\]  \hspace{1cm} (4.8)

Solving above equation with initial condition at \( \psi_{s,u} = \psi_{\tau} \) \( m(\psi_{s,u}) = m(\psi_{\tau}) \), we get

\[
m(\psi_{s,u}) = a \left[ 1 - \frac{(1 - F_1(\psi_{\tau}))(1 - F_2(\psi_{s,u}))}{1 - F_2(\psi_{\tau})} \right]
\]  \hspace{1cm} (4.9)

i.e.

\[
m(\psi_{s,u}) = \begin{cases}  
a \ F_1(\psi_{s,u}) & ;\psi_{s,u} \leq \psi_{\tau} \\
a \left[ 1 - \frac{(1 - F_1(\psi_{\tau}))(1 - F_2(\psi_{s,u}))}{1 - F_2(\psi_{\tau})} \right] & ;\psi_{s,u} > \psi_{\tau}
\end{cases}
\]  \hspace{1cm} (4.10)

### 4.1.5. Derivation of New and Existing Models

Now we derive several SRGMs which incorporate change-point concept. By using two-dimensional framework given in eq.(4.10) we have derived GO-model (Goel and Okumoto, 1979), Yamaha model (Yamada et al., 1983), Kapur(Kapur et al., 1999) model, gamma and Weibull distribution functions (Pham, 2006, Rausand and Hoyland, 2004).

#### 4.1.5.1. GO-Model with Change Point in Two-Dimensional, SRGM 1

In the first SRGM, we applying change point on (Goel and Okumoto, 1979)in two dimensional environment. Let
\[
\begin{align*}
F_1(\psi_{s,u}) &= 1 - \exp\left(-b_1\psi_{s,u}\right) \quad \text{for } \psi_{s,u} \leq \psi_0 \\
F_2(\psi_{s,u}) &= 1 - \exp\left(-b_2\psi_{s,u}\right) \quad \text{for } \psi_{s,u} > \psi_\tau
\end{align*}
\] (4.11)

Substituting \(F_1(\psi_{s,u})\) and \(F_2(\psi_{s,u})\) into Equation (4.10), we get:

\[
m(\psi_{s,u}) = \begin{cases} 
\left[1 - \exp\left(-b_1\psi_{s,u}\right)\right] & ; \psi_{s,u} \leq \psi_\tau \\
\left[1 - \exp\left(-b_2\psi_{s,u} - b_2\left(\psi_{s,u} - \psi_\tau\right)\right)\right] & ; \psi_{s,u} > \psi_\tau 
\end{cases}
\] (4.12)

In this model, if we consider \(\alpha = 1\) we can get change point based one-dimensional software reliability growth model (Shyur, 2003).

### 4.1.5.2. Yamada-Model with Change Point in Two-Dimensional, SRGM 2

Let \(F(\psi_{s,u})\) be a two-stage Erlangen distribution function (Yamada s-shaped model (Yamada et al., 1983)) i.e.,

\[
\begin{align*}
F_1(\psi_{s,u}) &= 1 - \left(1 + b_1\psi_{s,u}\right)\exp\left(-b_1\psi_{s,u}\right) \quad \text{for } \psi_{s,u} \leq \psi_\tau \\
F_2(\psi_{s,u}) &= 1 - \left(1 + b_1\psi_{s,u}\right)\exp\left(-b_2\psi_{s,u}\right) \quad \text{for } \psi_{s,u} > \psi_\tau
\end{align*}
\] (4.13)

Substituting \(F_1(\lambda_{s,u})\) and \(F_2(\lambda_{s,u})\) into Equation (4.10), we get:

\[
m(\psi_{s,u}) = \begin{cases} 
\left[1 - \left(1 + b_1\psi_{s,u}\right)\exp\left(-b_1\psi_{s,u}\right)\right] & ; \psi_{s,u} \leq \psi_\tau \\
\left[1 - \left(1 + b_1\psi_\tau\right)\left(1 + b_2\psi_{s,u}\right)\right] \times \exp\left(-b_2\psi_\tau - b_2\left(\psi_{s,u} - \psi_\tau\right)\right) & ; \psi_{s,u} > \psi_\tau 
\end{cases}
\] (4.14)

### 4.1.5.3. Kapur-Garg-Model with Change Point in Two-Dimensional, SRGM 3

Let \(F(\psi_{s,u})\) be given by:
\[
F_1(\psi_{s,u}) = \frac{1-\exp(-b_1\psi_{s,u})}{1 + \beta_1 \exp(-b_1\psi_{s,u})} \quad \text{for } \psi_{s,u} \leq \psi_\tau
\]
\[
F_2(\psi_{s,u}) = \frac{1-\exp(-b_2\psi_{s,u})}{1 + \beta_2 \exp(-b_2\psi_{s,u})} \quad \text{for } \psi_{s,u} > \psi_\tau
\]

Then the corresponding mean value function is given by:

\[
m(\psi_{s,u}) = \begin{cases} 
1 - \left( \frac{1 + \beta_1}{1 + \beta_1 \exp(-b_1\psi_{s,u})} \right) \exp(-b_1\psi_{s,u}) & ; \psi_{s,u} \leq \psi_\tau \\
1 - \left( \frac{1 + \beta_1}{1 + \beta_1 \exp(-b_1\psi_{\tau})} \right) \left( \frac{1 + \beta_2 \exp(-b_2\psi_{s,u})}{1 + \beta_2 \exp(-b_2\psi_{s,u})} \right) \exp(-b_1\psi_{\tau} - b_2(\psi_{s,u} - \psi_\tau)) & ; \psi_{s,u} > \psi_\tau 
\end{cases}
\]

(4.15)

4.1.5.4. Gamma- distribution with Change Point in Two-Dimensional, SRGM 4

Let \( F(\psi_{s,u}) \) be Gamma distribution function (Pham, 2006) and let

\[
F_1(\psi_{s,u}) = \Gamma(\psi_{s,u}, \xi_1, \delta_1) \quad \text{for } \psi_{s,u} \leq \psi_\tau \quad \text{and}
\]
\[
F_2(\psi_{s,u}) = \Gamma(\psi_{s,u}, \xi_2, \delta_2) \quad \text{for } \psi_{s,u} > \psi_\tau
\]

