A Study on the Reliability Growth Trend of Operational S/W Failure Reduction

Che Gyu Shik, Kim Yong Kyung
Konyang University

abstract - The software reliability growth depends on the testing time because the failure rate varies whether it is long or not. On the other hand, it might be difficult to reduce failure rate for most of the cases are not available for debugging during operational phase, hence, there are some literatures to study that the failure rate is uniform throughout the operational time. The failure rate reduces and the reliability grows with time regardless of debugging.

As a result, the products reliability varies with the time duration of these products in point of customer view. The reason of this is that it accumulates the products experience, studies the exact operational method, and then finds and takes action against the fault circumstances.

I propose the simple model to represent this status in this paper.

key words : SRGM, mean value function, operational reliability, fault detection rate, release time, failure intensity, failure rate

1. Introduction

The S/W reliability model that describes such debugging status is defined as SRGM[1]. Previous researcher[2] performed other study based on the theory that the reliability grows through the debugging for the failure during testing and operational stages. The researcher[3], however, made a conclusion that the S/W reliability deteriorates as time goes by because the failure rate does not decrease during operational stage.

But, the failure rate decreasing status is observed uninterruptedly even without any change of S/W products. Namely, the software reliability grows for the same time passage even if there is no failure debugging. This status has been observed informally by many researchers, and papers[4,5] also indicated this.

I propose simple approaching method to do this status model without model modification. considering operational S/W reliability in this paper.

2. Reliability and failure rate of operational S/W

2.1 operational reliability

According to the papers studied up to now the release time has been determined assuming A/S(after service) may be possible even after software release.[2, 7] But, furthermore reliability growth is difficult because the developer releases it to the unspecified people in case of universal software. The reliability equation during operation is as follows.

$$R_o(x|s) = \exp(-\int_0^x \lambda_r dt) = \exp[-\lambda(s)x] \quad (2.1)$$

I consider that the failure intensity decreases as exponential function as shown in Fig. 2-1. The initial decreasing portion represents the failure decreasing

and from that point $\lambda = \lambda_r$ the line is horizontal(solid line) in Fig. 2-1 future intensity curve. The reliability during testing stage is as follows.

$$R_t(x|s) = \exp[-\int_s^{s+x} \lambda(t)dt] = \exp(-S_{ABCD}) \quad (2.2)$$

$m(t)$ = mean value function, S_{ABCD}= trapezoidal area ABCD in Fig. 2-1.
Likewise, the equation(2.2) expressing operational reliability can be as follows.

$$R_o(x|s) = \exp(-\int_0^x \lambda_r dt) = \exp[-\lambda(s)x] \quad (2.3)$$

$S_{ABC'D}$ = rectangular area ABC'D in Fig. 2-1.

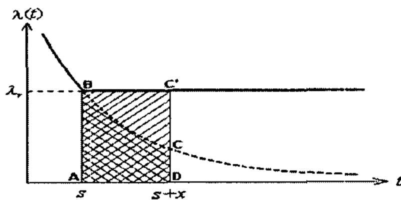

Fig. 2-1 λ(t) is monotonous decreasing in $t \geq 0$

If $\lambda(t)$ is a monotonous decreasing function $t \geqq 0$ for any time durng release time s=T≥0 and elapse time x>0, then following is concluded.

$$R_o(x|T) < R_t(x|T) \quad (2.4)$$

2.2 failure rate decreasing for the elapse time

In the meantime, the failure rate experienced in the software unit of products is very high for the initial several months and it decreases as the time goes on. This status is proven by the data collected through real experiment[8, 9]. The failur rate in this case of the products saled in t months is modelled as follows.

$$\lambda(t) = \lambda_0 \cdot \alpha^t + \lambda_f \ , \ 0 < \alpha < 1 \quad (2.5)$$

It means that there is an extra transient failure rate λ_0 if initial software products is installed. This transient failure rate decreases by decay constant a for each month. It reaches to steady-state failure rate λ_f after several months. It is shown in Fig. 2-2.

Total failure of products is not the unit failure rate times by total unit, but the sum of the released quantity times each released software failure rate. The decreasing portion from $t=0$ to $t=T$ of failure intensity curve in Fig. 2-4 is the testing phase part, and the portion from $\lambda = \lambda_\gamma$ is failure intensity of released products. It follows the curve B-E if the debugging is available after products releasing. But it is not possible so, follows the B-C curve.

