Testing Effort Control using Flexible Software Reliability Growth Model with Change Point

P. K. KAPUR1*, ANU GUPTA1, OMAR SHATNAWI2 and V.S.S. YADAVALLI3

1 Department of Operational Research, University of Delhi, Delhi, India
2 Department of Computer Science, Prince Hussein bin Abdullah Information Technology College, Mafraq, Jordon
3 Department of Industrial and Systems Engineering, University of Pretoria, South Africa

(Received on March 10, 2006)

Abstract: There exist various Software Reliability Growth models (SRGMs) in the software reliability engineering literature which assume diverse testing environments like distinction between failure and removal processes, learning of the testing personnel, possibility of imperfect debugging and error generation etc. But most of them are based upon constant or monotonically increasing Fault Detection Rate (FDR). In practice, as the testing grows, so does the skill and efficiency of the testers. With the introduction of new testing strategies and new test cases, there comes a change in FDR. The time point where the change in removal curve appears is termed as 'change point'. In this paper we incorporate the concept of change point in flexible SRGM with testing effort. We further extend to include the problem of 'Testing Effort Control'. When the testing is in its late stage and the product release date is approaching, an assessment is done to review the progress of testing and requirement for the additional efforts is worked out to meet the pre-specified reliability targets. The proposed model is extended to yield the trade off analysis with respect to aspiration level for reliability of the product. The predictive power and accuracy of the model has been worked on two real failure datasets. The results obtained show remarkable improvements and are fairly encouraging.

Key Words: non homogenous Poisson process, software reliability growth models, change point, cumulative testing effort function, testing effort control problem.

1. Introduction

With the intrusion of computers and technology in every walk of our lives- improper functioning / failure of software can cause serious problems. There are already numerous instances where failure of computer controlled systems has led to colossal loss of human lives and money. With increased complexity of products design, shortened development cycles and highly destructive consequences of software failures, a major responsibility lies in the areas of Software Debugging, Testing and Verification. As software systems have become more and more complex, the importance of effective, well planned testing has increased many folds.

* Communicating author’s email: pkkapur@or.du.ac.in
Testing is an important part of the software development process and is performed at the end of each stage of the lifecycle. A successful test strategy begins by taking requirements specification into consideration and it continues to be carried out by software developers and separate test groups even after code implementation. Software testing is defined as the execution of a program to find the faults, which might have been introduced in it during various stages of the Development Cycle. Thus, a successful test is one that finds a defect. Besides finding faults, testing is also performed to judge the performance, safety, fault-tolerance or security of the Software.

More importantly, testing provide a mathematical measure of software reliability (i.e. failures/execution time) which forms a vital input to the release decision. Though testing helps in assessing and improving system level performance and usability, it cannot be performed indefinitely. With intense competition in the market and increased complexity of the software systems, the targets and efforts are set for the testing. For projects of large size, more testing will usually reveal more bugs. The question then arises when to stop testing, and to what level of bugs.

It is the testing phase, which is amenable to mathematical modeling. Models that describe the failure phenomenon and consequent enhancement in reliability due to fault removal are termed as SRGMs. Several SRGMs, which relate the number of failures (faults identified/removed) and execution time, have been discussed in the literature. These models are used to predict fault content and reliability of the software with respect to the calendar time. There exists a very close relationship between reliability of software and the amount of testing it has been subjected to. Longer the testing, higher is the reliability of the software. The resources, which are involved during testing of any software basically, include manpower used for fault detection / removal and CPU time spent in executing software under test. Greater the amount of testing effort faster is the testing process. It has been observed that the relationship between the testing time / effort and the corresponding number of faults removed is either Exponential or S-Shaped. There is another important class of SRGMs, known as flexible SRGMs, which can depict, depending upon parameter values, both exponential and S-Shaped growth curve (Obha [12], Bittanti et. al. [2] and Kapur and Garg [6]). The amount of efforts spent during testing and debugging process indicates the length and extent of testing. To model the cumulative testing effort function, various distributions- Exponential, Rayleigh, Weibull, Logistic etc have been discussed (Musa et. al.[11]; Kapur et. al. [8]).

In most of the Software reliability growth models proposed so far, the faults are assumed to be uniformly distributed where each fault has an equal chance of detection e.g. Goel–Okumoto model [4]. It is often assumed that each fault detection leads to exactly one removal - Yamada Delayed S-shaped model [16], Kapur 3-stage Erlang model [8]. But later on Kapur & Garg [7], Obha inflection point model [12] and Bittanti [2] proposed models where fault detection rate is defined as function of number of faults already removed.

In this paper, we start with the discussion on flexible SRGM due to Kapur and Garg [7]. It is based on the assumption that failure intensity is not only function of residual fault content but also depends on the proportion of faults already detected. As the testing progresses, the testing team gains experience and with the employment of new tools and techniques, the fault detection rate gets changed. This change can also be caused by shift in testing strategy, defect density, introduction of new test cases, and induction of skilled
personnel in team or simply by the increase in efficiency of present team. The point of
time where the change in FDR is observed can be termed as ‘Change Point’. Very few
attempts have been made to incorporate the ‘Change point’ in failure growth
phenomenon. The work in this area started with Zhao [18] who introduced the change
point analysis in Hardware and Software reliability. Some pioneering work has been done
in the area by Shyur [13]; Chang [3]; Wang [15]. In this paper an attempt has been made
to relate various type of testing effort function with change point problem.