Then the corresponding mean value function is given by:

\[
m(\psi_{s,u}) = \begin{cases} 
\frac{a \cdot \Gamma(\psi_{s,u}, \xi_1, \delta_1)}{1 - \Gamma(\psi_{\tau}, \xi_1, \delta_1) \Gamma(\psi_{s,u}, \xi_2, \delta_2) (1 - \Gamma(\psi_\tau, \xi_2, \delta_2))} & ; \psi_{s,u} \leq \psi_\tau \\
\frac{a \cdot \Gamma(\psi_{s,u}, \xi_2, \delta_2)}{1 - \Gamma(\psi_{\tau}, \xi_2, \delta_2) \Gamma(\psi_{s,u}, \xi_1, \delta_1) (1 - \Gamma(\psi_\tau, \xi_1, \delta_1))} & ; \psi_{s,u} > \psi_\tau 
\end{cases}
\]

(4.17)

4.1.5.5. Weibull- distribution with Change Point in Two-Dimensional, SRGM 5

In last SRGM, we have applied two-dimensional and change-point concepts on Weibull distribution function.
Let \( F_1(\psi_{s,u}) = 1 - \exp\left(-b_1\psi_{s,u}^{k_1}\right) \quad \text{for} \quad \psi_{s,u} \leq \psi_{\tau} \) and
\[
F_2(\psi_{s,u}) = 1 - \exp\left(-b_2\psi_{s,u}^{k_2}\right) \quad \text{for} \quad \psi_{s,u} > \psi_{\tau} \quad \quad (4.19)
\]

Substituting \( F_1(\psi_{s,u}) \) and \( F_2(\psi_{s,u}) \) into Equation (4.10), we get:
\[
m\left(\psi_{s,u}\right) = \begin{cases} 
1 - \exp\left(-b_1\psi_{s,u}^{k_1}\right) & ; \psi_{s,u} \leq \psi_{\tau} \\
1 - \exp\left(-b_1\psi_{s,u}^{k_1} - b_2\left(\psi_{s,u}^{k_2} - \psi_{\tau}^{k_2}\right)\right) & ; \psi_{s,u} > \psi_{\tau}
\end{cases} \quad \quad (4.20)
\]

### 4.1.6. Data Sets and Model Validation

The data sets used for the two dimensional model are coverage data sets. These data sets are described as follows:

- **Data set 1 (DS-1):** Coverage Data set with 796 test cases and 9 cumulative numbers of faults removed with block coverage as 95.99% (Malaiya et al., 2002). The Change point for DS-1 is obtained at \( s = 44, u = 87 \) (Figure 4.2).

- **Data set 2 (DS-2):** Consists of 9 cumulative faults removal with 1196 test case covering 95.97% of the block coverage (Malaiya et al., 2002). The Change point for DS-2 is obtained at \( s = 20, u = 70.5 \) (Figure 4.3).

The Method of Least Squares is used for estimation of parameters by SPSS and SAS(SAS, 2004). The performance analysis of proposed model is measured by the five common goodness of fit criteria as MSE, BIAS, MRSPE, \( R^2 \) and Variation(Kapur et al., 2011, Pham, 2006).
4.1.7. Parameter Estimation and Goodness of Fit Criteria

The parameters of all releases are estimated and the related mean value functions are obtained. Table 4.1(a), Table 4.1(b) contains the estimated value of parameters of models related to DS1 and Table 4.2(a), Table 4.2(b) contain the estimated value of
parameters of models related to DS2. In addition, goodness of fit criteria of models are given in Table 4.1(c) and Table 4.2(c).

**Table 4.1(a): Model Parameter Estimation Results (DS-1)**

<table>
<thead>
<tr>
<th>Models</th>
<th>$a$</th>
<th>$b_1$</th>
<th>$b_2$</th>
<th>$\beta_1$</th>
<th>$\beta_2$</th>
<th>$\alpha$</th>
</tr>
</thead>
<tbody>
<tr>
<td>SRGM-1</td>
<td>9.0943</td>
<td>0.2132</td>
<td>0.2536</td>
<td>-</td>
<td>-</td>
<td>0.5087</td>
</tr>
<tr>
<td>SRGM-2</td>
<td>9.110</td>
<td>0.03639</td>
<td>0.5296</td>
<td>-</td>
<td>-</td>
<td>0.274</td>
</tr>
<tr>
<td>SRGM-3</td>
<td>10</td>
<td>0.0318</td>
<td>0.022</td>
<td>1.0284</td>
<td>0.001</td>
<td>0.5233</td>
</tr>
</tbody>
</table>

**Table 4.1(b): Model Parameter Estimation Results (DS-1)**

<table>
<thead>
<tr>
<th>Models</th>
<th>$a$</th>
<th>$b_1/\xi_1$</th>
<th>$b_2/\xi_2$</th>
<th>$\delta_1/k_1$</th>
<th>$\delta_2/k_2$</th>
<th>$\alpha$</th>
</tr>
</thead>
<tbody>
<tr>
<td>SRGM-4</td>
<td>9.099</td>
<td>0.5244</td>
<td>0.0017</td>
<td>0.0096</td>
<td>0.0138</td>
<td>0.833</td>
</tr>
<tr>
<td>SRGM-5</td>
<td>9.14</td>
<td>0.1185</td>
<td>0.5025</td>
<td>0.6252</td>
<td>0.5430</td>
<td>0.8632</td>
</tr>
</tbody>
</table>

**Table 4.1(c): Model Comparison Results (DS-1)**

<table>
<thead>
<tr>
<th>Models</th>
<th>$R^2$</th>
<th>MSE</th>
<th>Bias</th>
<th>Variation</th>
<th>RMSPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>SRGM-1</td>
<td>0.991</td>
<td>0.0661</td>
<td>0.0033</td>
<td>0.2711</td>
<td>0.2712</td>
</tr>
<tr>
<td>SRGM-2</td>
<td>0.990</td>
<td>0.0770</td>
<td>-0.003</td>
<td>0.2926</td>
<td>0.2926</td>
</tr>
<tr>
<td>SRGM-3</td>
<td>0.991</td>
<td>0.6847</td>
<td>-0.001</td>
<td>0.2756</td>
<td>0.2756</td>
</tr>
<tr>
<td>SRGM-4</td>
<td>0.994</td>
<td>0.4149</td>
<td>-0.001</td>
<td>0.2147</td>
<td>0.2147</td>
</tr>
<tr>
<td>SRGM-5</td>
<td>0.996</td>
<td>0.03</td>
<td>0.0012</td>
<td>0.1820</td>
<td>0.1825</td>
</tr>
</tbody>
</table>
Table 4.2(a): Model Parameter Estimation Results (DS-2)

