The growth models achieve a surprising trend in the evaluation of software reliability due to environment effects and change-point. The study of the software reliability growth models (SRGMs) by including environment effect and change-point concept, is called CE-SRGM. In the present investigation, we obtain the expected cost and reliability by considering the environment effects and change points in the software system. The module based error detection and removal process are taken into consideration while developing the reliability growth model. The fault detection rate (FDR) is assumed to be changed during testing phase because of environmental effects. The quantitative assessment of the total expected cost and software reliability of the system has been done by evaluating the mean value function before and after change-point. The maximum likelihood approach is employed to estimate the unknown parameters of the proposed model. Numerical results are facilitated to examine the trends of mean value functions, expected total cost and software reliability.

3.1 Introduction

In the modern society, the role of software has expanded rapidly for the last three decades and still it continues to be so. In many applications, the complex software systems are composed of a number of modules. The main objective of a software developer is that all the testing activities which he uses for different modules should be completed within a limited time-period and the total consumed amount on the efforts spent should also be limited. Successful development of a module software system leads to a great demand for high quality. Therefore quality, reliability and customer satisfaction have gradually become the essential elements for software development organizations in large scale projects and safety-critical applications such as nuclear plants, oil and gas industries, real time military and air traffic control, etc. Many household and medical appliances such as A.Cs, mobiles, switching devices, medical monitoring control, and many others are having their resembles with the above mentioned systems.
Now-a-days module based software testing is the most critical part of testing to be performed. During the development of software, it goes through many phases: such as system testing, module testing and installation testing etc., in which module testing is most important. During module testing, software failures are detected/isolated and removed individually for each module and it is checked whether the software product fulfill the user requirements or not. Lo et al. (2005) modified the SRGM based on software module structure. The estimation of parameters for NHPP based SRGM with change-point was done by Chang (2006). Tamura and Yamada (2006) developed a flexible stochastic differential equation model in distributed environment. Optimal software release policies and resource allocation problem for cost and reliability of modular software system have been investigated by Huang and Lo (2006), Jha et al. (2009) and others.

Two most important factors that affect reliability of the software among various SRGMs are the numbers of initial faults and fault detection rate (FDR). Many SRGMs assume that each failure occurs randomly and independently according to the same distribution during the fault detection process. In more realistic situations, software fails due to complexity, testing-effort, programmer skills, testing coverage, testing environment, etc.. New automated test tools, consultants or techniques could help us in detecting additional faults that are difficult to find during testing phase and usage (operational) phase. When these factors are changed during the testing phase, the failure intensity function could increase or decrease non-monotonically, which is defined as a change-point problem. Environmental factors also include other important factors that affect software reliability growth models. So it can be used to associate the fault detection rate before and after the change-point. Various software reliability measurements define different values for the same software under different environments (testing/operating). Therefore software has to be considered with environmental effects. Environmental effects on the software developing process have great impact. This type of impact on the software reliability was studied by Zhang and Pham (2000) and Zhang et al. (2001). Imperfect debugging process with change-point in reliability modeling was examined by Shyur (2003) and Chen et al. (2001). Zhao et al. (2006) considered change-point problem with environment effects. Software reliability models with change-point and testing efforts were
discussed by many researchers from time to time (cf. Huang, 2005b; Kapur et al., 2006; and Lin and Huang, 2008). To incorporate the change-point with SRGM under the different set of assumptions, the notable works have been done by many research workers (cf. Zhao, 1993; Kwang, 2001; Zou, 2003; Kapur et al., 2007). Kapur et al. (2008b) proposed SRGM with change-point and effort control using a power function of testing time. Recently, Li et al. (2010) analyzed the optimal release time of SRGM with testing efforts and multiple change-point.

To reduce the cost of the software and enhance the reliability to an acceptable level, software models considering cost, reliability, testing efforts and software quality were discussed by Huang et al. (1999b), Pham and Zhang (1999), Htoon and Thein (2005), Chiu et al. (2008) and many others. Optimum software release policies and time based on cost, reliability were suggested by Okumoto and Goel (1980), Ehrlich et al. (1993) and Huang et al. (1999a). In recent decade, the cost of developing software have become the major expenses in many computerized systems. Xie and Yang (2003) studied the effect of imperfect debugging on the software development cost. Cortellessa et al. (2008) proposed an optimization framework for “build-or-buy” decisions in software architecture. Software release policies with logistic-exponential test-effort function were explored by Rafi et al. (2010).

