A Cost-Reliability Model for The Optimal Release Time of a Software System

Won Hyung Lee · Yong Whi Kim

Department of Industrial Engineering, Korea University

Abstract

In this paper, faults existing in a software system is classified into three types ; simple, degenerative and regenerative faults. The reliability functions and failure rates of both a software module and system which have a mixture of such faults are obtained and the expected number of failures in the system after time T is also derived. Using the formulas obtained, a cost-reliability model and an efficient algorithm for optimal software release time are proposed via nonlinear programming formulation ; minimizing the total test cost with constraints on the failure rates of each module. Application of this model to several cases are presented and it appears to be more realistic.

1. Introduction

In this paper, faults are classified into three types : simple, degenerative, and regenerative faults, where all faults are regarded as simple faults in Black box model[4] or Markov model[7]. Simple fault is defined as that eliminating a fault does not affect on other faults in the software system; i.e.,this type of fault is independent of other faults. On the other hand, a fault is called as a degenerated fault when several faults are eliminated together if the fault is removed from the software system. Regenerative fault is distinguished from other faults since several new faults are added to the software system when a fault is eliminated from the software system. In this case, the number of rather increases after a fault is eliminated.

The main concerns of this paper are the reliability and failure rates of both software module and system which may have a mixture of these faults and it is ultimately aimed to find a more realistic cost-reliability model for optimal software release time. For this direction, we investigates reliability and failure rate for a module and then proceed to a system that consists of modules, and finally we formulate a cost-reliability model and present efficient algorithm for the solution with comparison to a existing model. First, Assumption and notations, which will be used throughout the paper, are summarized in the following section.

2. Assumptions and Notations

We assume that test starts at time τ after the completion of a software development as in Fig. 1-1. The reason to allow this time τ is to estimate the failure rate distribution by looking at the failure types and data during this time period. Of course, we can put $\tau=0$ if this is not necessary.

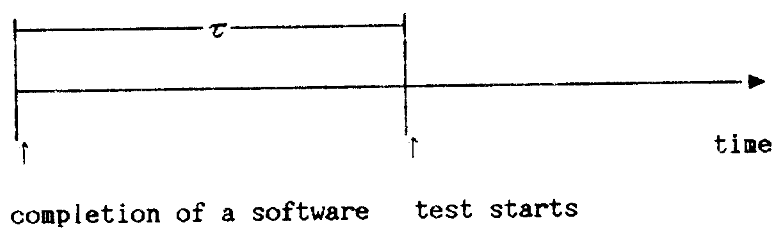

Fig 1.1 test design

Throughout this paper,the following assumptions and notations are used.

Assumptions

1) There are N faults by the completion time of software development.

2) M faults are eliminated during the time period τ.

3) Faults can be classified and corrected immediately.

4) Fault occurrence rate Φ has a gamma (α, β) distribution regardless of type, ie identically distributed.

5) Program failure rate Λ given that there are are k faults is given by

$$\Lambda = \Phi_1+\Phi_2+ \cdots +\Phi_k$$

6) Time to failure in a program ,given $\Lambda = \lambda$, has a exponential distribution with rate λ.

Notations

Λ : Program failure rate .

λ : A realization of random variable Λ.

Φ : Fault occurence rate.

ϕ : A realization of random variable Φ.

$R_m(t)$: Reliablility of a module at time t after τ.

$Rs(t)$: Reliablility of a system at time t after τ.

$E\{N^i(t)\}$, $E\{N^S(t)\}$: Expected number of failures in module i and system during time period t respectively after test start.

3. Reliability of a module

By the assumption that the occurrence rate of each fault is i.i.d. with gamma (α, β) distribution, the probability density funtion of a module failure rate Λ can be found as in Littlewood[7].

$$f(\lambda) = \frac{(\beta + \tau)\,[(\beta+\tau)\lambda]^{(N-M)-1} \exp[-(\beta+\tau)\lambda]}{[(N-M)\alpha-1]!} \quad \cdots\cdots\cdots\cdots\cdots(1)$$

, where $\alpha . \beta$: parameters of gamma distribution.