<table>
<thead>
<tr>
<th>Models</th>
<th>$a$</th>
<th>$b_1$</th>
<th>$b_2$</th>
<th>$\beta_1$</th>
<th>$\beta_2$</th>
<th>$\alpha$</th>
</tr>
</thead>
<tbody>
<tr>
<td>SRGM-1</td>
<td>9.2</td>
<td>0.0088</td>
<td>0.0141</td>
<td>-</td>
<td>-</td>
<td>0.4399</td>
</tr>
<tr>
<td>SRGM-2</td>
<td>9.22</td>
<td>0.0227</td>
<td>0.0277</td>
<td>-</td>
<td>-</td>
<td>0.2896</td>
</tr>
<tr>
<td>SRGM-3</td>
<td>10</td>
<td>0.01647</td>
<td>0.0229</td>
<td>1.5767</td>
<td>2.0745</td>
<td>0.5233</td>
</tr>
</tbody>
</table>

Table 4.2(b): Model Parameter Estimation Results (DS-2)

<table>
<thead>
<tr>
<th>Models</th>
<th>$a$</th>
<th>$b_1/\xi_1$</th>
<th>$b_2/\xi_2$</th>
<th>$\delta_1/k_1$</th>
<th>$\delta_2/k_2$</th>
<th>$\alpha$</th>
</tr>
</thead>
<tbody>
<tr>
<td>SRGM-4</td>
<td>9.63</td>
<td>1.3964</td>
<td>0.001</td>
<td>0.0042</td>
<td>0.0160</td>
<td>0.4766</td>
</tr>
<tr>
<td>SRGM-5</td>
<td>9.23</td>
<td>0.0042</td>
<td>0.0085</td>
<td>1.160</td>
<td>1.147</td>
<td>0.3487</td>
</tr>
</tbody>
</table>

Table 4.2(c): Model Comparison Results (DS-2)

<table>
<thead>
<tr>
<th>Models</th>
<th>$R^2$</th>
<th>MSE</th>
<th>Bias</th>
<th>Variation</th>
<th>RMSPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>SRGM-1</td>
<td>0.97</td>
<td>0.2380</td>
<td>-0.0564</td>
<td>0.5035</td>
<td>0.5066</td>
</tr>
<tr>
<td>SRGM-2</td>
<td>0.973</td>
<td>0.2113</td>
<td>-0.0228</td>
<td>0.4764</td>
<td>0.477</td>
</tr>
<tr>
<td>SRGM-3</td>
<td>0.973</td>
<td>0.2094</td>
<td>-0.0028</td>
<td>0.474948</td>
<td>0.47495</td>
</tr>
<tr>
<td>SRGM-4</td>
<td>0.975</td>
<td>0.1957</td>
<td>-0.0135</td>
<td>0.4589</td>
<td>0.4591</td>
</tr>
<tr>
<td>SRGM-5</td>
<td>0.972</td>
<td>0.2213</td>
<td>-0.0273</td>
<td>0.4873</td>
<td>0.48815</td>
</tr>
</tbody>
</table>
4.2. Modeling Two-Dimensional Software Multi Up-gradation and Related Release Problem

In the previous section, we have proposed two-dimensional NHPP based SRGM to represent the testing phase of SDLC. Now, we extend the concept of two-dimensional modeling to represent testing as well as operational phase of SDLC. Here we have used two-dimensional Kapur & Garg (1999) model for FRP of testing phase and two-dimensional Weibull model for FRP based on bugs reported by the user from the operational phase.

In addition, we address the problem faced by the most of the software managers, namely, time to stop testing or time of releasing software. This is a problem of decision-making under uncertainty and involves a tradeoff between reliability, cost and delivery time. Here we propose a software cost model based on Multi Up-gradation Software Reliability Growth Model (MUSRGM) in the two-dimensional environment. Multi Attribute Utility Theory (MAUT) has been used in order to determine optimal time and effort for the release of a new version of the software. A numerical example have been done to illustrate model validation and sensitivity analysis has been introduced to help management know how robust the decision is.

4.2.1. Assumptions:

The basic assumptions of the model are as follows:

1) The fault removal process for each release follows non-homogeneous poison process (NHPP).
2) The number of faults detected at any time is proportional to the remaining number of faults in the software.
3) The undetected faults of previous release are divided in two parts and a different growth curve is used to represent the FRP of each part.
4) The number of faults at the beginning of the testing phase is finite.
5) From fault detection point of view, all faults are mutually independent.
6) Each time a failure occurs, the causal fault is immediately fixed, and no other faults are introduced.

7) To capture the combined effect of testing time and effort, we make use of Cobb-Douglas production function.

### 4.2.2. Notations used:

The following notations have been used in the paper.

- \( s,u \) : Testing –time and Testing-Effort.
- \( \alpha \) : Time elasticity of fault removal process and is constant.
- \( m(s,u) \) : Expected number of faults removed by time \( s \) and testing effort \( u \).
- \( \psi_{s,u}^{s,u} \) : \( s^\alpha u^{1-\alpha} \), \( 0 \leq \alpha \leq 1 \).
- \( F_{i}^{test} (\psi_{s,u}^{s,u}) \) : Probability distribution functions for testing phase and operational phase, respectively.
- \( F_{i}^{op} (\psi_{s,u}^{s,u}) \) : Probability distribution functions for testing phase and operational phase, respectively.
- \( \psi_{i} \) : \( s_i^\alpha u_i^{1-\alpha} \), \( 0 \leq \alpha_i \leq 1 \).
- \( a_i \) : Initial fault content for \( i^{th} \) release, \( i = 1..4 \).
- \( a \) : Total fault content in the software.
- \( b(\psi_{s,u}^{s,u}) \) : Time-effort dependent fault detection rate function.
- \( \beta_i \) : Learning factor in Logistic model for \( i^{th} \) release, \( i = 1..4 \).
- \( b_{test} \) : Fault detection rate during testing phase.
- \( b_{op} \) : Fault detection rate during operational phase.
- \( \lambda_i \) : Proportion of faults* of \( i^{th} \) release during testing phase of \((i+1)^{th}\) release.
- \( 1-\lambda_i \) : Proportion of fault* of \( i^{th} \) release, removed on the basis of reported bugs.

[*] fault are undetected fault from previous release.
4.2.3. Modeling of the Two-Dimensional SRGM

Here in the section, we consider a SRGM, which incorporates testing-time and testing-effort as the factors. For representation of the effect of these factors on the Fault Removal Process (FRP) simultaneously, we use the Cobb-Douglas production function (Inoue et al., 2010). Modeling of the two-dimensional SRGM are given in chapter 4 section 4.1.3.

In this section $s,u$ represent testing-time and testing-effort, respectively and $\alpha$ is the degree of the impact of testing time to the FRP. By this technique, we are able to consider the effect of several factors simultaneously. The impact of each factor on FRP depends on the value of elasticity parameter $\alpha$.