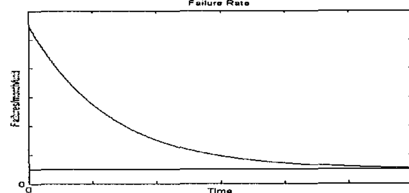

Fig. 2-2 failure rate of unit

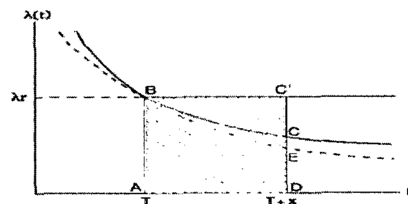

Fig. 2-3 $\lambda(t)$ decreases monotonously

Only such limited studies as the references[10-12] were performed.

2.3 parameter estimation

There are two parameter estimating methods[3]:

- maximum liklihood estimator[MLE]
- least square estimator[LSE]

The SSE is as follows.

$$SSE = \sum_{k=1}^{n} [\lambda - \lambda_k]^2 \quad (2.6)$$

I use this method as follows when the

failure rate is given as equation(2.7).

$$SSE(\alpha, \lambda_0, \lambda_f) = \sum_{k=1}^{n}[(\lambda_0\alpha^k + \lambda_f) - \lambda_k]^2 \qquad (2.7)$$

I differentiate partially this equation with a, λ_0, and λ_f . and take zero of the results.

$$\frac{\partial SSE}{\partial \alpha} = \sum_{k=1}^{n}\{\lambda_0\alpha^k + \lambda_f - \lambda_k\}k\alpha^{k-1} = 0 \qquad (2.8)$$

$$\frac{\partial SSE}{\partial \lambda_0} = \sum_{k=1}^{n}\{\lambda_0\alpha^k + \lambda_f - \lambda_k\}\alpha^k = 0 \qquad (2.9)$$

$$\frac{\partial SSE}{\partial \lambda_f} = \sum_{k=1}^{n}\{\lambda_0\alpha^k + \lambda_f - \lambda_k\} = 0 \qquad (2.10)$$

2.4 operational reliability

I propose operational reliability function applying failure in equation(2.7) as follows.

$$R_p(x|s) = \exp\left[-\int_s^{s+x}\lambda(t)dt\right] \qquad (2.16)$$

It uses failure intensity that is shown as solid line in Fig. 2-4.

I compare proposed equation with existing equations(2.3) and (2.4).

From equation(2.3) I get

$$R_t(x|s) = \exp\{-[m(s+x) - m(s)]\}$$
$$= \exp[-ae^{-bs}(1-e^{-bx})] = \exp[-m(x)e^{-bs}] \qquad (2.17)$$

From equation(2.4) I get

$$R_0(x|s) = \exp[-\lambda_r(s)x] = e^{-\lambda_r x} \qquad (2.18)$$

From equation(2.18) I get

$$R_p(x|s) = \exp\left[-\int_s^{s+x}\lambda(t)dt\right]$$
$$= \exp\left[\frac{\lambda_0\alpha^s}{\log\alpha}(1-\alpha^x) - \lambda_f x\right] \qquad (2.19)$$

3. Modeling of total failures

I want to describe how that the unit failure rate is time-related function. Therefore, if we know the products quantity soled by time t, we can expect total failure of products that we experience everyday in site.

The total failure rate of saled, delivered and operated S/W is the failure unit quantity among total saled units.

The definition of total failure rate is as follows.

$$F(t) = \frac{(faliure\ unit\ no.\ discoverd\ \ by\, t)}{(total\, operation\, units\, qty\ \ by\ \ t)} \qquad (3.1)$$

The total failure is as follows.[8]

$$F_t = F_{t-1}\alpha + \lambda_0 N_t +$$
$$\lambda_f(N_t + (1-\alpha)(N_1 + N_2 + \cdot\cdot\cdot + N_{t-1}))$$
$$= \alpha F_{t-1} + \lambda_f(N_1 + N_2 + \cdots N_t)$$
$$-\alpha\lambda_f(N_1 + N_2 \cdots + N_{t-1}) + \lambda_0 N_t \qquad (3.2)$$