N : the number of faults in the system.

M : the number of faults eliminated during τ.

Let M_1 be the number of faults found during the debugging time τ, then M in (1) can be calculated as follows depending on the fault types eliminated;

$M = M_1$ if all the eliminated faults are simple faults, and

$M = \Sigma$ di if all the eliminated faults are degenerated faults, where di is the number of faults eliminated together when the i^{th} fault is eliminated. We also have

$M = J + d\,(M_1 - J)$ if the faults eliminated are simple and degenerated types, where J is the number of simple faults and the rest of each degenerative fault eliminates d faults simultaneously, and

$M = J + d(M_1 - J - L) - rL$ if the faults eliminated are simple, degenerative and regenerative types, where J and L are the number of simple faults and regenerative faults respectively and d(r) is the number of faults eliminated (regenerated) by eliminating each degenerative (regenerative) fault

Performing the necessary integrations on (1), we get the probability density function for the time to failure as

$$f_m(t) = \int_0^\infty f_m(t \mid \Lambda = \lambda)\, f(\lambda)\, d\lambda$$

$$= \frac{(N-M)\,\alpha\,(\beta+\tau)^{(N-M)}}{(\beta+\tau+t)^{(N-M)+1}} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2)$$

From this probability density function, the reliability and failure rate of a module are obtained. The reliability of a module is

$$R_m(t) = \int_t^\infty f_m(u)\, du$$

$$= \left(\frac{\beta+\tau}{\beta+\tau+t} \right)^{(N-M)\,\alpha} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (3)$$

and the failure rate is

$$\lambda_m(t) = \frac{f_m(t)}{R_m(t)}$$

$$= \frac{(N-M)\,\alpha}{\beta+\tau+t} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (4)$$

4. Reliability of a system

In this section, the reliability of a system is discussed in detail. Let a system consist of w modules, and the occurrence rate of faults in each module have an identical independent gamma distribution. Suppose the ith module of the system containing N_i faults such that the occurrence rate of each fault has a gamma (α_i, β) distribution, and w modules in the system being tested simultaneously by m persons. Let T_i be the time to failure for module i and y_o be the time to failure for the system, then $y_o = \min\{T_1, T_2, \cdots, T_w\}$.

Therefore, the reliability of the system can be expressed, by defining $R_i(t)$ as the reliability of the ith module, as

$$
\begin{aligned}
R_S(t) &= \Pr(y_o > t) \\
&= P_r(T_1, T_2, \ldots, T_w > t) \\
&= R_1(t)\, R_2(t) \ldots R_w(t), \text{ since each module is independent.}
\end{aligned}
$$

Substituting equation (3) of the previous section into the above $R_i(t)$'s, $R_S(t)$ of a system is obtained as

$$
R_S(t) = \left(\frac{\beta + \tau}{\beta + \tau + t} \right)^{\Sigma (N_i - M_i)\alpha_i} \qquad \ldots\ldots\ldots\ldots\ldots\ldots\ldots (5)
$$

, where $\sum_{i=1}^{w}$ denotes Σ and M_i is the number of faults eliminated in the ith module during τ.

The probability density function of the time to failure is then obtained by

$$
f_S(t) = \frac{-\,dR_S(t)}{dt}
$$

$$= \frac{\Sigma(N_i - M_i)\alpha_i (\beta + \tau)^{\Sigma(N_i - M_i)\alpha_i}}{(\beta + \tau + t)^{\Sigma(N_i - M_i)\alpha_i + 1}} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots (6)$$

Therefore, the failure rate of a system becomes

$$\lambda_s(t) = \frac{f_s(t)}{R_s(t)}$$

$$= \frac{\Sigma(N_i - M_i)\alpha_i}{(\beta + \tau + t)} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots (7)$$

Here, the expression of M_i for each type of fault is the same as in the previous section.