4.2.4. Multi Release SRGM in Two-Dimensional Framework

4.2.4.1. Logistic model for testing phase

Logistic model is one of the most popular models in software reliability area and can fit on many types of failure data of testing phase because of its flexible nature (Kapur et al., 2011, Pham, 2006, Singh et al., 2011). The differential equation of this model for two-dimensional framework is given by:

$$\frac{dm(y_{s,u})}{dy_{s,u}} = b(y_{s,u})(a - m(y_{s,u})) \quad (4.21)$$

where

$$b(y_{s,u}) = \frac{b_{test}}{1 + \beta e^{-\alpha y_{s,u}}} \quad (4.22)$$

Solving, the above differential equation under the boundary condition $m(y_{s,u} = 0) = 0$, we get:
During debugging process, the fault removal rate is assumed to be time dependent. The reason for this assumption is to incorporate the effect of learning on the removal personnel. With each fault removal, insight is gained into the nature of faults present and is described by Eq. (4.22) called the logistic function. Also note that \( b(\psi_{s,u}) \) increases monotonically and

\[ \lim_{\psi_{s,u} \to 0} b(\psi_{s,u}) = b_{\text{test}}. \]

### 4.2.4.2. The Weibull Model for Operational Phase

The Weibull distribution is one of the most commonly used distributions in reliability theory. It is commonly used to model time to fail, time to repair and material strength. Here in this paper we use Weibull distribution function for modeling removal process during the Operational phase. The shape parameter \( k \) is what gives the Weibull distribution its flexibility. By changing the value of the shape parameter, the Weibull distribution can model a wide variety of data. If \( k = 1 \), Weibull distribution is identical to the exponential distribution, if \( k = 2 \), Weibull distribution is identical to the Rayleigh distribution. The scale parameter \( b_{\text{op}} \) determines the range of the distribution. The Weibull probability distribution and density function are given as:

\[
F_{\text{op}}(\psi_{s,u}) = (1 - e^{-b_{\text{op}}(\psi_{s,u})^k}), \quad (4.25)
\]

\[
f_{\text{op}}(\psi_{s,u}) = k b_{\text{op}}(\psi_{s,u})^{k-1} e^{-b_{\text{op}}(\psi_{s,u})^k}; k > 0, b_{\text{op}} > 0, \psi_{s,u} \geq 0.
\]

The Weibull distribution function is flexible that can empirically fit many types of failure data and has parameters that can account for different increasing and
decreasing trends in the failure rate. It reflects initial increase and eventual decrease in fault/failure occurrences, which can vary across multiple releases (Li et al., 2004). (For more details, see chapter 3 section 3.2)

### 4.2.4.3. Multiple Release Model Under Two-Dimensional Environment

In general, several factors affect the fault removal phenomenon. Researchers have discussed FRP based on the effect of only one factor like testing time, testing coverage or testing effort, etc for modeling SRGMs. In this section, we consider simultaneous effect of testing time and testing effort on the FRP of multi release SRGM. Also the proposed model is based on the assumption that the overall fault removal of the new release depends on the reported faults from the previous release of the software and on the faults generated due to adding some new functionalities (addons/up-gradations) to the existing system.

In practice, it is important to know that how many faults exist in the software at any time, so that different testing strategy and testing effort can be applied to remove those faults. Faults removed can be categorized with respect to source of occurrence of faults/failure. Some failures happen during testing phase which testing team removes immediately, while few others appear during operational phase of the previous version and the testing team removes these faults as reported from field.

Let us consider that testing begins at $\psi_{s,u} = 0$ and the first release of software take place at $\psi_{s,u} = \psi_1$. The mathematical equation of these finite fault count model is given as:

$$m_1(\psi_{s,u}) = a_1 F_1(\psi_{s,u}), \quad 0 < \psi_{s,u} \leq \psi_1$$

(4.26)

Where, $F_i(\psi_{s,u}) = \frac{1 - \exp\left(-b_{test} \cdot \psi_{s,u}\right)}{1 + \beta_1 \exp\left(-b_{test} \cdot \psi_{s,u}\right)}$, and $\psi_{s,u} = s^\alpha u^{1-\alpha}$

It should be noted that during the first release of the software (in the absence of any operational phase), FRP is associated only with the bugs encountered during initial testing phase.
The proposed model distinguishes between removal process related to faults of the new code and undetected faults of previous release. During testing, it is quite possible that some of faults of old code is removed directly by testing team (without using any bug reports of operational phase of the previous release of software) and some others are removed based on bugs reported during operational phase. In addition due to parallel operations, several people may report the same fault/failure which we call duplicates. Duplicates are not included while counting unique faults. Based on above framework, we can write the following mathematical equation for second release:

\[
m_2(\psi_{s,u}) = a_2F_{test}^2(\psi_{s,u} - \psi_1) + \lambda_1(a_1 - m_1(\psi_1))F_{test}^2(\psi_{s,u} - \psi_1) \\
+ (1 - \lambda_1)(a_1 - m_1(\psi_1))F_{op}^1(\psi_{s,u} - \psi_1) \\
= [a_2 + \lambda_1(a_1 - m_1(\psi_1))]F_{test}^2(\psi_{s,u} - \psi_1) \\
+ [(1 - \lambda_1)(a_1 - m_1(\psi_1))]F_{op}^1(\psi_{s,u} - \psi_1)
\]

(4.27)

Where, \( \psi_1 < \psi_{s,u} \leq \psi_2 \) and \( \psi_{s,u} = s^{\alpha_2}u^{1-\alpha_2} \),

\[
F_{op}^1(\psi_{s,u}) = (1 - e^{-b_2^s(\psi_{s,u})^{\beta_1}}),
\]

\[
F_2(\psi_{s,u}) = \frac{1 - \exp(-b_{test}^s(\psi_{s,u}))}{1 + \beta_2 \exp(-b_{test}^s(\psi_{s,u}))}
\]

(4.28)

At this stage, faults generated due to enhancement of the features are removed with \( F_{test}^2(\psi_{s,u}) \) and \( \lambda_1(a_1 - m_1(\psi_1)) \) represents some of undetected faults of the previous release which interact with new portion of code and are removed/detected by testing team of second release (without using any bug reported during operational phase of the previous release) i.e. \( F_{test}^2(\psi_{s,u} - \psi_1) \). Also the term \( (1 - \lambda_1)(a_1 - m_1(\psi_1))F_{op}^1(\psi_{s,u}) \) represent the remaining undetected faults related to previous release that are removed based on bugs reported from the operational phase.