In the next section we extend our model to present the testing effort control problem.
In practice, during testing, the managers have got the targets to meet in terms of reliability
achieved and number of bugs removed within the stipulated time. The presence of intense
competition and ever changing user requirements combined with other factors like higher
customer satisfaction level of quality have placed a major responsibility on the software
development organizations with respect to Release time and quality of the software.
Continuous reviews are made for the progress of the testing and debugging process and
new technologies and testing strategies combined with new tools are introduced if the
progress made does not seem to be satisfactory.

Using the collected failure data, the requirement for the testing resources to meet the
pre-decided reliability level is generated and the estimates for the extra resources are also
worked out if higher level is desired. The problem explained here is termed as ‘Testing
Effort Control’ problem. This type of study is carried out when the testing is in its last
stage and the release time is approaching. It is basically done to ensure the on-time and
efficient delivery by rescheduling the resources. It helps in advance planning for the
resources so that the software is ready for the release on its due date. If the fault removal
process lacks in efficiency, more efforts are put in to expedite it and to achieve the
aspiration level. In order to detect more faults in less time, new test cases, tools with more
efficient testing people & new testing strategies are infused in the process. Software with
improper debugging and high fault content can prove to be a hazard to the safety of the
systems and can have destructive consequences. On the other hand, the delayed release
can mean the future revenue lost with competitors offering early delivery. So the
organizations have got targets to meet with respect to good quality and release in the
market.

‘Testing Effort Control’ analysis helps in advance planning and maintaining long-
term user-client relationship by assuring high quality product.

The paper is organized as follows: Section 2 describes the Non Homogenous Poisson
Process and the assumptions for the proposed model. Section 3 discusses flexible Error
removal SRGM due to Kapur and Garg [7]. Then we incorporate the change point in the
SRGM to take care of sudden changes in FDR in Section 4. In the same section we
present testing effort control problem along with change point using four different types
distribution functions to describe cumulative effort function-Exponential; Rayleigh;
Weibull and Logistic. Sections 5 and 6 provide the method used for parameter estimation
and the criteria used for validation and evaluation of the developed model respectively.
The applications of the developed model to actual software reliability data collected from
real software development projects are shown in Section 7. In section 8 we extend the
model with ‘n’ number of change points that make the basis for our future work. Section 9
concludes the paper.
Acronyms

SRGM  Software Reliability Growth Model
NHPP  Non-Homogeneous Poisson Process
FDR  Fault Detection Rate
KG   Kapur and Garg Model
MLE  Maximum Likelihood Estimate
DS   Data Set
LOC  Lines of Source Code
MSE  Mean Square Fitting Error
R²   Coefficient of Multiple Determination

2. Basic Assumption

The SRGM presented in this paper is based upon NHPP. The NHPP models are based on the assumption that the software system is subject to failures at random times caused by manifestation of remaining faults in the system. Hence NHPP are used to describe the failure phenomenon during the testing phase. The counting process \( \{N(t), t \geq 0\} \) of an NHPP process is given as follows.

\[
\Pr \left\{ N(t) = k \right\} = \frac{(m(t))^k}{k!} e^{-m(t)}, \quad k = 0, 1, 2, \ldots
\]

The intensity function \( \lambda(x) \) (or the mean value function \( m(t) \)) is the basic building block of all the NHPP models existing in the software reliability engineering literature. These models assume diverse testing environments like distinction between failure and removal processes, learning of the testing personnel, possibility of imperfect debugging and error generation etc. Yamada Delayed S-shaped model [16]; Kapur and Garg [7]; Kapur et al. [8]; Trachtenberg [14]. Faults if present in the software are exposed when the software is run.

The proposed model is based upon the following basic assumptions.

1. Software failure phenomenon can be described by the Non-homogeneous Poisson Process (NHPP).
2. The number of failures during testing is dependent upon the number of faults remaining in the software as well as on the number of faults already identified at that time by the current testing effort level.
3. As soon as a failure occurs the fault causing that failure is immediately identified. Identified faults are removed perfectly and no additional faults are introduced during the process i.e. debugging process is perfect.
4. Cumulative Testing effort function is modeled by four different types of Distribution—Exponential, Rayleigh, Weibull and Logistic.

\* The singular and plural of an acronym are always spelled the same
3. Modeling Software Reliability

3.1. KG Model [7]

This model is based upon the assumption that the detection of errors also results in detection of some of the remaining errors without these errors causing any failure.

The differential equation for this model is given by

\[ m(t) = p(a - m(t)) + q \left( \frac{m(t)}{a} \right) (a - m(t)) \]  

(2)

The solution is,

\[ m(t) = a \left( \frac{1 - e^{-(p+q)t}}{1 + \frac{q}{p} e^{-(p+q)t}} \right) \]  

(3)

where

- \( p \) : Failure occurrence rate.
- \( q \) : Fault removal rate of additional removed faults.

