4th WSEAS Int. Conf. on COMPUTATIONAL INTELLIGENCE, MAN-MACHINE SYSTEMS and CYBERNETICS Miami, Florida, USA, November 17-19, 2005 (pp150-154)

Research on software reliability considering testing profile and operational profile

ZHAO Jing LIU Hong-Wei CUI Gang YANG Xiao-Zong
School of Computer Science and Technology
Harbin Institute of Technology
92 West DaZhi St., Harbin, 150001
CHINA

Abstract: - The testing and operation environments may be essentially different, thus the predicted reliability of the testing phase is different from that of the operational phase. On the basis of analyzing similarities and differences of the testing phase and the operational phase, using the operational reliability and the testing reliability, different forms of the comparison between the operational failure ratio and the predicted testing failure ratio are proposed, and the mathematical discussion and analysis are performed in detail. Finally, software optimal release is studied using software failure data, and the results show that conclusion can be derived by applying this method.

Key-Words: - software reliability, non-homogeneous Poisson process, testing reliability, operational reliability

1 Introduction

The software system is the nerve of a computer system. After finishing developing software, the quality requirement of the software is the key deciding whether the software can be put into practice, and one of the important characteristics of the software quality is the software reliability. Computers are increasingly used in many Safety Critical Systems [1], such as aviation and astronavigation, national defense and traffic. These systems are called SCS because their failures will cause a great loss of lives and properties. Therefore, the evaluation of the software reliability is especially important. The software reliability growth model can be used to estimate quantities associated with the software reliability, such as the number of remaining failures and the failure ratio, and so on [2]. The testing and operational environments of the software are assumed to be similar in most of the software reliability models, in which the modeling on the failure process of a software system is performed using the software failure information obtained by testing, the reliability of the software system is evaluated, and the field behaviors when the software actually runs are predicted. In fact, the software performance depends on its operational environment which includes the operation system, the hardware platform and the operational profile. It is difficult for the testing profile of software to reflect the operational profile truly, accordingly, the fault detection rate (FDR) of the operational phase and the testing phase of the software are different [3-6], and the operational reliability predicted using the failure data of the testing phase is different from the actual operational reliability.

2 The Operational Reliability And The Testing Reliability of The Software

The ideal software testing is performed according to the operational profile of users under the condition that the testing environment and the operation environment of end users are similar. When testing, a plurality of testing methods are used, and an estimated failure ratio value as reasonable as possible is obtained after enough time, thereby a credible statistic estimation of the software reliability is made. However, in order to reduce the cost and obtain a maximum failure coverage ratio in a time as short as possible, the software testing is usually performed under a pseudo operational profile, and sometimes even the simulative hardware environment is used. Accordingly, the testing environment will accelerate the failure process of the software, which results in the pessimistic estimation of failure ratio and the inconsistency between the predicted value and the actual value of the software reliability [7]. Additionally, some scholars assumed the following relation between the FDR of the testing phase and that of the operational phase when considering the testing resource consuming at the time of building the software reliability models [8].

$$r \cdot w(t) \geq r_0 \cdot \delta$$

Where r and r_0 represent the FDR per testing resource of the testing phase and the operational phase, respectively, $w(t)$ and δ represent the resource consumed until the time t.

In the testing phase of the software, the failure ratio of the testing pahse varies with the proceeding of the test because of the continuous elimation of faults and associated failures occuring no longer. After releasing the software, it is the field operational phase, and the elimination of faults is little. Thus, from the viewpoint of users, the failure ratio is almost constant during the operational phase. Assuming the failure ratio of the testing phase at the software release time T is denoted by λ_T, and the failure ratio of the actual operation phase is denoted by λ_o, the relation between them can be expressed using the following equation,

$$\lambda_o \leq \lambda_T \tag{1}$$

It can been seen from the above discussion according to the experience of most scholars that the failure ratio λ_o of the operation phase is usually smaller than the predicted failure ratio λ_T of the testing phase, and it is not accurate in the actual software testing that some scholars assumed $\lambda_o = \lambda_T$ when discussing the comparison between the testing reliability and the operational reliability [9].

The relation between the failure ratio of the testing phase and that of the operational phase can be expressed by fig.1.

