International Journal of Security and Its Applications
Vol.9, No.12 (2015), pp.21-28
http://dx.doi.org/10.14257/ijsia.2015.9.12.03

# Finite Server Queuing Model with Change-Point and Imperfect Debugging for Software Reliability Assessment

Zhang Nan

*Harbin University of Commerce, Harbin 150028, China*
*zhangnan@hrbcu.edu.cn*

***Abstract***

*Most software reliability growth models (SRGMs) take software faults detected that may be removed immediately into consideration. In fact, it costs often considerable resources to identify the root causes of faults detected and to remove them. In additional, detection effort and correction effort have the great effect on software debugging process. The detection effort function and correction effort function are defined as resource expenditures spent on software debugging process, respectively. In this paper, we investigate software debugging process using the queue theory. We propose finite server queuing models with resource and change-point under imperfect debugging environment. A real software failure project is demonstrated the effectiveness of proposed models, and numerical results demonstrate that new models can provide better fit and prediction.*

***Keywords:*** *Software reliability; Change-point; Imperfect debugging; Detection effort; Correction effort*

Notation

$a$ the expected number of initial faults
$b$ the rate of fault detection per unit detection effort
$m_d(t)$ the mean number of software faults detected in time $(0,t]$
$m_c(t)$ the mean number of software faults corrected in time $(0,t]$
$w(t)$ the testing effort at time $t$
$w_d(t)$ the detection effort at time $t$
$w_c(t)$ the correction effort at time $t$
$W_d(t)$ the cumulative detection effort consumption in time $(0,t]$
$W_c(t)$ the cumulative correction effort consumption in time $(0,t]$
$\zeta(t)$ a time-dependent scale function
$\zeta$ a constant scale parameter
$p(0,\tau_1]$ the probability that a fault detected at time $x_1$ is fully removed in time $(x_1,\tau_1]$
$q(\tau_1,t]$ the probability that a fault detected at time $x_2$ is removed in time $(x_2,t]$
$T_d$ fault detected time
$T_c$ fault removed time
$G_1(\tau_1-y_1)$ the cumulative distribution function of the fault correction time $(0,\tau_1]$
$F_1(y_1-x_1)$ the fault waiting time over the time interval $(0,\tau_1]$
$G_2(t-\tau_1)$ the cumulative distribution function of the fault correction in $(0,\tau_1]$
$F_2(y_2-x_2)$ the cumulative distribution function of the fault waiting time in $(\tau_1,t]$

## 1. Introduction

To estimate and predict software reliability, one of the critical techniques is to use the SRGMs [1]. Using these SRGMs, software developers or managers can derive

ISSN: 1738-9976 IJSIA

quantitative information about whether reliability goals have been achieved, and determine the resources allocation and release time needed to achieve desired reliability requirements [2].

Recently, researchers have used the queue theory to describe software debugging process behavior [3-5]. However, some issues are not involved in model, such as resource expenditures spent on software debugging process, change-point, and imperfect debugging environment. During software debugging process, software reliability is very depending on the amounts of resources [6]. Further, the rate of fault remove may be not constant or change as time, and new faults can be introduced as part of software debugging process.

Based on foregoing discussion, we will propose finite server queuing models with resource, change-point and imperfect debugging. The new models could determine resources and manpower needed to achieve desired reliability requirement. Then, software programmers estimate and predict software reliability using these models. In proposed models, software debugging processes are assumed to be the Non-Homogeneous Poisson Process (NHPP) and a general distribution process, respectively. Detection effort and correct effort represent resource expenditures spent on software debugging process, respectively. Besides, new faults are created when removing faults. This paper is organized as follows. The related work is summarized in Section 2, Section 3 will study how to derive proposed models. Section 4 gives the numerical example with the actual software failure data. Conclusions are drawn in Section 5.

## 2. Related Work

Detection effort function and correction effort function are considered as the testing effort consumption spent on software debugging processes, respectively. Furthermore, the actual testing effort expenditures can be described by distribution curves. We assume

$$w_d(t) = \zeta w(t) \tag{1}$$

$$w_c(t) = (1-\zeta) w(t) \tag{2}$$

In real world, the rate of fault remove relies on the capability of software programmers, the difficulty of faults removed, debugging resource, debugging environment and so on. Once these factors are change during the fault debugging process, software fault remove rate may be change. Thus, there is considered as a change-point problem. Many SRGMs with change-point have achieved a great improvement in the accuracy of estimation and prediction of software reliability [4, 7].