Generally, software development environment changes during testing because of the complexity of software system; such complex systems can be developed by integrating a number of independent modules. In this investigation, we incorporate both environment affect and change-point concepts for developing the software reliability growth model for modular software system. Both of these factors reflect more closely to a general SRGM called CE-SRGM. An overview of the proposed model along with assumptions and notations is given in section 3.2. In section 3.3, we evaluate the total expected cost to determine the optimal testing time. Section 3.4 is concerned with the reliability of the proposed model. To estimate the system parameters of software reliability growth model, the maximum likelihood estimation technique is presented in section 3.5. In section 3.6, sensitivity analysis is carried out to explore the effect of different parameters on the expected total cost and reliability. Finally, concluding remarks and future scope of the work are outlined in section 3.7.
3.2 SRGM with Change-Point and Environmental Effect

In software reliability growth models (SRGMs) based on non-homogeneous poisson process (NHPP), there are two important parameters to describe the model (i) total number of errors (ii) fault detection rate (FDR). We assume that no new faults are introduced into the software system during testing. The fault detection rate (FDR) which is a most important factor is used to analyze the effectiveness of testing techniques of fault detection and test cases. The model describes the difference between before and after the change-point because of testing environment. In many realistic situations, the environment during the software testing may not remain same. Therefore, in this model we consider a point of time when environment is changed during the testing, called change-point. After the change-point of the testing, FDR function is computed from environment function and before change-point. Because of the changes of the environment during testing, FDR changes.

The fault detection rate (FDR) function with change-point is defined as

\[
b_i(t) = \begin{cases} 
  b_i, & \text{for } 0 \leq t \leq \tau \\
  b_i', & \text{for } t > \tau 
\end{cases}
\]  

... (3.1)

where \( b_i \) is the fault detection rate before change-point and \( b_i' \) is the fault detection rate after change-point.

Assumptions

The following assumptions are made for modeling purpose:

- A software system consists of \( M \) types of modules.
- The software failure removal phenomenon follows the *non-homogeneous poisson process* (NHPP).
- Each time a failure occurs, it will be perfectly eliminated at once and no new faults are introduced.
- Correction of errors takes only negligible time and a detected error is removed with certainty.
- After the change-point of testing, the fault detection rate is the integrated result of environment effects and the fault detection rate before the change-point.
Nomenclature and Notations

The following notations are defined for the modelling purpose:

SRGM : software reliability growth model.
CE-SRGM : software reliability growth model with change-point and environment.
NHPP : non-homogeneous poisson process.
MLE : maximum likelihood estimation.
FDR : fault detection rate.
m(t) : mean value function.
a_i : total number of errors in the software including the initial and introduced errors.
b_i(t) : error detection rate.
δ : discount rate.
θ : market interest rate.
C_{0i} : setup cost of the software of i^{th} (i = 1, 2, ..., M) module.
C_{1i} : testing cost per unit time during testing-period of ith (i=1,2,...,M) module before change-point.
C_{2i} : testing cost per unit time during testing-period of ith (i = 1, 2, M) module after change-point.
C_{3i} : cost per unit time of removing an error during testing-period of i^{th} (i = 1,2,...,M) module.
C_{4i} : cost per unit time of removing an error detected during operational period of i^{th} (where i = 1, 2, ..., M) module.
u_y : expected time for removing an error during the testing period before change-point.
u'_y : expected time for removing an error during the testing period after change-point.
u_w : expected time for removing an error during the warranty period.
η_0 : scale coefficient.
η_1 : intercept value.
η_2 : degree of opportunity loss in time.
Chapter-3 Modular Software Reliability Growth Model....

\[ E\{C(t)\} : \text{total expected cost of the software.} \]
\[ T : \text{testing time of the software.} \]
\[ T^* : \text{optimal release time of the software.} \]
\[ R(x/T) : \text{reliability function, where } x \text{ is operating time of the software} \]
\[ \text{and } x \geq 0. \]

3.2.1 Time Dependent Environment Factors

Here, we discuss the environment factor which depends on the testing time. Assuming that the change-point occurs at time \( t \), and after that, the environment of the testing is changed i.e. environment function should be a time-dependent function and is given by:

\[ K_i(t) = \frac{b_i(t)}{b_i(t)} \quad t \in [\tau, \infty] \quad \ldots (3.2) \]

The average time-varying environment function is defined as follows:

\[ \bar{K}_i(t) = \frac{\bar{b}_i(t)}{b_i(t)} \quad \ldots (3.3) \]

where \( \bar{b}_i(t) \) and \( b_i(t) \) are the average fault detection rates before and after the change-point respectively.