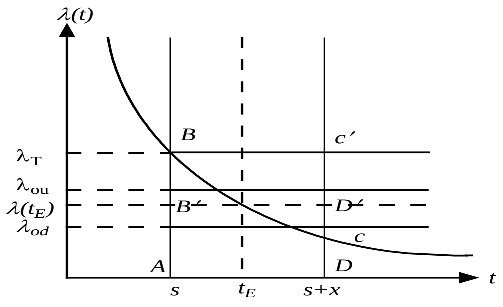

Fig.1 the testing reliability and the operational reliability when the failure ratio is decreasing

2.1 The Testing Reliability

Definition of the reliability [10]: the probability that the software does not cause system failure in a prescribed time under the prescribed conditions. According to the definition, the reliability of the system in the task time $[t, t+x]$ is:

$$R(x|t)=\Pr\{\text{there is no failures during } [t,t+x]\} \tag{2}$$

In the testing phase, if the elapse of time $[s, s+x]$ is in the testing phase, and the failure process of the software follows the NHPP process, the accumulated number of failures until time t is expressed by $N(t)$. Thus, the reliability of the testing phase $[s, s+x]$ can be expresses as:

$$\begin{aligned} R_{te}(x \mid s) &= \Pr\{N(s+x) - N(s) = 0\} \\ &= \exp\{-[m(s+x) - m(s)]\} \end{aligned} \tag{3}$$

The reliability of the testing phase $[s, s+x]$ can be expresses as in fig.1:

$$\begin{aligned} R_{te}(x \mid s) &= \exp\{-[m(s+x) - m(s)]\} \\ &= \exp[-\int_s^{s+x} \lambda(t)dt] = \exp(-s_{ABCD}) \end{aligned} \tag{4}$$

Where SABCD represents the area surrounded by the cure of failure ratio and the time $[s, s+x)$.

2.2 The Operational Reliability

If the software is released to users after testing s units, the operational reliability is calculated in the lapse of time [s, s+x] and can be expresses as:

$$R_{op}(x \mid s) = \exp(-\int_s^{s+x} \lambda_o dt) = \exp[(-\lambda_o)x] \tag{5}$$

According to equation (1), when $\lambda_o = \lambda_T = \lambda(s)$, the operation reliability can be written as:

$$R_{op}(x \mid s) = \exp[(-\lambda_T)x] = \exp[-S_{ABC'D}] \tag{6}$$

where SABC^1D represents the rectangular area surrounded by λ_T and the time $[s, s+x)$. It can been seen from Fig.1 that

$$S_{ABC'D} > S_{ABCD} \tag{7}$$

Here, assume $S_{ABC'D} = S_{ABCD}$, then

$$S_{\text{ABC'D}} - S_{ABCD} = \lambda(t_E) \cdot x - \int_s^{s+x} \lambda(t)dt = 0 \tag{8}$$

According to equation (8), $\exists t_E$, so that

$$\lambda(t_E) = \frac{1}{x} \int_s^{s+x} \lambda(t)dt \tag{9}$$

Assume the rectangular area surrounded by λ_o and the time $[s, s+x)$ is s_o, then

$$R_{OP}(x \mid s) = \exp[-S_o] \tag{10}$$

Therefore, the discussion is made according to the following three cases:

$$\begin{aligned} &1) when\ t < t_E,\ \ \lambda_o = \lambda_{ou}, S_o > S_{ABCD} \\ &2) when\ t = t_E,\ \ \lambda_o = \lambda(t_E), S_o = S_{ABCD} \\ &3) when\ t > t_E,\ \ \lambda_o = \lambda_{od}, S_o < S_{ABCD} \end{aligned} \tag{11}$$

3 Comparison Between The Two Reliabilities

From the viewpoints of users, the operational reliability is what users concern. However, most of the software reliability growth models give the testing reliability. In the testing phase, the mean function $m(t)$ is exponential, that is, the failure ratio $\lambda(t)$ is monotonically decreasing; or, the mean function $m(t)$ is S-shaped, that is, the failure ratio $\lambda(t)$ is firstly increasing and then decreasing. The following theorems are given in terms of two forms of the failure ratio.

Theorem 3.1 For any $T \geq 0$ *and* $x > 0$, when $t < t_E, i.e., \lambda_0 = \lambda_{ou}$, if $\lambda(t)$ is monotonically decreasing for any t, $R_{op}(x \mid T) < R_{te}(x \mid T)$.