Same situation will happen in the next release of software as described in Eq. (4.27). The mathematical equations for \( i^{th} \) release is given as:
$$m_i(\psi_{s,u}) = [a_i + \hat{\lambda}_{i-1}(a_{i-1} - m_{i-1}(\psi_{i-1} - \psi_{i-2}))] F_{i}^{\text{est}} (\psi_{s,u} - \psi_{i-1}) +$$
$$[(1 - \hat{\lambda}_{i-1})(a_{i-1} - m_{i-1}(\psi_{i-1} - \psi_{i-2}))] F_{i-1}^{\text{op}} (\psi_{s,u} - \psi_{i-1})$$
(4.29)

Where \(\psi_{i-1} < \psi_{s,u} \leq \psi_{i}\) and \(F_{i}^{\text{est}} (\psi_{s,u} - \psi_{i-1})\), \(F_{i-1}^{\text{op}} (\psi_{s,u} - \psi_{i-1})\) are Logistic and Weibull distribution, respectively. For simplification, we may rewrite Eq. (4.29) as:

$$m_i(\psi_{s,u}) = m_i^{\text{est}} (\psi_{s,u}) + m_i^{\text{op}} (\psi_{s,u})$$
(4.30)

### 4.2.5. Data set and model validation

Validity of the proposed models have been tested on Tandem computer (Wood, 1996, Pham, 2006) four release data set as pointed in chapter 2 section 2.1.5.

The Performance analysis of proposed model is measured by the five common criteria as MSE, BIAS, MRSPE, R2 and Variation(Pham, 2006, Kapur et al., 2011).

### 4.2.6. Parameter Estimation and Goodness of Fit and Data Analysis

The parameters of all releases are estimated and the related mean value functions are obtained. Estimated values of parameters for model in each release are given in Table 4.3(a).

In addition, Table 4.3(b) shows the goodness of fit criteria values related to proposed model. The proposed model gives a very good fit as exhibited by the values of various comparison criteria in Table 4.3(b). In addition, Figure 4.5 shows the estimated values of the number of faults removed for releases 1 to 4, separately.

**Table 4.3(a):** Parameter Estimates
Table.4.3(b): Goodness of fit Criteria

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Releases</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td>$R^2$</td>
<td>0.990</td>
</tr>
<tr>
<td>Bias</td>
<td>0.61707</td>
</tr>
<tr>
<td>MSE</td>
<td>8.1697</td>
</tr>
<tr>
<td>Variation</td>
<td>2.9418</td>
</tr>
<tr>
<td>RMSPE</td>
<td>3.0058</td>
</tr>
</tbody>
</table>

Also the basis of estimation result given in Table 4.3(a), it may be noted that during initial releases there are less number of reported bugs. It may be attributed to less number of users initially Where as the reported bugs from operational phase increases with time but with different rate for all four releases. Figure 4.4 represents the estimation results of the developed model for bug report.
Figure 4.4: the rate of growth of reported bug from operational field.

Table 4.4 gives the value of MSE for two-dimensional multi release SRGM with operational phase and compared with MSE of model in one-dimension in same structure as well as MSE of model without operational phase. (see Table 4.4)

Table 4.4: Comparison of MSE with and without consideration of Operational phase.

<table>
<thead>
<tr>
<th>Comparison</th>
<th>Releases</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td>MSE with Operational phase in two dimension</td>
<td>8.1697</td>
</tr>
<tr>
<td>MSE with Operational phase in one dimension</td>
<td>8.9792</td>
</tr>
<tr>
<td>MSE without Operational phase</td>
<td>8.9792</td>
</tr>
</tbody>
</table>
Figure 4.5: Estimated number of fault removal for Release 1 to 4

4.2.7. Optimal Release Planning of Software

A major concern in software development is the cost. Pham (1996); Kapur, Pham, Gupta & Jha (2011); Gokhale (2006), etc have investigated software release policies to minimize development cost while satisfying a reliability objective. Boland and Chuiv (2001) determined optimal times for software release when repair is imperfect. Some researchers have contributed to optimal software release policies by including testing-efforts in the cost function (Huang, 2005).

The quality of software is usually managed or controlled during the testing and maintenance phases. Although the length of testing phase directly relates to the number of errors removed but leads to a significant financial loss by increasing testing cost and delay in delivery. Further, releasing software in the market before reaching its desired level of reliability (which is fixed by the manager) may increase the maintenance cost during operational phase as well as create risk to lose future market.
To trade-off between these two conflicting objectives, multi-attribute utility theory (MAUT) is applied in our decision model.

### 4.2.7.1. Multi-attribute utility function approach

Multi-attribute utility theory is based on a set of assumptions[ see references (Keeney and Raiffa, 1976) and (Neumann and Morgenstern, 1947) for more details]. The approach has been used for choosing the most “desirable alternative” (or project) among many different alternatives. It has been used in a broad range of fields including energy, manufacturing and services, public policy, healthcare, etc.

MAUT can help in these situations by creating a decision model through the elicitation process of expert practitioner (Keeney and Raiffa, 1976). The MAUT process provides a framework through which multiple objectives and uncertainty can be combined to aid managers in making decisions.

### 4.2.7.2. Multi-Attribute Utility Function

In order to create a multi-attribute utility function, single utility functions must be assessed for every identified objective. In our case, we have identified two separate utility assessments. The objective list utilized for this preliminary analysis is minimization of cost and maximization of reliability. A Multi-Attribute Utility Function (MAUF) is defined as:

\[
U(x_1, x_2, ..., x_n) = f[u_1(x_1), u_2(x_2), ..., u_n(x_n)] \\
= \sum_{i=1}^{n} w_i u_i(x_i) ; \quad \sum_{i=1}^{n} w_i = 1
\]  

(4.31)

Here, \( U \) is a multi-attribute utility function over all utilities. \( u_i(x_i) \) is single utility function measuring the utility of attribute \( i \); and \( x_i \) is level of \( i^{th} \) attribute. The scaling constants \( w_i \) represent the weights for the utilities of attributes (also called the relative importance). In order to make a structure for utility functions, first we need to make
assumptions regarding utility independence and the additive independence assumption.

4.2.7.3. Assessing Utility Functions.

The utility functions are assessed in four steps. First, determination and quantification of attributes. Second, the component utility functions (\( u_i \)) are assessed. Third, weight parameters are estimated. Finally, the best alternative is obtained by maximizing the multi-attribute utility function.

**Step 1: Determination and quantification of attributes.**

When we are selecting the attributes, it is important to list all the attributes so that all important aspects are captured (completeness). Order the attributes (indicators) by their importance regardless of whether the data exist; if the data does not exist, find proxy variables that may be utilized. Also, eliminate redundant attributes (indicators) to avoid duplication in measuring the same attribute (Seung and Zhang, 2011).