This model was developed to account for some additional faults being detected without their causing failure. Here we can observe that if \( q=0 \); then on each failure, only the error causing the failure is removed and it corresponds to Exponential model. The failure growth curve defined by above model is S-shaped whose nature depends on \((q/p)\).

3.2 Alternative Formulation for KG Model [7]

Let us define:

\[ \frac{d}{dt} m(t) = b(t)(a - m(t)) \]  

(4)

With \( b(t) = \frac{b}{1 + \beta e^{-bt}} \)

Upon solving we get:

\[ m(t) = a \left( \frac{1 - e^{-bt}}{1 + \beta e^{-bt}} \right) \]  

(5)

By setting \( b=p+q \) and \( \beta=(q/p) \) it can be seen that (3) and (5) are same.

Here, in the alternate formulation, it may be observed that the fault detection rate for the above specified model is a logistic function and \( b(t) \to b \) as \( t \to \infty \). Here \( \beta \) is the learning factor and it represents the skill and experience gained by the testers during testing. If \( \beta=0 \), then \( b(t)=b \), i.e. constant.

4. Modeling Testing Effort

The most frequently used function to explain the testing effort are:

- Exponential
- Rayleigh
- Weibull
- Logistic
The first two can be derived from the assumption that, "the testing effort rate is proportional to the testing resources available".

\[ \frac{dW(t)}{dt} = \nu(t) \left[ \alpha - W(t) \right] \]  

(6)

Where \( \nu(t) \) is the time dependent rate at which testing resources are consumed, with respect to remaining available resources. Solving equation (6) under the initial condition \( W(0)=0 \), we get

**Case 1:**

When \( \nu(t)=\nu \), a constant:

\[ W(t) = \alpha \left( 1 - e^{-\nu t} \right) \]  

(7)

**Case 2:**

If \( \nu(t)=\nu t \), we get Rayleigh type curve:

\[ W(t) = \alpha \left( 1 - e^{-\nu \frac{t^2}{2}} \right) \]  

(8)

**Case 3:**

If \( \nu(t)=\nu t^l \), we get Weibull function:

\[ W(t) = \alpha \left( 1 - e^{-\nu t^l} \right) \]  

(9)

Exponential and Rayleigh curves become special cases of the Weibull curve for \( l=1 \) and \( l=2 \) respectively.

**Case 4:**

If we define

\[ \frac{dW(t)}{dt} = \nu \frac{W(t)}{\alpha} \left[ \alpha - W(t) \right] \]  

(10)

On solving, the cumulative testing effort consumed in the interval \((0, t)\) is given by

\[ W(t) = \frac{\alpha}{1 + l e^{-\nu t}}, \quad \text{where} \quad W(0) = \frac{\alpha}{1 + l} \]  

(11)

Where \( \alpha, \nu, \) and \( l \) are constants. This is the Logistic testing effort function.

To study the testing effort process, one of the above functions can be selected.

### 4.1. KG Model with Change Point Using Testing Effort Function

Most of the SRGMs proposed in the literature so far assume constant or monotonically increasing fault detection rate - Goel Okumoto Model [4]; Kapur et al [8]; Musa et. al. [11]. The failure intensity is assumed to be a continuous function of testing time. But in reality the fault detection rate can be affected by many factors such as the testing strategy and resource allocation. During a software testing process, it is quite possible that the fault detection rate function is changed at some point of time. Let this point of time be referred as 'change point'. The position of the change point can be judged from the graph of the actual failure data. Let the failure detection rate be defined as
\[ b(t) = \begin{cases} 
\frac{b_1}{1 + \beta e^{-b_1 W(t)}} & \text{when } 0 \leq t \leq \tau \\
\frac{b_2}{1 + \beta e^{-b_2 W(t)}} & \text{when } t > \tau 
\end{cases} \] (12)

where \( \tau \) is the change point.

Under the assumptions described above, the fault removal process can be described by the following differential equation:

\[ \frac{dm(t)}{dt} = b(t)(a - m(t)) \] (13)

Here \( w(t) \) is the current testing effort consumption at time \( t \).

**Case 1: For \( 0 \leq t \leq \tau \)**

\[ \frac{dm(t)}{dt} = \frac{b_1}{1 + \beta e^{-b_1 W(t)}} (a - m(t)) \] (14)

Solving the above differential equation with initial condition \( m(t=0)=0 \), we get:

\[ m(t)=a \left( \frac{1 - e^{-b_1 W'(t)}}{1 + \beta e^{-b_1 W(t)}} \right) \] (15)

where \( W'(t)=W(t)-W(0) \).

Here \( W(t)=\int_{0}^{t} w(x)dx \) is the cumulative testing effort function.

Here it can be noted that \( W(0) \) is equal to zero for all above specified Testing Effort function except for Logistic function.