Demonstration: Because $R_{OP}(x \mid s) = \exp[-S_o]$ according to equation (10), according to Eq (4), and $S_o > S_{ABCD}$ according to Eq.(11)

$$R_{OP}(x \mid s) = \exp[-S_o] < R_{te}(x \mid s) = \exp[-\int_s^{s+x} \lambda(t)dt] = \exp(-s_{ABCD})$$

Theorem 3.2: For any $T \geq 0$ and $x > 0$, when $t = t_E, i.e., \lambda_0 = \lambda(t_E)$, if $\lambda(t)$ is strictly decreasing for any t, $R_{op}(x \mid T) = R_{te}(x \mid T)$.

Theorem 3.3: For any $T \geq 0$ and $x > 0$, when $t > t_E, i.e., \lambda_0 = \lambda_{od}$, if $\lambda(t)$ is strictly decreasing for any t, $R_{op}(x \mid T) > R_{te}(x \mid T)$.

The demonstrations of theorems 3.2 and 3.3 are same as that of theorem 3.1. Most of the cases of software reliability are similar to those of theorems 3.1, 3.2, 3.3, and little software is released when the failure ratio is still increasing, so the S-shaped failure ratio is not considered here.

4 Influences of Two Reliabilities on The Software Optimal Release

After the software is developed, the quality requirement is the key deciding whether the software can be put into practice. An important characteristic is the software reliability, and the longer the software testing time is, the higher the software reliability is. Meanwhile, the software testing has a great influence on the developing cost and the distribution time of software. It is important when to stop testing the software to be released and whether the released software is reliable in the operational phase. The operational reliability is what users concern, thus, using different reliability concepts will affect the optimal release and the testing cost of the software.

The total cost of the software during the testing and operational phases is denoted by $C(T)$, then the optimal release problem is usually expressed as the following formula:

$$\min\{C(T)\} \tag{12}$$

$$\text{satisfying } R(x \mid T) \geq R_0 \tag{13}$$

The software reliability of the testing phase can be expressed as:

$$\exp\{-[m(T+x) - m(T)]\} \geq R_0 \tag{14}$$

The software reliability of the operational phase can be expressed as:

$$\exp[-\lambda_o x] \geq R_0 \tag{15}$$

Equations (12) and (14) in combination constitute the optimal release principle of the testing phase, and equations (12) and (15) in combination constitute the optimal release principle of the operational phase. Now, the following definitions are given:

T_R^1 Satisfying the minimum value of equation (14), $T_R^1 \geq 0$

T_R^2 Satisfying the minimum value of equation (15), $T_R^2 \geq 0$

T_C Satisfying the minimum value of equation (13), $T_C \geq 0$

T_{p1}^* Satisfying the optimal releasing value of testing phase

T_{p2}^* Satisfying the optimal releasing value of operational phase

Theorem 4.1 when $t < t_E$, i. e. $\lambda_o = \lambda_{ou}$, if $\lambda(t)$ is strictly decreasing for any t, there are three cases as follows:

1) 1f $R_{op}(x \mid 0) \geq R_0$, $T_{p1}^* = T_{p2}^* = T_c$

2) If $R_{te}(x \mid 0) \geq R_0 > R_{op}(x \mid 0)$, $T_{p1}^* = T_c, T_{p2}^* = \max\{T_c, T_R^2\}$

a) If $T_c \geq T_R^2$, $T_{p1}^* = T_{p2}^* = T_c$

b) If $T_c < T_R^2$, $T_{p1}^* = T_c < T_{p2}^* = T_R^2$

3) If $R_{te}(x \mid 0) < R_0$, $T_{p1}^* = \max\{T_c, T_R^1\}, T_{p2}^* = \max\{T_c, T_R^2\}$

a) If $T_c \geq T_R^2$, $T_{p1}^* = T_{p2}^* = T_c$

b) If $T_R^1 < T_c < T_R^2$, $T_{p1}^* = T_c < T_{p2}^* = T_R^2$

c) If $T_c \leq T_R^1$, $T_{p1}^* = T_R^1 < T_{p2}^* = T_R^2$

Theorem 4.2 when $t = t_E$, i. e. $\lambda_o = \lambda(t_E)$, if $\lambda(t)$ is strictly decreasing for any t, the cases are same as those of theorem 4.1.