**Step 2: Assessment of Components Utility Functions.**

Elicitation of single utility function for each attribute and assessing value to the parameter or component of utility functions is discussed here. The single utility function for each attribute represents management’s satisfaction level towards the performance of each attribute. It is usually assessed by a few particular points on the utility curve (Keeney and Raiffa, 1976) The component utility function for attribute \( i \) (\( u_i \)) is assessed by the use of lottery (Seung and Zhang, 2011) as follows:

\[
\begin{align*}
\hat{u}_i(x^{CE}_i) &= p u_i(x^B_i) + (1-p) u_i(x^W_i) \\
&= (p \cdot 1 + (1-p) \cdot 0) u_i(x^{CE}_i)
\end{align*}
\]

(4.32)

To find \( p \), for a given \( x^{CE}_i \), we need to ask the decision maker or stakeholders for allotting a value to \( p \) such that the above equation holds or use lottery. Three data points obtained from the above equation are used to determine the unknown coefficients in the utility function. The three data points are \( u_i(x^B_i) = 1, u_i(x^W_i) = 0 \) and
$u_i(x_i^{CE}) = p$. These three points are commonly used to determine parameters in the single utility function for each attribute. By substituting above values in Eq.(4.33), the value of parameter $a$, $b$ and $c$ are determined.

Finally, to determine functional form of utility functions either an additive or exponential form [see Eq. (4.33)], needs to be examined through interviews, surveys or lottery. It may be noted that we use lottery when there is a preference or indifference between two lotteries. If they are equal to each other, management is risk neutral and the linear form should be used. Otherwise, if management is not risk neutral then the exponential form will be selected.

$$u(x_i) = k.e^{l-x_i} + m \quad \text{or} \quad u(x_i) = k.x_i + m$$

(4.33)

Where $k, l$ and $m$ are constant parameters that guarantee that the utility is normalized between 0 and 1. Note that the additive form of multi-attribute utility function is based on the utility independence and the additive independence assumptions (Seung and Zhang, 2011, Keeney and Raiffa, 1976).

**Step 3: Estimation of weight parameters**

Here we discuss about estimation of weight parameter $w_i$. The weight in Eq. (4.31) is assumed to reflect the relative importance of moving an attribute from worst to best level. Thus, they are defined on ratio scale. Many approaches for obtaining numerical weights have been proposed, including direct tradeoff methods, direct judgment of swing weight and lottery-base utility assessment. By these methods, management can assign different importance to each attribute. In our case the number of attributes considered are only two, so a probabilistic scaling (lottery weight) technique is recommended (useful when there is small number of attribute).

Consider two attributes $R$ and $C$ as software reliability and software development cost. Let $(r^B, c^B)$ and $(r^W, c^W)$ denote the best and worst possible consequences, respectively (see right hand side in Figure 4.6). There is a certain joint outcome $(r^B, y^W)$ comprising of two attributes $R$ and $C$ at the best and worst level with
probability \( p \) and \((1- p)\), respectively. In these situations, the weight for attribute \( R \) equals \( p \), where \( p \) is the indifference probability between them. (Winterfeldt and Edwards, 1986).

\[ U(R_i, C_i) = w_{R_i} \times u(R_i) - w_{C_i} \times u(C_i) \]

(4.34)

\( w_{R_i} + w_{C_i} = 1 \),

Figure 4.6: Assessing scaling constants

and \( r, c \) are the levels of utility function corresponding to reliability and cost functions, respectively.

Step 4: Maximization of multi-attribute utility function.

Finally, in this step we can calculate MAUF based on the previous steps. The additive form of the MAUF in our problem is given as:

\[ \text{Max} : U(R_i, C_i) = w_{R_i} \times u(R_i) - w_{C_i} \times u(C_i) \]

Where \( R_i = R_i(s, u) ; C_i = C_i(s, u) ; s = \text{Time} \ & \ u = \text{Effort} \) and \( w_{R_i} \) and \( w_{C_i} \) are the weight parameters for the attributes, reliability and cost, respectively. \( u(R_i) \) and \( u(C_i) \) are the single utility function for each of these attribute. It may be noted that the \( U(R_i, C_i) \) function is of Max type and it has been written in terms of \( R_i \) and \( C_i \).

From manager’s point of view, \( R_i \) is to be maximized while \( C_i \) is to be minimized. To synchronize the two utility together, we put ‘−’ sign before cost utility. By maximizing this multi-attribute utility function, the optimal time to release, \( s^* \) and optimal effort, \( u^* \) will be obtained.
4.2.7.4. Structure of the Cost Function

Here we discuss about structure of the cost function, which is based on two factors, i.e., time and effort. For our model, consider that \((i-1)^{th}\) version of the software has been released in the market and company wants to know the time to stop testing for \(i^{th}\) version.

For this purpose, we propose a software cost model that includes costs of removing faults before and after the release. Also, we distinguish between cost of removing faults based on reported bugs in \(i^{th}\) and \((i-1)^{th}\) releases. The behavior of cost function is given in Figure 4.7.

Using the proposed model, the components of expected cost of the software product for \(i^{th}\) version at a given point \(\psi_{i,u}\) are defined as:

- \(C_1(\psi_{i,u})\) is the cost of the number of faults removed during testing phase, which is given as:

\[
C_1(\psi_{i,u}) = c_i^{\text{test}} \cdot (a_i + \lambda_i \cdot (a_{i-1} - m_{i-1}(\psi_{i-1} - \psi_{i-2}))) \cdot \frac{1 - e^{-\beta \psi_{i,u}}}{1 + \beta e^{-\beta \psi_{i,u}}} \tag{4.35}
\]

Where, \(c_i^{\text{test}}\) is cost of removing a fault during testing phase of \(i^{th}\) version

- Similarly \(C_2(\psi_{i,u})\) is the cost incurred by fault removal activities during the operational phase of release \(i-1\), which is given as:

\[
C_2(\psi_{i,u}) = c_{i-1}^{\text{op}} \cdot \left[ (1 - \lambda_i \cdot (a_{i-1} - m_{i-1}(\psi_{i-1} - \psi_{i-2}))) \cdot (1 - e^{-\beta \psi_{i,u}}) \right] \tag{4.36}
\]

where \(c_{i-1}^{\text{op}}\) is the cost of removing a fault during operational phase of release \((i-1)\) by testing team of release \(i\) on the basis of reported bugs.
- $C_3(\psi_{s,u})$ is assumed to be the total cost of removing faults in the operational phase of release $i$. $C_3(\psi_{s,u})$ is formulated as:

$$C_3(\psi_{s,u}) = c_i^{op} \cdot (a_i + (a_{i-1} - m_{i-1}(\psi_{s,u,i-2})) - m_i(\psi_{s,u}))$$ (4.37)

where $c_i^{op}$ cost of removing a fault during operational phase of release $i$. It may note that $0 < c_i^{test} < c_i^{op}$.