The decreasing trend of environmental factor \( K(t) \) can be described as follows (see Yamada et al., 1983).

\[ \bar{K}(t) = \beta \exp(-\beta t) \quad \ldots (3.4) \]

In fact, there are both differences and links between the fault detection rates before and after the change-points. Thus

\[ \frac{\bar{b}_i(t)}{K(t)} = \frac{b_i(t)}{K(t)} \exp(\beta t) \quad \ldots (3.5) \]

3.2.3 Mean Value Function (MVF)

We consider a software system consisting of \( M \) modules. The environment is changed during the testing; the change-point occurs at time \( \tau \) which results in the change of MVF. Before change-point the expected number of faults detected and
removed by time \( \tau \) is \( m_i^{bf}(t) \), where suffix \( i \) is used for \( i^{th} \) \((i = 1, 2, 3, \ldots, M)\) module. The mean value function before change-point is given by

\[
m_i^{bf}(t) = \sum_{i=1}^{M} a_i \left[ 1 - \exp\left( -b_i t^k \right) \right] \quad \text{for} \quad 0 \leq t \leq \tau \quad \ldots (3.6)
\]

When \( t \to \infty \), the expected number of faults to be removed is \( m(\infty) = a_i \). After the change-point, the expected number of faults detected and removed by time \( t \) is \( m_i^{af}(t) \).

The mean value function for \( i^{th} \) \((i = 1, 2, 3, \ldots, M)\) module after change-point is

\[
\frac{dm_i^{af}(t)}{dt} = \sum_{i=1}^{M} b_i(t) \left[ a_i - m(\tau) - m_i^{af}(t) \right] \quad \text{for} \quad t > \tau \quad \ldots (3.7)
\]

so that

\[
m_i^{af}(t) = \sum_{i=1}^{M} \left[ a_i - m(\tau) \right] \left[ 1 - \exp\left( -G^+(t) \right) \right] \quad \text{for} \quad t > \tau \quad \ldots (3.8)
\]

where \( G^+(t) = \sum_{i=1}^{M} \int_{\tau}^{t} b_i(x) \, dx = \sum_{i=1}^{M} \frac{1}{\beta_\delta} \left[ \exp(\delta t) - \exp(\delta \tau) \right] \).

Thus

\[
m_i(t) = \begin{cases} 
\sum_{i=1}^{M} a_i \left[ 1 - \exp\left( -b_i t^k \right) \right], & \text{for} \quad 0 \leq t \leq \tau \\
\sum_{i=1}^{M} a_i \left[ 1 - \exp\left\{ -b_i \left( \tau^k + \frac{1}{\beta_\delta} \left( e^{\delta \tau} - e^{\delta \tau} \right) \right) \right\}], & \text{for} \quad t > \tau 
\end{cases} \quad \ldots (3.9)
\]

Corresponding failure intensity function is obtained as

\[
\lambda(t) = \frac{dm_i(t)}{dt} = \begin{cases} 
\sum_{i=1}^{M} a_i b_i k t^{k-1} \exp\left( -b_i t^k \right), & \text{for} \quad 0 \leq t \leq \tau \\
\sum_{i=1}^{M} a_i b_i e^{\delta t} \exp\left\{ -b_i \left( \tau^k + \frac{1}{\beta_\delta} \left( e^{\delta \tau} - e^{\delta \tau} \right) \right) \right\}, & \text{for} \quad t > \tau 
\end{cases} \quad \ldots (3.10)
\]

**Special Cases:**

(i) When \( k = 1 \), the expected number of faults removed is represented by exponential type curve.

(ii) When \( k = 2 \), the expected number of faults removed reveals Rayleigh type curve.
3.3 Expected Total Cost of the Software

The quality of the software depends on the testing time and testing methodologies. The cost of removing errors in operational period is much higher than that in the testing period \((C_{4i} > C_{3i})\) as debugging process may increase some external costs in the operational phase. The cost function involves the following cost elements:

(i) The software development cost with discount rate during testing and operational period.

(ii) Cost due to change-point in testing time.

(iii) Risk cost due to software failure.

(iv) Opportunity cost of delaying the release of the software.