Theorem 4.3 when $t > t_E$, i. e. $\lambda_o = \lambda_{od}$, if $\lambda(t)$ is strictly decreasing for any t, there are three cases as follows:

1) If $R_{te}(x \mid 0) \geq R_0$, $T_{p1}^* = T_{p2}^* = T_c$

2) If $R_{op}(x \mid 0) \geq R_0 > R_{te}(x \mid 0)$, $T_{p1}^* = \max\{T_c, T_R^1\}, T_{p2}^* = T_c$

a) If $T_c \geq T_R^1$, $T_{p1}^* = T_{p2}^* = T_c$

b) If $T_c < T_R^1$, $T_{p2}^* = T_c < T_{p2}^* = T_R^1$

3) If $R_{op}(x \mid 0) < R_0$, $T_{p1}^* = \max\{T_c, T_R^1\}, T_{p2}^* = \max\{T_c, T_R^2\}$

a) If $T_c \geq T_R^1$, $T_{p1}^* = T_{p2}^* = T_c$

b) If $T_R^2 < T_c < T_R^1$, $T_{p2}^* = T_c < T_{p1}^* = T_R^1$

c) If $T_c \leq T_R^2$, $T_{p2}^* = T_R^2 < T_{p1}^* = T_R^1$

Demonstration of theorem 4.3 : when $t > t_E$, i. e. $\lambda_o = \lambda_{od}$, according to theorem 3.3, $T \geq 0$, and $x > 0$, $R_{op}(x \mid T) > R_{te}(x \mid T)$

1）If $R_{te}(x \mid 0) \geq R_0$, $R_{op}(x \mid T) > R_{te}(x \mid T) \geq R_0$。Since the optimal solutions of the testing phase and the operational phase are not limited by the reliability level, $T_{p1}^* = T_{p2}^* = T_c$

2) If $R_{op}(x|0) \geq R_0 > R_{te}(x|0)$, similar to case 1), the operational phase satisfies the reliability requirements, and thus the optimal solution of the operational phase only needs to satisfy the condition of $C(T)$, $T_{p2}^* = T_c$, and the testing phase needs to satisfy $T_{p1}^* = \max\{T_c, T_R^1\}$.

3) If $R_{op}(x|0) < R_0$, since the optimal solutions of the testing phase and the operational phase are limited by the reliability level, $T_{p1}^* = \max\{T_c, T_R^1\}, T_{p2}^* = \max\left\{T_c, T_R^2\right\}$

The demonstrations of theorems 4.1 and 4.2 are similar to that of theorem 4.3.

5 Example Analysis

To illustrate the above theorems, the analysis is performed on the optimal release problem of the software using data [11]. The testing data of the software is verified by the Goel-Okumoto (G-O) model, that is,

$m(t) = a\left[1 - \exp(-bt)\right]$ (16)

The software cost model uses [12]

$C(T) = c_1 m(T) + c_2[m(\infty) - m(T)] + c_3 T$ (17)

The estimated value obtained using the maximum likelihood is $\hat{a}=1348$, $\hat{b}=0.124$.

The parameters of cost model are usually estimated through experienced data, and are adopted as follows [12].

$x = 5, c_1 = 1, c_2 = 5, c_3 = 4, R_0 = 0.8$, then

$$T_c = \frac{1}{b} \ln \frac{ab(c_2 - c_1)}{c_3} = 52.5$$

$$T_R^1 = \frac{1}{b} \ln \frac{a[1 - \exp(-bx)]}{\ln(1/R_0)} = 65.82$$

As shown in fig.2, when $t = t_E$, i.e. $\lambda_o = \lambda(t_E)$, from equation (9) we get $\lambda(t_E) = \frac{1}{x}\int_s^{s+x} \lambda(t)dt$, and then obtain when $t < t_E$, i.e. $\lambda_o = \lambda_{ou}$, it can be obtained from the item 1) of theorem 4.1 that, if $T_c \leq T_R^1$, $T_{p1}^* = T_R^1 = 65.82 < T_{p2}^* = T_R^2$, and the operational reliability level at this time is lower than that predicted when the test is stopped, thus the test must be continued until $T_{p2}^* = T_R^2$ meets the requirements of users.

The mathematical discussion of the testing reliability level and the operational reliability level and the analysis conclusions in this paper will help the software testers to select reliability criterions accurately and further decide the optimal release time of the software, thereby optimizing the resource allocation.