- Finally $C_4(\psi_{s,u})$ is the cost of software testing and resources as following:

$$C_4(\psi_{s,u}) = c_{\text{unit}} \psi_{s,u}$$ (4.38)

where $c_{\text{unit}}$ Cost of testing per unit of time/effort.

Using above components, the total cost function is given by:

$$\begin{align*}
\text{Cost} &= C_1(\psi_{s,u}) + C_2(\psi_{s,u}) + C_3(\psi_{s,u}) + C_4(\psi_{s,u}) \\
&= c_i^{test} \cdot m_i^{test}(\psi_{s,u}) + c_{i-1}^{op} \cdot m_{i-1}(\psi_{s,u}) + c_i^{op} \cdot (m_i(\psi_{s,u}) - m_i(\psi_{s,u,i-2})) + c_{\text{unit}} \psi_{s,u}
\end{align*}$$

(4.39)

Figure 4.7: The cost function in two-dimensional SRGM
4.2.8. Numerical Example

We applied the MAUF approach for the Tandem data set (Wood, 1996). It includes fault count data for 4th release of software. First version of software is released after 20 weeks with 10000 unit effort and time/effort for next three releases are 19, 12 and 19 testing weeks and 10270,5053 and 11305 are the amount of testing effort consumed, respectively.

In this section, we investigate about time/effort for release 4 (i=4) and check whether 19 weeks of testing time and 11305 unit effort for this release is sufficient for testing or software needs more time/effort for testing? Additionally, we check if there is any delay for releasing this version of software or not?

To answer these questions, the MAUT as discussed in section 4.2.6.1, is used. The determination of optimal planning testing time/resource is presented in the following steps.

Step 1: Determination and Quantification of Attributes

In this problem, two attributes as cost and reliability are selected. These attributes are two important factors for determination of optimal planning testing time/resource of software. The objective function of cost attribute given in the first attribute problem should be minimized.

\[
\text{Min : } C_4 = \frac{\text{cost}}{\text{budget}}
\]

(4.40)

We set parameters \(c_{4\text{est}} = 9, c_{3\text{op}} = 12, c_{4\text{op}} = 14\) and \(c_{\text{unit}} = 5\) as parameter of cost function and total budget is fixed at 400 units of currency. Also \(m_4^{\text{est}}(\psi_{s,u}), m_3^{\text{op}}(\psi_{s,u}), (a_4 + (a_3 - m_3(\psi_3 - \psi_2)) - m_4(\psi_{s,u})), \psi_{s,u}\) in the cost function are calculated by the value of estimated parameters given in the Table.4.3(a)

Although minimizing cost is important but in several cases, if this attribute is used as solitary attribute, it might cause risk for the company and users as well. Based on this
idea, manager uses reliability attribute as risk-relief measure involved with the project. The software reliability function is defined as:

\[
R(\Delta \psi \mid \psi) = \exp^{-\left(m(\psi + \Delta \psi) - m(\psi)\right)} = R^{(1)}
\]  

(4.41)

Huang (2005a) defined a new measure for software reliability as the ratio of the number of cumulative faults detected at \(\psi_{s,u}\) to the number of initial faults in the software system, i.e.

\[
R_i(\psi_{s,u}) = \frac{m_i(\psi_{s,u})}{a_i} = R^{(2)},
\]

(4.42)

Where \(a_i^* = a_i + (a_{i-1} - m_{i-1}(\psi_{i-1} - \psi_{i-2}))\).

Both forms are useful but we use second form because of simplicity and direct relation with number of faults removed. Second attribute for 4th release is given as:

\[
\text{Max : } R_4 = \frac{m_4(\psi_{s,u})}{\text{initial fault content on 4th release}}.
\]

(4.43)

It may be noted that reliability function has increasing behavior and approach to 1 when \(\psi_{s,u}\) becomes infinitely large (see Figure 4.8).

![Figure 4.8: Reliability function in two dimensional SRGM](image-url)
Step 2: Assessment of Components Utility Functions.

The single utility function for each attribute is elicited based on the management’s strategy. In our numerical example, management strategies are given as:

- For reliability attribute, management has verified that at least 60% of software faults should be detected or more the better; its highest reliability expectation is achieved when 90% of software faults are detected.
- Under minimization cost strategy, management indicates that at least 50% of budget must be consumed.
- Management demonstrates its risk neutral attitude for each attribute.

According to the above strategy, some important points on the utility curve are obtained. In particular, the lowest budget consumption requirement is $C^w_4 = 0.5$ and the highest budget consumption $C^b_4 = 1$. The lowest reliability requirement is $R^w_4 = 0.6$ and the highest reliability for this release considered as $R^b_4 = 0.9$.

In addition, the linear form of the single utility function is selected, based on management’s risk neutral attitude towards these two attributes and simple structure which is applicable in several areas (Li et al., 2011). By using Eq. (4.33) parameters $k, l$ and $m$ are determined. Specifically, we have the following equations:

$$u(C_4) = 2C_4 - 1 \; ; \; u(R_4) = \frac{10}{3}R_4 - 2$$

(4.44)

Step 3: Estimation of Weight Parameters.

In this stage, the weight parameter $w_{C_i}$ is estimated by comparing the two choices in Figure 4.6, by lottery approach. Management has claimed that it is indifferent between these two choices when $p$ is equal to 0.5, hence $w_{C_i} = 0.5$. It is easy to calculate $w_{R_i}$ because the sum of weight parameters is equal to one, therefore $w_{R_i}$ is also equal to 0.5.
Step 4: Maximization of Multi-Attribute Utility Function.

Here, based on the single utility functions and the weight parameters which have been determined in previous steps, the MAUF is evaluated and drawn in Figure 4.9.

\[
\text{Max: } \text{MUTF}(R_4, C_4) = w_{R_i} \times u(R) - w_{C_i} \times u(C_i) \\
\text{St: } l \leq C_4 \leq 1 ,
\]

(4.45)

Where

\[R_4 = g(s,u) ; C_4 = h(s,u) ; s = \text{Time } \& \ u = \text{Effort}, w_{R_i} = w_{C_i} = 0.5; w_{R_i} + w_{C_i} = 1\]

The MAUF is maximized by using Maple software package and the optimal time and effort to release are \(s^* = 24.920, u^* = 17264\) when \(l = 0.5\). Noted that decreasing \(C_4\) less than 0.5 i.e. 50 percent of total budget is restricted by cost attribute (Sensitivity of \(l\) is discussed in Section 4.2.8). According to Tandem failure data, real time to 4\(^{th}\) release is 19 weeks with 11305 unit testing effort. Based on optimal result, we can say that software in this release needs 5 more weeks of testing and 5959 units of effort.