(v) Since software has always a risk of failures after release as such the criteria of fault free software does not exist in real time software system. The software risk cost is obtained as

\[ C_{si}(T) = C_{si}\{1 - R_i(x/T)\} \]

(vi) There is one more cost element related to opportunity cost. The postponing the release time of the software may lead to tangible and intangible losses. As such this cost factor is governed by power law and is given by

\[ C_{6i}(T) = \eta_i(\eta_i + T)^\eta \]

The total expected cost for modular software system can be computed as follows:

\[
E[C(T)] = \sum_{i=1}^{M} E[C_i(T)] = \sum_{i=1}^{M} C_{0i} + \sum_{i=1}^{M} C_{Ti}T^s + \sum_{i=1}^{M} C_{2i}m_1(\tau)u_x + \sum_{i=1}^{M} C_{3i}m_1(T - \tau)u_y + \sum_{i=1}^{M} C_{4i}[m_i(T + T_w) - m_i(T)]u_w + \sum_{i=1}^{M} C_{5i}(T) + \sum_{i=1}^{M} C_{6i}(T)
\]

\[
\sum_{i=1}^{M} E[C_i(T)] = \sum_{i=1}^{M} C_{0i} + \sum_{i=1}^{M} C_{Ti}T^s + \sum_{i=1}^{M} C_{2i}u_x \int_0^T \lambda_i(t)e^{-\theta\tau}dt + \sum_{i=1}^{M} C_{3i}u_y \int_{\tau}^{T+\theta\tau} \lambda_i(t)e^{-\theta\tau}dt + \sum_{i=1}^{M} C_{4i}u_w \int_T^{T+T_w} \lambda_i(t)e^{-\theta\tau}dt + \sum_{i=1}^{M} C_{5i}[1 - R_i(\frac{x}{T})] + \eta_i(\eta_i + T)^\eta.
\]

\[ \text{...(3.11)} \]
3.4 Reliability Evaluation

The reliability $R(x/T)$ of the software is the probability that the software failure does not occur in time interval $(T, T+x]$ given that the software failure has occurred at time $T$ and $(T > 0, x > 0)$.

The reliability function for the software can be obtained as

$$R(x/T) = \sum_{i=1}^{M} \exp[-\{m_i(T+x) - m_i(T)\}]$$

$$= \sum_{i=1}^{M} \exp\left[ a_i e^{-b_i \tau} \left( \exp\left( -\frac{b_i}{\beta} (e^{\beta(T+x)} - e^{\beta T}) \right) - \exp\left( -\frac{b_i}{\beta} (e^{\beta T} - e^{\beta \tau}) \right) \right) \right] \quad \ldots (3.12)$$

3.5 Optimization Problem

It is very important for software developer to control the testing resources such as testing time, cost, testing efforts and manpower. Since during the testing period, the detected software faults in each software module are removed and this process continuous until the software modules achieve the specific standard i.e., each software module gets a certain degree of reliability.

Thus the optimization problem can be formulated as follows:

Minimize $\sum_{i=1}^{M} E[C_i(t)]$

Subject to $R(x/T) \geq R_0$ \quad \ldots (3.13)

In order to determine optimal release time $T=T^*$, we establish some results in the form of theorems as follows:

**Theorem 1** : $\sum_{i=1}^{M} E[C_i(t)]$ gives a minimum value at $T^* = T_i$ where $0 < T_i < \infty$.

**Proof** : For proof see appendix A.
Theorem 2 : \( E[C(T)] = \sum_{i=1}^{M} EC_i(t) \) is a convex function with respect to \( T \).

Proof: Here \( \frac{dEC(T)}{dT} = 0 \) has only one finite solution and gives a minimum value at \( T^* = T \). Also it satisfies \( \frac{d^2EC(T)}{dT^2} > 0 \).

Thus, \( \sum_{i=1}^{M} EC_i(t) \) is a convex function with respect to \( T \).

3.6 Parameter Estimation

In this section, logarithmic maximum likelihood function (Chen et al., 2001) is used to estimate the unknown parameters i.e. \( a_i, b_i, \tau \) for the proposed CE-SRGM.

Let \( t_1, t_2, \ldots, t_N \) be the random failure time of \( N \) items where \( 0 < t_1 < t_2 < \ldots < t_N \). Let \( y_j \) (1 \( \leq j \leq N \)) be the cumulative number of detected errors. Then log likelihood function (LLF) is given by

\[
\log L[a, b, \tau] = \sum_{i=1}^{M} \sum_{j=1}^{N} \left[ y(t_j) - y(t_{j-1}) \right] \left[ m(t_j) - m(t_{j-1}) \right] - \log \left[ m(t_j) - m(t_{j-1}) \right] - \log \left[ y(t_j) - y(t_{j-1}) \right] \]
\]

... (3.14)

For mean value function, we consider the two cases (i) before change-point and (ii) after change-point.