**Case 2: For \( t > \tau \)**

\[ \frac{dm(t)}{dt} = \frac{b_2}{1 + \beta e^{-b_2 W(t)}} (a - m(t)) \] (16)

Solving the differential equation under initial condition \( m(t=\tau)=m(\tau) \), we get:

\[ m(t) = a \left[ 1 - \left( \frac{1 + \beta e^{-b_1 W'(0)}}{1 + \beta e^{-b_1 W'(t)}} \right) \left( \frac{1 + \beta e^{-b_2 W'(0)}}{1 + \beta e^{-b_2 W'(t)}} \right) e^{-\left( b_1 W'(t)-b_2 W'(t-\tau) \right)} \right] \] (17)

where \( W(t-\tau)=W(t)-W(\tau) \).
4.2. KG Model [7] With Change Point and Testing Effort Control

During testing, often the manager is not satisfied with the progress of the testing and the growth of the failure growth curve. Then there arises need for employing additional testing efforts in terms of new techniques, testing tools, more manpower so as to remove more faults than what could be possibly achieved with the current level of testing efforts in a pre specified time interval. In this paper we suggest a testing effort trade-off with respect to aspiration level for the removal process. This analysis gives an insight into the current level of progress in testing and later on helps in the estimation of extra efforts/cost required to achieve the aspiration level.

Let us consider the case when testing has been in progress for the time \( T_1 \) \((T_1 > \tau)\) and the number of faults removed by time \( T_1 \) is \( m(T_1) \). Let \( T_2 \) be the release time of the product in the market. Then by time \( T_2 \), the number will rise to \( m(T_2) \). But this is not the level \( m^* \) \((m^* > m(T_2))\), which the manager is aspiring for. To accelerate and to increase the efficiency of the testing he is willing to put in extra efforts in terms of additional man-hours, new testing techniques, tools and more skilled testing personnel. Now the question arises how much additional efforts above the current level need to be employed to achieve \( m^* \). The problem presented in this paper can be termed as the ‘Testing Effort Control Problem’ where for different aspiration levels; we try to estimate the requirement for additional efforts. It generates a trade off between \( m^* \) and effort requirement. The problem is addressed as follows:

First by using the failure data for time \((0, T_1)\), we estimate the parameters of the SRGM \((17)\). Using the estimated values of the parameters the number of faults, which can be removed, by time \( T_2 \) is \( m(T_2) \). If \( m^* < m(T_2) \), then the current level of testing is sufficient to reach the targeted reliability level. But if \( m^* > m(T_2) \), then there is the urgency of accelerating the fault removal rate by increasing efficiency of the testing. The aim is to estimate the requirement for the testing efforts for the time interval \((T_1, T_2)\) so as to remove \((m^* - m(T_1))\) faults by time \( T_2 \).

If the current level of testing efficiency is maintained then, the number of faults removed by time \( T_2 \) is given by:

\[
m(T_2) = a \left[ 1 - \left( \frac{1 + \beta e^{-h_1 W(0)}}{1 + \beta e^{-h_1 W(T_1)}} \right) \left( \frac{1 + \beta e^{-h_2 W(T_1)}}{1 + \beta e^{-h_2 W(T_2)}} \right) e^{-\left(h_1 W(0) + h_2 W(T_1) - h_2 W(T_2)\right)}}\right]
\]

\( (18) \)

If \( m^* < m(T_2) \), then there is nothing to worry. The team has to just sustain the current level and the product is expected to be ready for the delivery without any urgency.

Now let us consider the case when \( m^* > m(T_2) \).

Here \( m^* = m(T_1) + m'(T_2 - T_1) \)

Here \( m'(T_2 - T_1) \) is the additional number of faults that need to be removed to reach \( m^* \) by time \( T_2 \). Let \( W(T_1) \) be the cumulative level of the testing effort used for time \((0, T_1)\). Let \( W(T_2 - T_1) \) be the additional amount of efforts required to remove \( m'(T_2 - T_1) \) during interval \((T_1, T_2)\).

For \( t > T_1 \)

The removal process can be represented by the following differential equation:
\[
\frac{d m(t)}{dt} = \frac{b_2}{w(t)} \left( a - m(T_1) - m(t) \right) \tag{19}
\]

Let us define \(a_1 = a - m(T_1)\), then above differential equation can be written as:

\[
\frac{d m(t)}{dt} = \frac{b_2}{1 + \beta e^{-b_2 w(t)}} \left( a_1 - m(t) \right) \tag{20}
\]

Solving it with initial condition \(m(t=T_1)=0\) and \(W(t=T_1)=0\) we get:

\[
m(t) = m(T_1) + a_1 \left( 1 - e^{-b_2 W(t)} \right) \frac{1}{1 + \beta e^{-b_2 W(t)}} \tag{21}
\]

If the desirable level for the fault removal is \(m^*\), then the requirement for the additional efforts can be generated by the following expression:

\[
m^* = m(T_1) + a_1 \left( 1 - e^{-b_2 W^*} \right) \frac{1}{1 + \beta e^{-b_2 W^*}} \tag{22}
\]

With the estimated values of parameters \(a_1, a_2, \beta\) and \(m(T_1)\), above expression can be solved to find the value of \(W^*\) corresponding to different values of \(m^*\).

Here \(W^* = W(T_2) - W(T_1)\) \tag{23}

where \(W^*\) represents the amount of additional efforts required for the time interval \((T_1,T_2)\) to remove \(m^*\) faults from the software.