5. The Failure Rate And The Average Number of Failures in The System after Time T

This section presents the failure rate and the average number of failures for each module and the corresponding system at time $t = T+\tau$, ie T hours after τ based on the formulas developed in the previous section.

Let $Q = N-M$ at time τ and the number of faults eliminated between τ and $\tau+T$ be K, then the number K has a binomal distribution so that

$$P_r\ (K=k) = \binom{Q}{k} P^k (1-P)^{Q-k}.$$

Here P is the probability that a fault occurs during$(\tau, \tau+T)$ and thus given by P=F(T), where from equation (3)

$$F(T) = 1 - \left(\frac{\beta + \tau}{\beta + \tau + T}\right)^{\alpha} \quad \text{for each fault.}$$

Thus,

$$P_r\ (K=k) = \binom{Q}{k} \left[1-\left(\frac{\beta + \tau}{\beta+\tau+T}\right)^{\alpha}\right]^{k}\left[\left(\frac{\beta + \tau}{\beta+\tau+T}\right)^{\alpha}\right]^{(Q-K)} \quad \text{..........(8)}$$

The unconditional reliability function of a module at time τ+T+t can be written as a function of t, with little algebraic manipulations.

$$R_m(t) = \sum_{k=1}^{Q} R_m\ (t \mid K=k\ )\ Pr(K=k)$$

$$= \sum_{k=1}^{Q} \left(\frac{\beta+\tau+T}{\beta+\tau+T+t}\right)^{(Q-k)\alpha} \binom{Q}{k} \left[1 - \left(\frac{\beta+\tau}{\beta+\tau+T}\right)^{\alpha}\right]^{k} \left[\left(\frac{\beta+\tau}{\beta+\tau+ T}\right)^{\alpha}\right]^{(Q-k)}$$

$$= \left[1 - \left(\frac{\beta+\tau}{\beta+\tau+T}\right)^{\alpha} + \left(\frac{\beta+\tau}{\beta+\tau+T}\right)^{\alpha} \left(\frac{\beta+\tau+T}{\beta+\tau+T+t}\right)^{\alpha} \right]^{Q} \quad \text{.................(9)}$$

Therefore, the reliability of a system consisting of w module is

$$R_S(t) = \prod_{i=1}^{w} R_i(t)$$

, where $R_i(t)$ is the reliability function of the ith module.

On the other hand, the average number of faults found between τ and τ + T for the ith module, denoted by $E[N^i(t)]$, can be written as equation(10).

$$E[N^i(T)] = Q_i\ P_i$$

$$= Q_i\left[1- \left(\frac{\beta+\tau}{\beta+\tau+T}\right)^{\alpha i}\right] = (N_i-M_i)\left[1-\left(\frac{\beta+\tau}{\beta+\tau+T}\right)^{\alpha i}\right] \quad \text{..........(10)}$$

, where Q_i, P_i are the number of faults in module i at time τ and the probability that a fault occurs in module i during $[\tau, \tau+T]$ respectively.

The failure rate of ith module, $\lambda_i(t)$, can be obtained by

$$\lambda_i(t) = \frac{f_i(t)}{R_i(t)} = \frac{-R_i'(t)}{R_i(t)}$$

Thus we have, at t=T

$$\lambda_i(T) = \frac{Q_i \alpha_i (\beta+\tau)^{\alpha_i}}{(\beta+\tau+T)^{\alpha_i+1}} = \frac{(N_i-M_i)\alpha_i(\beta+\tau)^{\alpha_i}}{(\beta+\tau+T)^{\alpha_i+1}} \quad \cdots\cdots\cdots\cdots(11)$$

, where $f_i(t)$ is the pdf of time to failure for the ith module.