Case (i): Before change-point, the mean value function is

\[
m_i^{bf}(t) = \sum_{i=1}^{M} a_i \left[ 1 - \exp \left( -b_i t^k \right) \right], \quad \text{for} \ 0 \leq t \leq \tau
\]

After putting the value of \( m_i^{bf}(t) \) in equation (3.14) and differentiating it with respect to \( a_i \) and \( b_i \), we obtain

\[
\frac{\partial L}{\partial a_i} = \sum_{i=1}^{M} \sum_{j=1}^{N} \left[ y(t_j) - y(t_{j-1}) \right] \left[ e^{-b_i t_j^k} - e^{-b_i t_{j-1}^k} \right] \frac{1}{a_i} \]
\]

... (3.15)

\[
\frac{\partial L}{\partial b_i} = \sum_{i=1}^{M} \sum_{j=1}^{N} \left[ \left( t_j^k a_i e^{-b_i t_j^k} - t_{j-1}^k a_i e^{-b_i t_{j-1}^k} \right) \left[ y(t_j) - y(t_{j-1}) \right] - \frac{1}{a_i} \left[ e^{-b_i t_j^k} - e^{-b_i t_{j-1}^k} \right] \right]
\]

... (3.16)
Chapter-3 Modular Software Reliability Growth Model....

The parameters $a_i$ and $b_i$, can be estimated by using numerical method to solve the simultaneous equations:

$$\frac{\partial \ln(L)}{\partial a_i} = \frac{\partial \ln(L)}{\partial b_i} = 0$$

**Case (ii):** After change-point, the mean value function is

$$m_i^{af}(t) = \sum_{i=1}^{M} a_i \frac{b_i}{\beta} e^{\beta t} \cdot \exp\left\{-b_i \left( \tau^k + \frac{1}{\beta \delta} \left(e^{\beta t} - e^{\beta \tau}\right)\right)\right\}, \text{ for } t > \tau$$

After putting the value of $m_i^{af}(t)$ in equation (3.14) and differentiating with respect to $a_i$, $b_i$ and $\tau$, we get the following equations:

$$\frac{\partial L}{\partial a_i} = \sum_{i=1}^{M} \sum_{j=1}^{N} \left[ y(t_j) - y(t_{j-1}) \right] \left[ \frac{b_i}{\beta} e^{\beta t_j} \left( e^{\beta t_j} \exp\left(-\frac{b_i}{\beta \delta} \left(e^{\beta t_j} - e^{\beta \tau}\right)\right) - e^{\beta t_j} \exp\left(-\frac{b_i}{\beta \delta} \left(e^{\beta t_j} - e^{\beta \tau}\right)\right)\right) - \frac{1}{a_i}\right]$$

$$\frac{\partial L}{\partial b_i} = \sum_{i=1}^{M} \sum_{j=1}^{N} \left[ y(t_j) - y(t_{j-1}) \right] \left[ \frac{b_i}{\beta} e^{\beta t_j} \left( e^{\beta t_j} \exp\left(-\frac{b_i}{\beta \delta} \left(e^{\beta t_j} - e^{\beta \tau}\right)\right) - e^{\beta t_j} \exp\left(-\frac{b_i}{\beta \delta} \left(e^{\beta t_j} - e^{\beta \tau}\right)\right)\right)\right]$$

$$\frac{\partial L}{\partial \tau} = \sum_{i=1}^{M} \sum_{j=1}^{N} \left[ b_i \left( e^{\beta \tau} k^{k-1} \right) \left\{ y(t_j) - y(t_{j-1}) \right\} \frac{a_i b_i}{\beta} \left( e^{\beta t_j} A - e^{\beta t_j} \right) - 1\right]$$

where $A = \exp\left(-b_i \left( \tau^k + \frac{1}{\beta \delta} \left(e^{\beta \tau} - e^{\beta \tau}\right)\right)\right)$, $B = \exp\left(-b_i \left( \tau^k + \frac{1}{\beta \delta} \left(e^{\beta \tau} - e^{\beta \tau}\right)\right)\right)$. The parameters $a_i$, $b_i$, and $\tau$ can be obtained by using numerical method to solve the simultaneous equations:

$$\frac{\partial \ln(L)}{\partial a_i} = \frac{\partial \ln(L)}{\partial b_i} = \frac{\partial \ln(L)}{\partial \tau} = 0$$

The above equations are typical and difficult to be solved analytically. However to obtain the estimated values of the unknown parameters $a_i$, $b_i$ and $\tau$, for both cases
Chapter-3 Modular Software Reliability Growth Model....

i.e. before change-point and after change-point, we can employ a numerical technique
by using softwares namely MATLAB, MAPPLE, MATHEMATICA, etc..