### 5. Parameter Estimation

Testing-effort data of a software can be collected in the form of testing-effort \(W_i(0<W_i< W_2<\ldots<W_k)\) consumed in test cases \((0,t_i]\) where \(i=1,2,\ldots,k\) in which \(m_i(0<m_1<m_2<\ldots<m_k)\) faults are detected. Then the parameters \((a>0, \ 0<\gamma<1, \ l\geq0)\) in the testing-effort functions are estimated by the method of least squares as follows

\[
\text{Minimize} \quad \sum_{i=1}^{k} \left[ W_i - \hat{W}_i \right]^2 \tag{24}
\]

Subject to \(\hat{W}_i = W_i\)

where \(\hat{W}_i = W_i\) implies that the estimated value of the testing-effort is equal to the actual value.

Using these estimated parameters values; we estimate the parameters in the proposed model by the method of MLE. The Likelihood function \(L\) for the unknown parameters with the mean value function \(m(t)\) is given as

\[
L(a,b_1,b_2,\beta \mid (W_i,x_i)) = \prod_{i=1}^{j} \frac{[m(t_i) - m(t_{i-1})]}{(x_i - x_{i-1})!} \exp \left[ -\left( m(t_i) - m(t_{i-1}) \right) \right] \tag{25}
\]
The MLE of the SRGM parameters can be obtained to by maximizing $L$ in (25) with respect to the following constraints: $(a > 0, 0 < b_1, 2 < 1, \beta \geq 0)$.

6. Model Validation and Comparison Criteria

6.1 Model Validation

To assess the performance of the proposed SRGM with change point and effort control, we have carried out the parameter estimation on two real software failure datasets. The first data set (DS-I) had been collected during 35 months of testing a radar system of size 124 KLOC and 1301 faults were detected during testing Brooks and Motley [1]. The second data set (DS-II) had been collected during 21 weeks of testing a real time command and control system and 136 faults were detected during testing Musa et. al. [11].

6.2 Comparison Criteria: Goodness of Fit Criteria

The Mean Square Fitting Error (MSE): Using the estimated values of the parameters, the estimated values for failure data are calculated. The difference between the estimated values, $m(t_i)$ and the observed values $y_i$ is measured by MSE as follows

$$MSE = \frac{1}{k} \sum_{i=1}^{k} \left( m(t_i) - y_i \right)^2$$  \hspace{1cm} (26)

Where $k$ is the number of observations. Lower value of MSE indicates less fitting error, thus better goodness of fit.

Coefficient of Multiple Determination ($R^2$): This measure can be used to investigate whether a significant trend exists in the observed failure intensity. This coefficient is defined as the ratio of the Sum of Squares (SS) resulting from the trend model to that from a constant model subtracted from 1, that is,

$$R^2 = 1 - \frac{\text{residual SS}}{\text{corrected SS}}$$  \hspace{1cm} (27)

$R^2$ measures the percentage of the total variation about the mean accounted for by the fitted curve. It ranges in value from 0 to 1. Small values indicate that the model does not fit the data well.

7. Data Analyses and Model Comparison

To check the accuracy of the proposed SRGM with respect to different type of testing effort function, we have first estimated the testing effort function using effort cumulative consumption data. Here we assume that location of change point $\tau$ can be judged from the data curve and hence need not be estimated. Using Non-linear regression technique remaining parameters are estimated. To judge the predictive power and accuracy of the proposed SRGM, we have compared the results to the those of Kapur and Garg model with testing effort without change point [7] and used MSE and $R^2$ as the performance measures.
7.1 KG Model [7] With Change Point

7.1.1 First Application (using DS-I)

The estimated values of parameters for the different effort function are provided in Table-1. Once the estimation for the effort function is over, parameters of mean value function \( m(t) \) given by (17) are worked out. For this purpose, we assume that change point \( \tau \) can be judged from the data and need not be estimated. To identify the location of change point, the graph of cumulated number of faults is evaluated and wherever a sudden change in detection rate is observed, the corresponding time point is termed as change point. For DS-I, the change is observed at around 17th value. So here we take \( \tau = 17 \). The estimation results are provided in Table-2. Here we observe that \( b_2 \) is less than \( b_1 \) irrespective of the effort function used. It shows the slow down in detection rate after the change point. Table 2 gives the estimated values for the parameters of the Kapur and Garg model with testing effort without change point and the proposed model with change point.

### Table-1: Estimation of Testing Effort Function Parameters

<table>
<thead>
<tr>
<th>Testing Effort Function</th>
<th>Parameter Estimation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( \alpha )</td>
</tr>
<tr>
<td>Exponential</td>
<td>1020677</td>
</tr>
<tr>
<td>Rayleigh</td>
<td>2873</td>
</tr>
<tr>
<td>Weibull</td>
<td>2670</td>
</tr>
<tr>
<td>Logistic</td>
<td>2067</td>
</tr>
</tbody>
</table>