On the other hand, software fault remove process is controlled by debugging programmers, however faults detected are not always fully removed, and new faults can be introduced as part of the fault remove process. Many SRGMs with imperfect debugging have proposed a relaxation of perfect debugging assumption, and these models have a fairly accurate predication capability [8-9, 11].

## 3. Model Formulation

The proposed model is based on the assumptions [3-5]:

1. Software debugging processes are finite server queuing models with NHPP arrival and general service time distribution
2. In$(t, t+\Delta t]$, the mean number of faults detected by the current detection effort expenditures is proportional to the mean number of remaining faults
3. The fault remove time is considered and the number of faults removed delays total number of faults detected.

 

4. The time-dependent behavior of detection effort and of correction effort can be defined by distribution curves.
5. The new models are imperfect debugging.

According to the above assumption, we get the differential equation [6, 10]

$$\frac{dm_d(t)}{dt} = bw_d(t)[a(t) - m_d(t)] \tag{3}$$

Here, we supposed $a(t) = a + km_d(t)$. Solving Eq.(3) yields

$$m_d(t) = \frac{a}{1-k}[1 - \exp(-b(1-k)(W_d(t) - W_d(0)))] \tag{4}$$

Let $N_c(t)$ be the number of fully removed in time$(0, t]$. If there is one change-point in fault remove process, $m_c(t)$ is given by

$$m_c(t) = m_c(0, \tau_1] + m_c(\tau_1, t] \tag{5}$$

In time $(0, \tau_1]$, $m_c(0, \tau_1]$ is computed by

$$m_c(0, \tau_1] = m_d(0, \tau_1]p(0, \tau_1] \tag{6}$$

In real software project, there is finite software resource to remove faults. Then $p(0, \tau_1]$ is composed of two parts

$$p(0,\tau_1] = \int_0^{\tau_1} P\{(T_d = x_1) \cap (T_c \le \tau_1 - x_1)\}\mathrm{d}x_1 + \int_0^{\tau_1}\int_{x_1}^{\tau_1} P\{(T_d = x_1) \cap (T_q \le y_1 - x_1) \cap (T_c \le \tau_1 - y_1)\}\mathrm{d}y_1\mathrm{d}x_1$$

$$= \int_0^{\tau_1} \frac{m_d'(x_1)}{m_d(\tau_1)} G_1(\tau_1 - x_1)\mathrm{d}x_1 + \int_0^{\tau_1}\int_{x_1}^{\tau_1} \frac{m_d'(x_1)}{m_d(\tau_1)} F_1(y_1 - x_1)G_1(\tau_1 - y_1)\mathrm{d}y_1\mathrm{d}x_1 \tag{7}$$

Substituting Eq.(7) into Eq.(8), we have

$$m_c(0, \tau_1] = \int_0^{\tau_1} m_d'(x_1)G_1(\tau_1 - x_1)\mathrm{d}x_1 + \int_0^{\tau_1}\int_{x_1}^{\tau_1} m_d'(x_1)F_1(y_1 - x_1)G_1(\tau_1 - x_1)\mathrm{d}y_1x_1 \tag{8}$$

In time $(\tau_1, t]$, the cumulative fault remove counting consists two parts: faults detected in $(0, \tau_1]$ are fully removed in $(\tau_1, t]$; anther, faults detected in $(\tau_1, t]$ are fully removed in $(\tau_1, t]$. Due to the finite removed resource, $q(\tau_1, t]$ is also composed of two parts

$$q(\tau_1, t] = \int_{\tau_1}^{t} P\{(T_d = x_2) \cap (T_c \le t - x_2)\}\mathrm{d}x_2 + \int_{\tau_1}^{t}\int_{x_2}^{t} P\{(T_d = x_2) \cap (T_q \le y_2 - x_2) \cap (T_c \le t - y_2)\}\mathrm{d}y_2\mathrm{d}x_2$$

$$= \int_{\tau_1}^{t} \frac{m_d'(x_2)}{m_d(t) - m_d(\tau_1)} G_2(t - x_2)\mathrm{d}x_2 + \int_{\tau_1}^{t}\int_{x_2}^{t} \frac{m_d'(x_2)}{m_d(t) - m_d(\tau_1)} F_2(y_2 - x_2)G_2(t - y_2)\mathrm{d}y_2\mathrm{d}x_2 \tag{9}$$