3.7 Numerical Results

In this section, we provide numerical results by coding computer program in
MATLAB software to examine the validity and tractability of analytical results of the
proposed module based CE-SRGM. The sensitivity analysis is presented to visualize
the effect of different parameters such as initial fault content, FDR, cost of removing
an error in testing and operational period. The optimal release time of the software has
been obtained, which satisfy both cost and reliability requirements. To obtain the
optimal testing time, we set default parameters as \( a_1=50, a_2=70, b_1=.005, b_2=.006, \)
\( \tau=20, \beta=13, \delta=0.1, \eta_0=1.1, \eta_1=.01, \eta_2=2. x=.9, u_y=.02, u_y'=0.02, u_w=.02, \theta=.5, T_w=1. \)

Some cost elements have been chosen for different data sets which are as follows:

Set I : \( C_0=800, C_1=450, C_2=500, C_3=550, C_4=600, C_5=700. \)

Set II : \( C_0=900, C_1=450, C_2=500, C_3=550, C_4=600, C_5=700. \)

Set III : \( C_0=800, C_1=500, C_2=500, C_3=550, C_4=600, C_5=700. \)

Set IV : \( C_0=800, C_1=450, C_2=530, C_3=550, C_4=600, C_5=700. \)

Set V : \( C_0=800, C_1=450, C_2=500, C_3=580, C_4=600, C_5=700. \)

Set VI : \( C_0=800, C_1=450, C_2=500, C_3=550, C_4=650, C_5=700. \)

Set VII : \( C_0=800, C_1=450, C_2=500, C_3=550, C_4=600, C_5=800. \)

In table 3.1, the optimal release time, corresponding minimal cost and the
reliability for testing and operational periods are summarized.

Figs 3.1(i-iii) depict the trend of the cumulative errors (MVF) with respect to
time by varying the parameters \( a_1, b_1 \) and \( \beta \). We notice that the number of errors
increase sharply with respect to time initially and after some time it becomes almost
constant. From figs 3.1(i) and 3.1(ii), it is clear that MVF increases when \( a_1 \) and \( b_1 \)
increase. In fig. 3.1(iii), we see that the number of errors increases as \( \beta \) increases up to
time \( T=20 \) and after that the number of errors decreases as \( \beta \) increases and
approximately after \( T=60 \), it tends to be linearly constant. In fig. 3.1(iv), MVF is
exhibited for single and double modules; we notice that it is quite different for these
modules. It is found that the number of errors in single module system is less than that
of double module system.
In figs 3.2(i-iv), we have displayed the difference between expected cost for system with single module (M=1) and double modules (M=2). From fig. 3.2 (i) we notice that EC(T) first decreases upto T=20 then it increases approximately upto till T=70, then after again decreases upto T=80 and finally it increases linearly. In figs 3.2(ii) and 3.2(iii), the cost increases as \( a_1 \) and \( b_1 \) respectively, increase. However, fig. 3.2(iv) shows the reverse pattern with respect to \( \beta \) i.e. as \( \beta \) increases, the cost decreases. Furthermore, in first module system, EC(T) is less than that for the double module system.

In figs 3.3 (i-ii)-3.4(i-ii), we demonstrate the reliability of the modular system. Figs 3 (i) and 3 (ii) show the reliability curve for single module (M=1) with respect to time. The effects of \( a_1 \) and \( b_1 \) are noticed to be same in figs 3.3(i) and 3.3(ii), respectively. We see that the reliability decreases on increasing \( a_1 \) whereas there is negligible effect of the increase in \( b_1 \) on the reliability. Figs 3.4(i) and 3.4(ii) demonstrate the reliability for double module (M=2) system. In both cases (M=1 & M=2), first reliability decreases upto T=50 and after that it increases sharply upto T=120 and finally gradually becomes constant. The effects of variation in \( a_1 \) and \( b_1 \) are same as in case of module 1. By the comparison of these figs with respect to time, we notice that single module software is more reliable than the double module software, which is same as we expect in real time system.