### Table-2: Model Parameters Estimation Results

<table>
<thead>
<tr>
<th>Models under Comparison</th>
<th>Parameter Estimation</th>
<th>Type of Testing effort Function</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Exponential</td>
<td>Rayleigh</td>
</tr>
<tr>
<td></td>
<td>( a )</td>
<td>1331</td>
</tr>
<tr>
<td></td>
<td>( b_1 )</td>
<td>.0042</td>
</tr>
<tr>
<td></td>
<td>( \beta )</td>
<td>20.19</td>
</tr>
<tr>
<td></td>
<td>( R^2 )</td>
<td>.99904</td>
</tr>
<tr>
<td></td>
<td>MSE</td>
<td>204</td>
</tr>
<tr>
<td>Kapur &amp; Garg Model without change point</td>
<td>( a )</td>
<td>1322</td>
</tr>
<tr>
<td></td>
<td>( b_1 )</td>
<td>.004458</td>
</tr>
<tr>
<td></td>
<td>( b_2 )</td>
<td>.004418</td>
</tr>
<tr>
<td></td>
<td>( \beta )</td>
<td>25.3774</td>
</tr>
<tr>
<td></td>
<td>( R^2 )</td>
<td>.99930</td>
</tr>
<tr>
<td></td>
<td>MSE</td>
<td>149.65</td>
</tr>
</tbody>
</table>

The fitting of the proposed model with change point to DS-I is graphically illustrated in Figures 1, 2, 3 and 4. It clearly shows that the model fits the data excellently.

7.1.2 Second Application (using DS-II)

The estimated values of parameters for the different effort function are provided in Table-3. After the estimation of the effort function, parameters of mean value function \( m(t) \) given by (17) are worked out. For this purpose, first the position of change point \( \tau \) is judged from the data. To identify \( \tau \), the graph of actual cumulated no. of faults is drawn
and we look for the point where a sudden change in detection rate is observed. For DS – II, which consists of the 136 no. of faults removed over a period of 21 weeks, change point is observed at around 8th week. So here we take $\tau=8$. The estimation results are listed in Table-4. Here we observe that $b_2$ is less than $b_1$ for different types of effort functions except for the Logistic Effort Function. Table 4 gives the estimated values for the parameters of the Kapur and Garg model with testing effort without change point and the proposed model with change point.

<table>
<thead>
<tr>
<th>Testing Effort Function</th>
<th>Parameter Estimation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$\alpha$</td>
</tr>
<tr>
<td>Exponential</td>
<td>35395</td>
</tr>
<tr>
<td>Rayleigh</td>
<td>22128</td>
</tr>
<tr>
<td>Weibull</td>
<td>26.78</td>
</tr>
<tr>
<td>Logistic</td>
<td>45.15</td>
</tr>
</tbody>
</table>
Table 4: Model Parameters Estimation Results

<table>
<thead>
<tr>
<th>Models under Comparison</th>
<th>Parameter Estimation</th>
<th>Type of Testing effort Function</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Exponential</td>
</tr>
<tr>
<td>Kapur &amp; Garg Model</td>
<td></td>
<td></td>
</tr>
<tr>
<td>without change point</td>
<td>$a$</td>
<td>151.01</td>
</tr>
<tr>
<td></td>
<td>$b$</td>
<td>0.5693</td>
</tr>
<tr>
<td></td>
<td>$\beta$</td>
<td>583.37</td>
</tr>
<tr>
<td></td>
<td>$R^2$</td>
<td>.99716</td>
</tr>
<tr>
<td></td>
<td>MSE</td>
<td>6.73</td>
</tr>
<tr>
<td>Kapur &amp; Garg Model</td>
<td>$a$</td>
<td>148</td>
</tr>
<tr>
<td>with change point</td>
<td>$b_1$</td>
<td>.654857</td>
</tr>
<tr>
<td></td>
<td>$b_2$</td>
<td>.603865</td>
</tr>
<tr>
<td></td>
<td>$\beta$</td>
<td>852.04</td>
</tr>
<tr>
<td></td>
<td>$R^2$</td>
<td>.99761</td>
</tr>
<tr>
<td></td>
<td>MSE</td>
<td>5.67</td>
</tr>
</tbody>
</table>

Fig. 5  Goodness-of-Fit for DS-II (using Exponential curve)  
Fig. 6  Goodness-of-Fit for DS-II (using Rayleigh curve)  
Fig. 7  Goodness-of-Fit for DS-II (using Weibull curve)  
Fig. 8  Goodness-of-Fit for DS-II (using Logistic curve)
The fitting of the proposed model with change point to DS-II is graphically illustrated in Figures 5, 6, 7 and 8. It is clear that the model fits the data excellently.

For both DS-I and DS-II the lower MSE and higher $R^2$ with respect to different types of effort function shows the higher achieved accuracy when change point is introduced in the flexible model.


Let us consider the case when testing is in progress for time $T_1$. Let the release time be $T_2$. To estimate the additional requirement for the resources so as to achieve the pre-set reliability objective, first we consider the failure data for the time $(0,T_1)$ and estimate the testing effort function using effort cumulative consumption data. Here location of \( \tau \) is taken same, as we have done in section 7.1. Using Non-linear regression technique remaining parameters are estimated.

After the estimation for the parameters is done by using failure data for the interval $(0,T_1)$, the expected number of faults for the interval $(T_1, T_2)$ is calculated. The testing effort function, which yields the values nearest to the actual values, is selected for generating requirement for additional testing resources to meet the desired reliability level.