Substituting $q(\tau_1, t]$ into $m_d(\tau_1, t]q(\tau_1, t]$ yields

$$m_d(\tau_1,t]q(\tau_1,t] = [m_d(t) - m_d(\tau_1)]\left[\int_{\tau_1}^{t} \frac{m_d'(x_2)}{m_d(t) - m_d(\tau_1)} G_2(t - x_2)\mathrm{d}x_2 + \int_{\tau_1}^{t}\int_{x_2}^{t} \frac{m_d'(x_2)}{m_d(t) - m_d(\tau_1)} F_2(y_2 - x_2)G_2(t - y_2)\mathrm{d}y_2\mathrm{d}x_2\right]$$

$$= \int_{\tau_1}^{t} m_d'(x_2)G_2(t - x_2)\mathrm{d}x_2 + \int_{\tau_1}^{t}\int_{x_2}^{t} m_d'(x_2)F_2(y_2 - x_2)G_2(t - y_2)\mathrm{d}y_2\mathrm{d}x_2 \tag{10}$$

$m_c(\tau_1, t]$ is defined as

$$m_c(\tau_1, t] = [m_d(\tau_1) - m_c(\tau_1)]G_2(t - \tau_1) + m_d(\tau_1, t]q(\tau_1, t]$$

$$= [m_d(\tau_1) - m_c(\tau_1)]G_2(t - \tau_1) + \int_{\tau_1}^{t} m_d'(x_2)G_2(t - x_2)\mathrm{d}x_2 + \int_{\tau_1}^{t}\int_{x_2}^{t} m_d'(x_2)F_2(y_2 - x_2)G_2(t - y_2)\mathrm{d}y_2\mathrm{d}x_2 \tag{11}$$

From Eq.(5), Eq.(6) and Eq.(11), we can obtain

$$m_c(t) = \int_0^{\tau_1} m_d'(x_1)G_1(\tau_1 - x_1)\mathrm{d}x_1$$
$$+ \int_0^{\tau_1}\int_{x_1}^{\tau_1} m_d'(x_1)F_1(y_1 - x_1)G_1(\tau_1 - y_1)\mathrm{d}y_1\mathrm{d}x_1 + [m_d(\tau_1) - m_c(\tau_1)]G_2(t - \tau_1)$$
$$+ \int_{\tau_1}^{t} m_d'(x_2)G_2(t - x_2)\mathrm{d}x_2 + \int_{\tau_1}^{t}\int_{x_2}^{t} m_d'(x_2)F_2(y_2 - x_2)G_2(t - y_2)\mathrm{d}y_2\mathrm{d}x_2 \quad (12)$$

The paper assumes $G_{k+1}(x_{k+1}) = 1 - \exp[-\rho_{k+1}W_c(\tau_{k+1}) + \rho_{k+1}W_c(\tau_{k+1} - x_{k+1})]$, $F_{k+1}(x_{k+1}) = 1 - \exp[-\mu_{k+1}W_c(y_{k+1}) + \mu_{k+1}W_c(y_{k+1} - x_{k+1})]$. Substitute them into Equation (12).

## 4. Numerical Example

### 4.1. Data Description and Comparison Criteria

The real software project detected and removed 136 software faults in 21 weeks' testing and 25.3 CPU hours. Some criteria are presented to decide the performance of software reliability [2].

$$\mathrm{SSE} = \sum_{i=1}^{n}[m_i - \hat{m}(t_i)]^2 \quad (13)$$

$$RMSE = \sqrt{\sum_{i=1}^{n}\frac{[m_i - \hat{m}(t_i)]^2}{n}} \quad (14)$$

The Laplace factor is given by [2]

$$u(j) = \frac{\sum_{i=1}^{j}(i-1)n(i) - \frac{(j-1)}{2}\sum_{i=1}^{j}n(i)}{\sqrt{\frac{(j^2-1)}{12}\sum_{i=1}^{j}n(i)}} \quad (15)$$

The C chart is tool that is used to monitor the software process. The C chart is considered as Euation. (16), Equation (17) and Equation (18)

$$\overline{C} = \frac{\sum_{i=1}^{k}n(i)}{k} \quad (16)$$

$$UCL = \overline{C} + 3\sqrt{\overline{C}} \quad (17)$$

$$LCL = \overline{C} - 3\sqrt{\overline{C}} \quad (18)$$

This paper, the Laplace trend test and the C chart results are shown in Figure1 and Figure 2. As seen from Figure 1 and Figure 2, this may be two change-points. However, researchers suggest that the numbers of change-points are not too many. Thus, this paper chooses one change-point.