Consequently, the average number of failures during $(\tau, \tau+T)$ and the failure rate at time $\tau+T$ for the system, denoted by $E[N^s(T)]$ and $\lambda_s(T)$ respectively, can be written as

$$E[N^s(T)] = \sum E[N(T)] = \sum(N_i-M_i)\left[1-\left(\frac{\beta+\tau}{\beta+\tau+T}\right)^{\alpha_i}\right] \quad \cdots\cdots\cdots\cdots(12)$$

$$\lambda^s(T) = \sum\left[\frac{(N_i-M_i)\alpha_i(\beta+\tau)^{\alpha_i}}{(\beta+\tau+T)^{\alpha_i+1}}\right] \quad \cdots\cdots\cdots\cdots(13)$$

6. A Mathematical Model

In this section a mathematical model for the optimal release time is formulated as followings, utilizing the formulas developed in the previous sections.

$$\text{Minimize } C_1E[N^s(T)] + C_2\{E[N^s(T_{LC})]-E[N^s(T)]\} + C_3g(T)$$

$$\text{subject to } \lambda_s(T_s) \le R_s$$
$$\lambda_1(T_1) \le R_1$$
$$\vdots$$
$$\lambda_w(T_w) \le R_w$$

, where $C_2 > C_1 > 0$, $C_3 > 0$ and

$E[N^S(T)]$: The expected number of faults found in a system during the period of T.

$E[N^S(T_{LC})]$: The expected number of faults found during the life cycle time T_{LC}.

$g(T)$: A monotonicaliy increasing function expressing the benefit related to the testing time T.

$\lambda_s(T_s)$: The failure rate of the whole system at the testing time T_s.

$\lambda_i(T_i)$: The failure rate of the ith module at the time T_i defined earlier.

R_s : The upper limit of the failure rate for the whole system.

R_i : The upper limit of the failure rate for the ith module.

The above model is similar to the Yamada and Osaki's model [14]. However the difference is that this model incooperates more realistic situations by considering the following two aspects: the function g(t) is included in the objective function to represent the benefit related to debugging time T, and the failure rate is restricted for each module as well as whole system since each module is tested individually.

Consequently the objective function can be interpreted as follows :
$C_1E[N(T)] + C_3\, g(T)$ describes the cost during debugging and
$C_2\{E[N(T_{LC})] - E[N(T)]\}$ expresses the cost after releasing a software.

Using the formula (11), the time satisfying a constraint can be derived as

$$\lambda_i(T_i) = \frac{(N_i-M_i)\,\alpha_i(\beta+\tau)^{\alpha_i}}{(\beta+\tau+T)^{\alpha_i+1}} \leq R_i \qquad \text{for } i=1, \ldots\ldots, w.$$

Therefore ,

$$T_i \geq \left[\frac{(N_i-M_i)\alpha_i(\beta+\tau)^{\alpha_i}}{R_i} \right]^{1/(\alpha_i+1)} - (\beta+\tau) \text{ for } i=1, \ldots, w.$$

The most favorite time of the ith constraint is then

$$T_i^* = \left[\frac{(N_i-M_i)\alpha_i(\beta+\tau)^{\alpha_i}}{R_i} \right]^{1/(\alpha_i+1)} - (\beta+\tau) \text{ for } i=1, \ldots, w.$$

On the other hand, it is not easy deriving a formula to find the time satisfying the constraint for the whole system. From the constraint for the system, one can see that it is not possible transforming the above inequality into the formula similar to that for T_i due to the denominator. In order to find favorite time satifying the above constraints, a point searching method is developed as the following

PSA(Point Searching Algorithm)

Step 0. Initialization

a) specify ε, R_s, T_s

b) $K = 0$, $I I = 2$

Step 1. Computing $\lambda(T_K)$ for the initial T_s

Step 2.

a) If $| R_s - \lambda_s(T_K) | \leq \varepsilon$, go to Step 5

b) If $| R_s - \lambda_s(T_K) | > \varepsilon$, and

i) $R_s - \lambda_s(T_K) \geq 0$, go to Step 3

ii) $R_s - \lambda_s(T_K) < 0$, go to Step 4

Step 3.