6 Conclusion

According to the similarities and differences between the testing phase and the operational phase, several different forms of the comparison between the failure ratio of the operational phase and that of the testing phase are analyzed in this paper, and mathematical discussions and analysis are performed in detail. Because the operational reliability is what users concern, using different reliabilities have an influence on the optimal release time and the cost of the software. Finally, a verification is performed on a certain data set using the classic G-O model by an example, and conclusion about the influence of using different reliabilities on the optimal release time is derived, that it is to continue testing to meet the reliability level requirements of users.

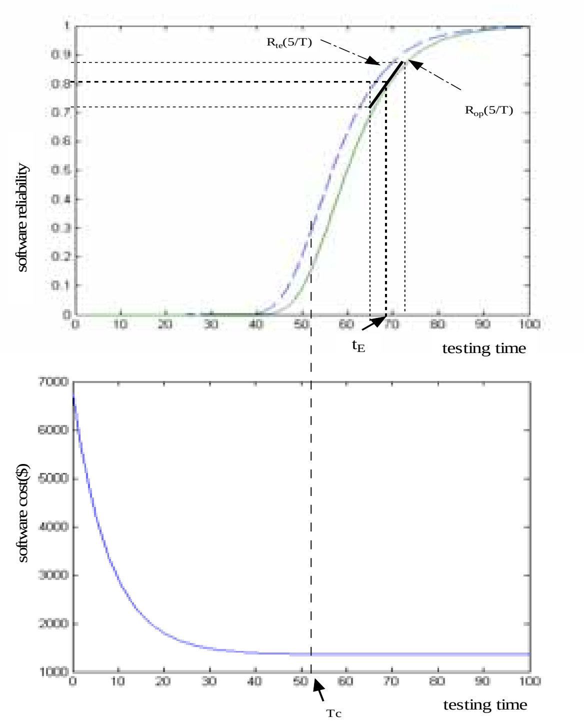

Fig.2 Illustration of optimal release of the software

Acknowledgements

This research was supported in part by NSF 60503015

References:

[1]Yang Shi-Ping, Nan Sang, Xiong Guang-Ze, Integrated safety critical systems on reliable real time network, *Proceedings of IEEE International*

Conference on Parallel and Distributed Computing, Application and Technologies, Vol.11, 2003,pp.66-70

[2] Xuemei Zhang, Xiaolin Teng, and Hoang Pham, Considering Fault Removal Efficiency in Software Reliability Assessment, *IEEE Transctions on Systems, Man, and Cybernetics*, Vol 33, No 1, 2003, pp.114-120

[3] J. D. Musa, Operational Profiles in Software Reliability Engineering, *IEEE Software Magazine*, 1993, pp. 14-32

[4] J. D. Musa, Sensitivity of Field Failure Intensity to Operational Profile Errors, *Proceedings of the 5th International Symposium on Software Reliability Engineering,* 1994, pp. 334-337

[5] Y. Chen, Modeling Software Operational Reliability via Input Domain-Based Reliability Growth Model, *Proceedings of the 28th International Symposium on Fault-Tolerant Computing* (*FTCS*), 1998, pp. 3 14-323

[6] J. D. Musa, Adjusting Measured Field Failure Intensity for Operational Profile Variation, *Proceedings of the 5th International Symposium on Software ReliabilityEngineering*, 1994, pp. 330-333

[7]Xuemei Zhang,Daniel R.Jeske, and Hoang Pham, Calibrating software reliability models when the test environment does not match the user environement. *Appl.Stochastic Models Bus.Ind.*, Vol.18, 2002, pp.87-99

[8]Hiroshi Ohtera and Shigeru Yamada, Optimal Software –Release Time Considering an Error-Detection Phenomenon during operatio, *IEEE Trans.On Reliability,* Vol. 39, No. 5, 1990, pp.596-599

[9]B.Yang, M.Xie, A study of operational and testing reliability in software reliability analysis, *Reliability Engineering and System Safety*, Vol.70, 2000, pp.323-329

[10]ANSI/IEEE Standard Glossary of Software Engineering Terminology,STD-729-1991, *ANSI/IEEE*,1991

[11]Goel,A.L.(1978), A Software Error Detection Model With Applications, *2nd Software Life Cycle Management Workshop*, sponsored by U.S. Army, Atlanta, Georgia, Aug.1978.

[12]Kazu Okumoto, Amrit L.Goel, Optimal Release Time For Software Systems, *Proc IEEE Comput Soc Int Comput Software Appl Conf 3rd*, Vol.15, 1979, pp.500-503