**Figure 4.9:** MAUT for two attribute utility function [for 4\(^{th}\) release].
4.2.9. Sensitivity Analysis of the Model Parameters.

From the discussion given in the preceding section, it’s good to know that the optimal decision-making depends on various parameters that may not be precise.

The use of sensitivity analysis will help the analyst to understand how changing the parameters of the model will affect the decision outcome. The decision model is then rerun by holding all other parameters constant. We have conducted sensitivity analysis by calculating relative change of optimal time and effort based on parameters $w_{R_i}, R_i^w, C_i^B, C_i^W$.

The sensitivity of the optimal software release time/effort with respect to a model parameter, can be quantified by $\Omega_{p,0}^s, \Omega_{p,0}^u$, which are the relative change of $s^*, u^*$ when $\theta$ is changed by 100$p$ percent, i.e.

$$\Omega_{p,0}^s = \left| \frac{s^*(\theta + p\theta) - s^*(\theta)}{s^*(\theta)} \right|$$  \hspace{1cm} (4.46)

$$\Omega_{p,0}^u = \left| \frac{u^*(\theta + p\theta) - u^*(\theta)}{u^*(\theta)} \right|$$  \hspace{1cm} (4.47)

Where, $p\theta$ is amount of change in positive or negative direction and $\theta$ is parameter which is selected for the change. Two sub-cases for sensitivity are considered as follows:

Case 1: Sensitivity Analysis Based on Time

Here relative change of optimal time are calculated by Eq. (4.46). The optimal software release time obtained under the original values of model parameters is $s^* = 24.920$ and relative change based on above parameters have been calculated and shown in Table 4.5.
Table 4.5: Sensitivity Analysis Results based on Change of time

<table>
<thead>
<tr>
<th>$\Omega_{p,o}$</th>
<th>-30%</th>
<th>-20%</th>
<th>-10%</th>
<th>+10%</th>
<th>+20%</th>
<th>+30%</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Omega_{p,w_k}$</td>
<td>0.143</td>
<td>0.015</td>
<td>0.0529</td>
<td>0.0248</td>
<td>0.088</td>
<td>0.07</td>
</tr>
<tr>
<td>$\Omega_{p,R_w}$</td>
<td>0.0197</td>
<td>0.004</td>
<td>0.0450</td>
<td>0.0183</td>
<td>0.015</td>
<td>IS</td>
</tr>
<tr>
<td>$\Omega_{p,C_4}$</td>
<td>0.01</td>
<td>0.012</td>
<td>0.0176</td>
<td>IC</td>
<td>IC</td>
<td>IC</td>
</tr>
<tr>
<td>$\Omega_{p,C_4^{sw}}$</td>
<td>0.0196</td>
<td>0.022</td>
<td>0.0208</td>
<td>0.00329</td>
<td>0.0208</td>
<td>0.0184</td>
</tr>
<tr>
<td>$\Omega_{p,l}$</td>
<td>IC</td>
<td>IC</td>
<td>IC</td>
<td>0.1825</td>
<td>IS</td>
<td>IS</td>
</tr>
</tbody>
</table>

(IC: impossible change, IS: infeasible solution)

It can be seen that the sensitivity of $s^*$ with respect to model parameters $w_{k_i}$, $R_w^W$, $C_4^B$, $C_4^W$ is at acceptably low levels, e.g., when $w_{k_i}$, $C_4^W$ increases by 10 percent, the relative changes in $s^*$ is 2.4 and 3.2 percent, respectively. Also when $w_{k_i}$, $C_4^W$ decreases by 20 percent, the relative changes in $s^*$ is 1.5 and 2.2 percent, respectively. Results in Table 4.5 reveal that $w_{k_i}$ is a slightly more sensitive parameter than other parameters.

Since, the sum of the scaling constants is always equal to one, $w_{c_4}$ is not investigated in the sensitivity analysis. From Table 4.5, it can be seen that the positive change of $C_4^B$ is not possible and 30% change of $R_w^W$ doesn’t give any feasible solution. In addition, increase of more than +10% in the value of $l$ is restricted by the boundary value of the reliability attribute, i.e. $R_4^B = 0.9$.

Case 2: Sensitivity Analysis Based on Effort
Relative change of optimal efforts is calculated by Eq. (4.47) by changing the value of parameter, 30% in positive and negative direction and the result is shown in Table 4.6. The optimal effort to release the software is obtained at \( u^* = 17264 \).

**Table 4.6: Sensitivity Analysis Results Based on Change of Effort**

<table>
<thead>
<tr>
<th>( \Omega_{p,q}^{w} )</th>
<th>-30%</th>
<th>-20%</th>
<th>-10%</th>
<th>+10%</th>
<th>+20%</th>
<th>+30%</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \Omega_{p,w_i}^{w} )</td>
<td>0.0558</td>
<td>0.1659</td>
<td>0.447</td>
<td>0.3819</td>
<td>0.2498</td>
<td>0.0167</td>
</tr>
<tr>
<td>( \Omega_{p,w_i}^{w} )</td>
<td>0.005</td>
<td>0.0140</td>
<td>0.1466</td>
<td>0.275</td>
<td>0.396</td>
<td>0.398</td>
</tr>
<tr>
<td>( \Omega_{p,w_i}^{c_i} )</td>
<td>0.284</td>
<td>0.283</td>
<td>0.1985</td>
<td>IC</td>
<td>IC</td>
<td>IC</td>
</tr>
<tr>
<td>( \Omega_{p,w_i}^{c_i} )</td>
<td>IS</td>
<td>0.438</td>
<td>IS</td>
<td>0.229</td>
<td>0.1985</td>
<td>0.112</td>
</tr>
<tr>
<td>( \Omega_{p,w_i}^{l} )</td>
<td>IC</td>
<td>IC</td>
<td>IC</td>
<td>0.2207</td>
<td>IS</td>
<td>IS</td>
</tr>
</tbody>
</table>

(IC: impossible change, IS: infeasible solution)

It can be seen that the positive change of \( C_{4}^{w} \) is impossible and change of \( C_{4}^{w} \) with -30% and -10% did not give any feasible solution. In addition, based on Table 4.6, \( w_{k_i} \) is more sensitive parameter than the other parameters, which is consistent with sensitivity analysis, as based on time.