7.2.1 First Application (Using DS-I)

For this data set we have fixed $T_1$=30. First 30 data values are used for estimation purpose. The estimated values of parameters for the different effort function are provided in Table-5. Once the estimation for the effort function is over, parameters of mean value function $m(t)$ given by (17) are worked out. Here we have taken $\tau=17$. The estimation results are provided in Table-6.

<table>
<thead>
<tr>
<th>Testing Effort Function</th>
<th>Parameter Estimation</th>
<th>( \alpha )</th>
<th>( \gamma )</th>
<th>( l )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exponential</td>
<td>962052.95999</td>
<td>0.000046179</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Rayleigh</td>
<td>3245.4306271</td>
<td>.000744455</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Weibull</td>
<td>3766.7065429</td>
<td>0.000751515</td>
<td>1.938471936</td>
<td></td>
</tr>
<tr>
<td>Logistic</td>
<td>1894.4599153</td>
<td>.172649465</td>
<td>40.3011508</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Models under Comparison</th>
<th>Parameter Estimation</th>
<th>Type of Testing effort Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kapur &amp; Garg Model with change point</td>
<td>( a )</td>
<td>1330</td>
</tr>
<tr>
<td></td>
<td>( b_1 )</td>
<td>.004742</td>
</tr>
<tr>
<td></td>
<td>( b_2 )</td>
<td>.0047027</td>
</tr>
<tr>
<td></td>
<td>( \beta )</td>
<td>24.38</td>
</tr>
<tr>
<td></td>
<td>$R^2$</td>
<td>.99912</td>
</tr>
<tr>
<td></td>
<td>MSE</td>
<td>144.86</td>
</tr>
</tbody>
</table>

Suppose it is decided that the testing is to be terminated after 35 weeks i.e. $T_2=35$. For further analysis, Rayleigh is selected to model testing effort function on the basis of
criterion specified above. Now using the expression for \( m(t) \) given by (17), the number of faults expected to be removed by time \( T_2 \) is calculated. Using the estimated values of parameters in expression (17), \( m(T_2) = 1302 \) while \( m(T_1) = 1267 \). So if the same efforts are continued, then no. of faults removed during (30, 35) is 35. But if the management is aiming for this number to be higher than 35 then extra efforts need to be put in. To generate the requirement for extra efforts \( W^\# \) with respect to the desired level \( m^\* \), expression (22) is used. The estimated values of \( W^\# \) with respect to different levels \( m^\* \) are provided in Table-7.

### Table-7: Requirement of Additional Testing Effort

<table>
<thead>
<tr>
<th>( M^* )</th>
<th>1303</th>
<th>1304</th>
<th>1305</th>
<th>1306</th>
<th>1307</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Faults to be removed in ( (T_1,T_2) )</td>
<td>36</td>
<td>37</td>
<td>38</td>
<td>39</td>
<td>40</td>
</tr>
<tr>
<td>Additional Testing Effort required ( W^# )</td>
<td>365.2278</td>
<td>382.6004</td>
<td>400.8106</td>
<td>419.9444</td>
<td>440.1016</td>
</tr>
</tbody>
</table>

#### 7.2.2 Second Application (Using DS-II)

For DS-II, \( T_1=19 \). Initial 19 data values are used for estimation of parameters for effort function as well as \( m(t) \). The estimated values of parameters for the different effort function are provided in Table-8. After the estimation of the effort function, parameters of mean value function \( m(t) \) given by (17) are worked out. For this purpose, we take \( \tau=8 \). The results are listed in Table-9.

### Table-8: Testing Effort Function Estimation

<table>
<thead>
<tr>
<th>Testing Effort Function</th>
<th>Parameter Estimation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( \alpha )</td>
</tr>
<tr>
<td>Exponential</td>
<td>24072.764</td>
</tr>
<tr>
<td>Rayleigh</td>
<td>15329.827894</td>
</tr>
<tr>
<td>Weibull</td>
<td>28.553056124</td>
</tr>
<tr>
<td>Logistic</td>
<td>42.374147139</td>
</tr>
</tbody>
</table>

### Table-9: Model Parameters Estimation Results

<table>
<thead>
<tr>
<th>Models under Comparison</th>
<th>Parameter Estimation</th>
<th>Type of Testing effort Function</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( a )</td>
<td>( b_1 )</td>
</tr>
<tr>
<td>Kapur &amp; Garg Model with change point</td>
<td>162</td>
<td>.74292</td>
</tr>
<tr>
<td></td>
<td>147</td>
<td>.30185</td>
</tr>
<tr>
<td></td>
<td>155</td>
<td>.580.65</td>
</tr>
<tr>
<td></td>
<td>147</td>
<td>.000000001</td>
</tr>
</tbody>
</table>

Let \( T_2=21 \). For this data set, Weibull has provided values nearest to actual values for the interval \( (T_1, T_2) \). Using the estimated values of parameters in expression (17), \( m(T_2) = 135 \) while \( m(T_1) = 125 \). So if the same efforts are continued, then number of faults removed during (19,21) is 10. But if the management is aiming for higher level of removal, then extra efforts need to be put in. To generate the requirement for extra efforts
with respect to the desired level \( m^* \), expression (22) is used. The estimated values of \( W^\# \) with respect to different levels \( m^* \) for DS-II are provided in Table-10.