 

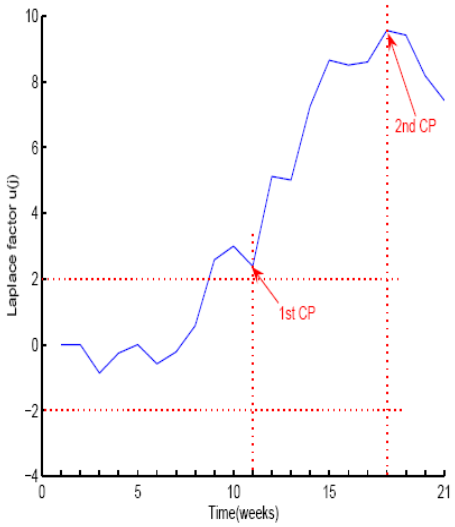


**Figure 1. Laplace Trend Analysis**

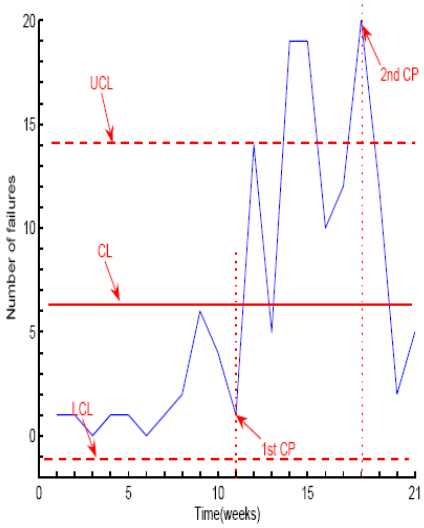


**Figure 2. C Chart**

### 4.2. Performance Analysis

We will analyze and discuss the performance of new models according to the actual software failure data in this section. The formulations of some models are listed in Table 1. Here, the results of the SSE and the RMSE are shown in Table 2. From Table 2, we can see the new models provide the lower value of $SSE_d$ and $SSE_c$, the higher value of $RMSE_d$ and $RMSE_c$ than other SRGMs. Therefore, the new model is suitable for modeling the software reliability and the fitted distribution curves are highly significant for the actual software failure data set.

**Table 1. Mean Value Functions of Comparison of Goodness-of-fit of SRGMs**

| No. | Model |
|---|---|
| 1 | $W_d(t) = \zeta\alpha\left[1-\exp(-\beta t^m\exp(-\lambda t))\right]^\theta$<br>$W_c(t) = (1-\zeta)\alpha\left[1-\exp(-\beta t^m\exp(-\lambda t))\right]^\theta$<br>$m_d(t) = \frac{a}{1-k}[1-\exp(-b(1-k)W_d^\star(t))]$<br>$m_c(t) = \int_0^{\tau_1} m_d'(x_1)[1-\exp(-\rho_1 W_c(\tau_1)+\rho_1 W_c(x_1))]\mathrm{d}x_1$<br>$+\int_0^{\tau_1}\int_{x_1}^{\tau_1} m_d'(x_1)[1-\exp(-\rho_1 W_c(\tau_1)+\rho_1 W_c(y_1))]$<br>$[1-\exp(-\mu_1 W_c(y_1)+\mu_1 W_c(x_1))]\mathrm{d}y_1\mathrm{d}x_1$<br>$+\int_{\tau_1}^{t} m_d'(x_2)[1-\exp(-\rho_2 W_c(t)+\rho_2 W_c(x_2))]\mathrm{d}x_2$<br>$+\int_{\tau_1}^{t}\int_{x_2}^{t} m_d'(x_2)[1-\exp(-\rho_2 W_c(t)+\rho_2 W_c(y_2))]$<br>$[1-\exp(-\mu_2 W_c(y_2)+\mu_2 W_c(x_2))]\mathrm{d}y_2\mathrm{d}x_2$<br>$+[m_d(\tau_1)-m_c(\tau_1)][1-\exp(-\rho_1 W_c(t)+\rho_1 W_c(\tau_1))]$ |
| 2 | $W_d(t) = \zeta\alpha\left[1-\exp(-\beta t^m\exp(-\lambda t))\right]^\theta$<br>$W_c(t) = (1-\zeta)\alpha\left[1-\exp(-\beta t^m\exp(-\lambda t))\right]^\theta$<br>$m_d(t) = a[1-\exp(-bW_d^\star(t))]$<br>$m_c(t) = \int_0^{\tau_1} m_d'(x_1)[1-\exp(-\rho_1 W_c(\tau_1)+\rho_1 W_c(x_1))]\mathrm{d}x_1$<br>$+\int_0^{\tau_1}\int_{x_1}^{\tau_1} m_d'(x_1)[1-\exp(-\rho_1 W_c(\tau_1)+\rho_1 W_c(y_1))]$<br>$[1-\exp(-\mu_1 W_c(y_1)+\mu_1 W_c(x_1))]\mathrm{d}y_1\mathrm{d}x_1$<br>$+\int_{\tau_1}^{t} m_d'(x_2)[1-\exp(-\rho_2 W_c(t)+\rho_2 W_c(x_2))]\mathrm{d}x_2$<br>$+\int_{\tau_1}^{t}\int_{x_2}^{t} m_d'(x_2)[1-\exp(-\rho_2 W_c(t)+\rho_2 W_c(y_2))]$<br>$[1-\exp(-\mu_2 W_c(y_2)+\mu_2 W_c(x_2))]\mathrm{d}y_2\mathrm{d}x_2$<br>$+[m_d(\tau_1)-m_c(\tau_1)][1-\exp(-\rho_1 W_c(t)+\rho_1 W_c(\tau_1))]$ |
| 3 | $W(t) = \alpha\left[1-\exp(-\beta t^m\exp(-\lambda t))\right]^\theta$<br>$m_d(t) = \frac{ab}{b+k}[\exp(kW^\star(t))-\exp(-bW^\star(t))]$<br>$m_c(t) = \frac{ab}{(b+k)^2}[b\exp(kW^\star(t))+k\exp(-bW^\star(t))]$<br>$-\frac{ab}{(b+k)}(1+bW^\star(t))\exp(-bW^\star(t))$ |