The optimal release time of the software is exhibited by fig. 3.5. We examine that how much money the manufacturer should spent on the software so that the total expected cost EC(T) should be minimized and the reliability should be maximized. We analyze that till T=20, EC(T) decreases and reliability increases. After T=20, EC(T) increases but reliability decreases till T=55 approximately because of environment effects, and after that again EC(T) decreases and reliability increases. Finally at T=80, EC(T) attains the minimum value and reliability achieves the maximum value. This is the optimum point at which software should be released.

Overall, we conclude the following from the sensitivity analysis performed.

- By increasing the initial error content \( a_1 \) and \( a_2 \), MVF increases but EC(T) and reliability decrease.
By the comparison of module-based software, we infer that single module system is less costly than that of the double module system. Also the cost and reliability is high for single module system in comparison to double module system.

As testing time increases, EC(T) first decreases, then increases after some time, then after it again decreases and finally gradually increases.

3.8 Conclusion

In this chapter, we have developed a modular software reliability growth model for testing of the software. From the perspective of environment effect and change-point in the SRGMs, we have discussed the optimal testing time of the software. Our study is concerned with a more realistic model for the software development and may be helpful to the manufacturer who can make a best decision in practice. In our model, we have incorporated the Weibull distribution for the cumulative number of faults detected (MVF). The effect of change-point and environment on the MVF is also taken into consideration. Various interesting indices such as total expected cost and reliability for the software have been obtained. Due to incorporation of environment effect and change-point concept, the model provides a more versatile and feasible description of the real time software system than other SRGMs. The suggested optimal testing time based on the optimal release policies provides an insight to the software developer to ensure the better quality of the software under pre-specified reliability.
Table 3.1: Optimal testing time for different data sets for $EC(T^*)$, $R(X/T^*)$ and $R(X/T^*+T^*_w)$

<table>
<thead>
<tr>
<th>Set</th>
<th>$T^*$</th>
<th>$EC(T^*)$</th>
<th>$R(X/T^*)$</th>
<th>$R(X/T^<em>+T^</em>_w)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set I</td>
<td>78.9</td>
<td>791.08</td>
<td>0.8443</td>
<td>0.8456</td>
</tr>
<tr>
<td>Set II</td>
<td>80.8</td>
<td>891.83</td>
<td>0.8465</td>
<td>0.8472</td>
</tr>
<tr>
<td>Set III</td>
<td>80.8</td>
<td>791.83</td>
<td>0.8465</td>
<td>0.8472</td>
</tr>
<tr>
<td>Set IV</td>
<td>78.9</td>
<td>787.79</td>
<td>0.8443</td>
<td>0.8456</td>
</tr>
<tr>
<td>Set V</td>
<td>79.6</td>
<td>778.13</td>
<td>0.8453</td>
<td>0.8463</td>
</tr>
<tr>
<td>Set VI</td>
<td>78.1</td>
<td>814.17</td>
<td>0.8428</td>
<td>0.8446</td>
</tr>
<tr>
<td>Set VII</td>
<td>79</td>
<td>791.17</td>
<td>0.8445</td>
<td>0.8457</td>
</tr>
</tbody>
</table>

Table 3.1: Optimal testing time for different data sets for $EC(T^*)$, $R(X/T^*)$ and $R(X/T^*+T^*_w)$
Fig. 3.1: Cumulative errors by varying (i) $a_1$ (ii) $b_1$ (iii) $\beta$ and (iv) module based MVF by varying $a_1$. 
Fig. 3.2: Expected cost for the module based software system by varying
(i) T (ii) a₁ (iii) b₁ (iv) beta.
Fig. 3.3: Software reliability for M=1 by varying (i) $a_1$ (ii) $b_1$.

Fig. 3.4: Software reliability profile for M=2 by varying (i) $a_1$ (ii) $b_1$. 
Fig. 3.5: Optimal release time of the software
Chapter-3 Modular Software Reliability Growth Model....