Table-10: Requirement of Additional Testing Effort

<table>
<thead>
<tr>
<th>( m^* )</th>
<th>136</th>
<th>137</th>
<th>138</th>
<th>139</th>
<th>140</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Faults to be removed in ( (T_1, T_2) )</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
</tr>
<tr>
<td>Additional Testing Effort required ( W^# )</td>
<td>5.8099</td>
<td>6.5155</td>
<td>7.2613</td>
<td>8.0525</td>
<td>8.8947</td>
</tr>
</tbody>
</table>

8. Extended Change Point Model (Scope For Future Work)

If the software being developed is big and testing is supposed to continue for considerably large interval of time then it is quite possible that change in FDR is observed more than once. The frequent changes in testing team, test cases or the management can alter the overall testing growth resulting in changes in removal rate.

Now we extend the proposed model to account for more than one change point.

Let us first describe the most generalized case when there are ‘\( n \)’ numbers of change points.

In such a case the fault detection rate can be described as:

\[
b(t) = \begin{cases} 
  \frac{b_1}{1 + \beta e^{-b_1 W(t)}} & 0 \leq t \leq \tau_1 \\
  \frac{b_2}{1 + \beta e^{-b_2 W(t)}} & \tau_1 < t \leq \tau_2 \\
  \vdots \\
  \frac{b_{n+1}}{1 + \beta e^{-b_{n+1} W(t)}} & t > \tau_n 
\end{cases}
\]

Here \( \tau_1, \tau_2, \ldots, \tau_n \) are \( n \) change points.

Then we have

Case 1: For \( 0 \leq t \leq \tau_1 \)

Solving the differential equation (13) with initial condition \( m(t=0)=0 \), we get:

\[
m(t) = a \left( \frac{1 - e^{-b_1 W^\#(t)}}{1 + \beta e^{-b_1 W(t)}} \right)
\]

where \( W^\#(t)=W(t)\) - \( W(0) \).

Case 2: For \( \tau_1 < t \leq \tau_2 \)

Solving the differential equation (13) under initial condition \( m(t=\tau_1)=m(\tau_1) \) we get:
\[
m(t) = a \left[ 1 - \left( \frac{1 + \beta e^{-hW(0)}}{1 + \beta e^{-1W(1)}} \right) \left( \frac{1 + \beta e^{-2W(1)}}{1 + \beta e^{-3W(1)}} \right) e^{-\theta t} \right] \tag{30}
\]

**Case n+1:** For \( t > \tau_n \)

Solving with initial condition \( m(t=\tau_n) = m(\tau_n) \), we get:

\[
m(t) = a - a \times \left( \frac{1 + \beta e^{-hW(0)}}{1 + \beta e^{-1W(1)}} \right) \left( \frac{1 + \beta e^{-2W(1)}}{1 + \beta e^{-3W(1)}} \right) e^{-\theta t} \exp \left( -(b_1 W^*(\tau_1) + b_2 W(t-\tau_n) + \sum_{j=2}^{n} b_j W(\tau_j - \tau_{j-1})) \right) \tag{31}
\]

**8.1. Particular Cases**

We have already derived the case when there is one change point. Now consider the case when \( n=2 \) (i.e., 2 change points are observed in the data)

Then mean value function is given by:

\[
m(t) = a - a \times \left( \frac{1 + \beta e^{-hW(0)}}{1 + \beta e^{-1W(1)}} \right) \left( \frac{1 + \beta e^{-2W(1)}}{1 + \beta e^{-3W(1)}} \right) e^{-\theta t} \exp \left( -(b_1 W^*(\tau_1) + b_2 W(\tau_2 - \tau_1) + b_3 W(t-\tau_2)) \right) \tag{32}
\]

In this paper we have included the estimation results for the \( n=1 \). The estimation for \( n=2 \) and higher values is being worked out and will be brought out in a future paper.

**9. Conclusion**

In this paper a flexible class model due to Kapur and Garg [7] with testing effort function has been assessed with the change point. The incorporation of ‘change point’ in this model helps in achieving a considerable improvement in terms of lower MSE and higher \( R^2 \). The introduction of change factor in FDR helps in better predictability and more accuracy. In the present paper we have limited ourselves to introduction of just one change point. But an insight has been provided to include more than one change point, which seems quite near to real SDLC. The study of change point is not limited to the area of Software Reliability but it can be extended to other application-oriented fields like Distributed Software systems, Hardware Reliability or Marketing areas. Our future work deals with application of this concept in area of Marketing. We extend our model to provide a trade-off analysis between additional testing effort requirement and aspiration level for number of faults removed. The analysis discussed here, also known as testing effort control problem is quite useful tool for the manager to plan and work out the details of the requirement for testing resources so that the product is ready for the release by its due date. It helps the organization to beat the intense competition by launching the reliable and good quality product well on time and thus retaining the present clientele and building future ones.
References