**Table 2. Comparison Results for Different SRGMs based on System T1**

| Mo | $SSE_d$ | $SSE_c$ | $RMSE_d$ | $RMSE_c$ |
|---|---|---|---|---|
| 1 | 843.6 | 589.7 | 7.602 | 5.426 |
| 2 | 1075 | 610.7 | 7.729 | 5.691 |
| 3 | 1052 | 1050 | 7.692 | 7.692 |

## 5. Conclusions

We have applied the queue theory to model software debugging process. Finite server queuing models with detection effort functions, correction effort functions, and change-point under imperfect debugging are proposed to estimate and predict software reliability in this paper. Compared with the existing other models, the new queuing models can give a better result of goodness-of-fit for this software failure data set.

## Acknowledgements

This paper is supported by the Harbin University of Commerce under Grant (No. 15RW21).

 

## References

[1] J. D. Musa, A. Jannino, and K. Okumoto, "Software Reliability, Measuremen, Prediction, Application", (McGram Hill, New York, 1987).
[2] M. R. Lyu, "Handbook of Software Reliability Engineering" (McGram Hill, New York, 1996).
[3] C. Y. Huang and W. C. Huang, "Software reliability analysis and measurement using finite and infinite server queuing models", IEEE Trans. on Reliability, no. 57, vol. 1, **(2008)**, pp. 192-203.
[4] C. Y. Huang and T. Y. Huang, Software reliability analysis and assessment using queuing models with multiple change-points, Computers and Mathematics with Applications, no. 60, vol. 7, **(2010)**, pp. 2015-2030.
[5] P. K. Kapur, S. Anada, S. Inoue, and S.Yamada, "A unified approach for developing software reliability growth model using infinite server queuing model", International J. of Reliability Quality Safety Engineer, vol. 17, no. 5, **(2010)**, pp. 401-424.
[6] S. Yamada, J. Hishitani, and S. Osaki, "Software reliability growth with a weibull test-effort a model and application", IEEE Trans. on Reliability, vol. 42, no. 1, **(1993)**, pp. 100-105.
[7] C. Y. Huang and M. R. Lyu, "Estimation and analysis of some generalized multiple change-point software reliability models", IEEE Trans. on Reliability, vol. 60, no. 2, **(2011)**, pp. 498-514.
[8] P. K. Kapur, H. Pham, S. Anand, and K. Yadav, "A unified approach for developing software reliability growth models in the presence of imperfect debugging and error generation", IEEE Trans. on Reliability vol. 60, no. 1, **(2011)**, pp. 331-340.
[9] C. T. Lin, "Analyzing the effect of imperfect debugging on software fault detection and correction processes via a simulation framework", Mathematical and Computer Modeling 54 **(2011)**, pp. 3046-3064.
[10] R. Peng, Q. P. Hu, S. H. Ng, and M. Xie, "Testing effort dependent software FDP and FCP models with consideration of imperfect debugging, in Pro. 4th IEEE International Conf. on Secure Software Integration and Reliability Improvement", **(2010)**.
[11] D. Wenfei, G. Jinaying, "The Study on the Optimization of Software Reliability Prediction Model", Journal of Harbin University of Science and Technology, vol. 14, no. 4, **(2009)**, pp. 67-70.

 