4. Example

Table 4-1 given data

mo	failure qty	cumulative	mo sales	cumulative	failure rate
1	367	367	4618	4618	0.080
2	853	1220	14385	19003	0.045
3	835	2055	5608	24611	0.034
4	791	2846	6186	30797	0.026
5	956	3802	9829	40626	0.024
6	805	4607	5584	46210	0.017
7	967	5574	8240	54450	0.018
8	1218	6792	7656	62106	0.020
9	1031	7823	4914	67020	0.015
10	1144	8967	5295	72315	0.016
11	1058	10025	7418	79733	0.013

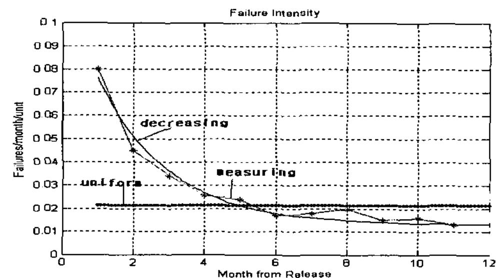

Fig. 4-1 failure rate

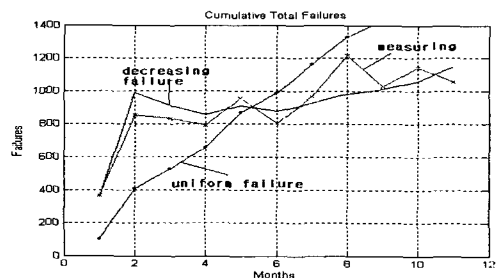

Fig. 4-2 total failures

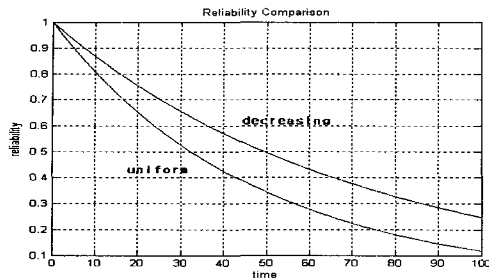

Fig. 4-3 reliability

5. Conclusion

The software failure rate decreases as the time goes on because the user becomes to know how to manipulate and to avoid failure circumstances even without any fault debugging. Almost of the existing papers did not consider this situation.

I propose simple model to express this phenomenon. And prove its fittness by testing using collected existing data.

References

[1] C. V. Ramamoorthy, F. B. Bastani, "Software reliability - Status and perspectives", IEEE Trans. on Software Eng., vol. SE-8, 1982 Aug., pp354-371

[2] Shigeru Yamada, Shunji Osaki, " Cost-Reliability Optimal Release Policies for Software Systems", IEEE Trans. on Reliability, vol.R-34, no.5, 1985,12. pp422-424

[3] B. Yang, M. Xie, " A study of operational and testing reliability in software reliability analysis", Reliability Engineering & System Safety, 70, 2000,12. pp323-329

[4] Chillarege, S. Biyani, J. Rosenthal, "Measurement of failure rate in widely distributed software", Proc. 25th Fault Tolerant Computing Symposium, FTCS-25, 1995,pp424-433

[5] S. Kan, D. Manlove, B. Gintowt, "Measuring system availability - field performance and in-process metrics", ISSRE 2003, Supplementary Proceedings, pp189-199

[6] Hiroshi Ohtera, Shigeru Yamada, "Optimal Software - Release Time Considering an Error-Detection Phenominon during Operation", IEEE Trans. on Reliability, vol.39, no.5, 1990,12. pp596-599

[7] Min Xie, "On the Determination of Optimum Software Release Time", IEEE, 1991, pp218-224

[8] P. Jalote, B. Murphy, "Reliability Growth in Software Products", Proced. of 15th Intern. Symposium on RSE, 2004, pp1-7

[9] Alan P. Wood, "Software Reliability from the Customer View", Computing Practice, 2003.8, pp37-42

[10] Musa JD,"A theory of software reliability and its application", IEEE trans. on software engineering, 1975, SE-1, pp312-327

[11] Govil KK, "A new model of software reliability prediction", Microelectronics and reliability, 1983, 23, pp1009-1010

[12] Suresh N, Babu AJG, "Software reliability estimation and optimization, a nonhomogeneous Poisson process approach", International journal of quality and reliability management", 1997, 14, pp287-300.

[13] Chin-Yu Huang, Sy-Yen Kuo, "Analysis of Incorporating Logistic Testing-Effort Function into Software Reliability Modeling", IEEE Trans. on Reliability, vol.51, 2002. Sep., pp261-270