Appendix - A

From equation (3.14), we have

\[ \sum_{i=1}^{M} E[C(T)] = \sum_{i=1}^{M} C_{i} + \sum_{i=1}^{M} C_{i} \int_{0}^{\tau} e^{-\beta t} \lambda_{i}(t) e^{-\beta t} dt + \sum_{i=1}^{M} C_{i} \int_{0}^{\tau} e^{-\beta t} \lambda_{i}(t) e^{-\beta t} dt + \sum_{i=1}^{M} C_{i} \int_{0}^{T} e^{-\beta t} \lambda_{i}(t) e^{-\beta t} dt 

+ \sum_{i=1}^{M} C_{i} \left[ 1 - R \left( \frac{Y}{T} \right) + \eta_{i} \eta_{i} + \eta_{i} T \right] \]

\[ = \sum_{i=1}^{M} C_{i} + \sum_{i=1}^{M} C_{i} e^{-\theta T} \left[ 1 - e^{-\theta T} \right] + \sum_{i=1}^{M} C_{i} \int_{0}^{\tau} a_{i} b_{i} k \tau^{i-1} e^{-b_{i} t} dt + \sum_{i=1}^{M} C_{i} \int_{t}^{T} a_{i} b_{i} \exp \left[ \left( \tau^{i} + \frac{1}{b_{i}} (e^{\beta t} - e^{\beta T}) \right) \right] e^{-\beta t} dt 

+ \sum_{i=1}^{M} C_{i} \left[ 1 - \exp \left[ a_{i} e^{-b_{i} \tau^{i}} \right] \left[ \exp \left( -\frac{b_{i}}{\beta} (e^{\beta T} - e^{\beta \tau}) \right) \right] - \exp \left( -\frac{b_{i}}{\beta} (e^{\beta T} - e^{\beta \tau}) \right) \right] + \eta_{i} \eta_{i} T \]

Now for \( k = 1 \), we get

\[ E[C_{i}(t)] = \sum_{i=1}^{M} C_{i} + \sum_{i=1}^{M} C_{i} e^{-\theta T} \left[ 1 - e^{-\theta T} \right] + \sum_{i=1}^{M} C_{i} \int_{0}^{\tau} a_{i} b_{i} \left( \frac{1}{b_{i} + \theta} \right) \left[ e^{-\frac{1}{b_{i}} (e^{\beta \tau})} - 1 \right] \]

\[ + \sum_{i=1}^{M} C_{i} \int_{t}^{T} a_{i} b_{i} \exp \left[ \left( \tau^{i} + \frac{1}{b_{i}} (e^{\beta t} - e^{\beta \tau}) \right) \right] e^{-\beta t} dt 

+ \sum_{i=1}^{M} C_{i} \left[ 1 - \exp \left[ a_{i} e^{-b_{i} \tau^{i}} \right] \left[ \exp \left( -\frac{b_{i}}{\beta} (e^{\beta T} - e^{\beta \tau}) \right) \right] - \exp \left( -\frac{b_{i}}{\beta} (e^{\beta T} - e^{\beta \tau}) \right) \right] + \eta_{i} \eta_{i} T \]

Also

\[ \frac{d}{dT} E[C_{i}(t)] = \sum_{i=1}^{M} C_{i} \left[ 1 - e^{-\theta T} \right] (\theta) e^{-\theta T} + \sum_{i=1}^{M} C_{i} \int_{t}^{T} a_{i} b_{i} \left[ \left( 1 - b_{i} \tau^{i} + \frac{b_{i}}{\beta} (e^{\beta \tau} - 1) \right) \right] e^{-\beta t} dt 

+ \sum_{i=1}^{M} C_{i} \left[ 1 - \exp \left[ a_{i} e^{-b_{i} \tau^{i}} \right] \left[ \exp \left( -\frac{b_{i}}{\beta} (e^{\beta T} - e^{\beta \tau}) \right) \right] - \exp \left( -\frac{b_{i}}{\beta} (e^{\beta T} - e^{\beta \tau}) \right) \right] + \eta_{i} \eta_{i} T \]

\[ = \frac{1}{2} \left( \frac{b_{i}}{\beta} - \theta \right) \left[ T - \frac{1}{2} (\frac{b_{i}}{\beta} - \theta) \right] + \sum_{i=1}^{M} C_{i} \int_{t}^{T} a_{i} b_{i} \left[ \left( 1 - b_{i} \tau^{i} + \frac{b_{i}}{\beta} (e^{\beta \tau} - 1) \right) \right] e^{-\beta t} dt 

+ \sum_{i=1}^{M} C_{i} \left[ 1 - \exp \left[ a_{i} e^{-b_{i} \tau^{i}} \right] \left[ \exp \left( -\frac{b_{i}}{\beta} (e^{\beta T} - e^{\beta \tau}) \right) \right] - \exp \left( -\frac{b_{i}}{\beta} (e^{\beta T} - e^{\beta \tau}) \right) \right] + \eta_{i} \eta_{i} T \]
Chapter-2: Optimal testing time for software reliability ....