a) If $R_s - \lambda_s(T_{K-1}) \geq 0$,

$$T_K = T_K - \frac{T_K}{I\ I}, \quad K=K+1$$

go to Step 1

b) If $R_S - \lambda_S(T_{K-1}) < 0$,

$I\ I = I\ I + 1$

$$T_K = T_K - \frac{T_K}{I\ I}, \quad K=K+1$$

go to Step 1

Step 4.

a) If $R_S - \lambda_S(T_{K-1}) < 0$,

$$T_K = T_K + \frac{T_K}{I\ I}, \quad K=K+1$$

go to Step 1

b) If $R_S - \lambda_S(T_{K-1}) \geq 0$,

$I\ I = I\ I + 1$

$$T_K = T_K + \frac{T_K}{I\ I}, \quad K=K+1$$

go to Step 1

Step 5. Terminate

The favorite time $T_S^* = T_K$

The favorite time satisfying all constraint is $\max\{T_S^*, T_1^*, \ldots, T_w^*\}$.

Expanding the objective function using the formula(12), we get a nonlinear function,

$$f(T) = (C_1 - C_2)E[N^S(T)] + C_2\, E[N^S(T_{LC})] + C_3\, g(T)$$

$$= (C_1 - C_2)\Sigma(N_i - M_i)\left[1 - \left(\frac{(\beta + \tau)}{(\beta+\tau+T)}\right)^{\alpha_i}\right]$$

$$+ C_2\,\Sigma\,(N_i - M_i)\left[1 - \left(\frac{\beta + \tau}{\beta+\tau+T_{LC}}\right)^{\alpha_i}\right] + C_3\, g(T)$$

f(T) is nothing but a convex function as can be proved easily by the first and second derivatives of f(T) with respect to T. Using Newton's point searching method, one can obtain the minimum point of the above convex function. The minimum point is obtained by using the following formula successively until the difference between T_{k+1} and T_k is acceptable.

$$T_{k+1} = T_k - \frac{(C_1 - C_2)\ \Sigma(N_i - M_i)\alpha_i \dfrac{(\beta+\tau)^{\alpha_i}}{(\beta+\tau+T_k)^{\alpha_i+1}} + C_3\, g'(T_k)}{(C_2 - C_1)\ \Sigma(N_i - M_i)\alpha_i(\alpha_i + 1)\dfrac{(\beta+\tau)^{\alpha_i}}{(\beta+\tau+T_k)^{\alpha_i+2}} + C_3 g''(T_k)}$$

Let T_o be the optimal point of f(T), Then the optimal release time which minimizes the total cost and satisfies all constraints is selected among $\{T_o, T_s^*, T_1^*,\ldots,T_w^*\}$. Two cases are possible; $T_o > \max\{T_s^*, T_1^*,\ldots,T_w^*\}$ or $T_o \le \max\{T_s^*, T_1^*,\ldots,T_w^*\}$. Considering both cases, the optimal release time T^* is equal to $\max\{T_o, T_s^*, T_1^*,\ldots,T_w^*\}$.; the minimal point of f(T) satisfying all constraints.

In the next section, computational results for five different benefit functions are considered to observe the dependence of optimal release time on g(T). The functions taken are : $g(T) = T^{0.5}$, $g(T) = T^{0.75}$, $g(T) = T$, $g(T) = T^{1.25}$, $g(T) = T^{1.5}$.

7. Computational Results

A set of test problem have been solved to observe the optimal release time of the model. All test problems have 11 constraints; one constraint for the whole system and 10 constraints for modules, and two cases were considered : $\tau = 0$ and $\tau \neq 0$. In case of $\tau = 0$, all faults in the system were assumed to be simple faults, and for $\tau \neq 0$, were assumed 3 cases; simple ,degenerative, simple and degenerative. The model were coded in FORTRAN-77 and run on a Macintosh SE personal computer, with appropriate data for τ, C_i's, R_s,T_s, R_i's, etc.

Table 1. Optimal Release Times with g(T) = T

$\tau = 0$	$\tau \neq 0$		
simple	simple	degenerative	simple+degenerative
113.5 (T_8*)	87.6(T_8*)	76.2(T_3*)	90.01(T_3*)
71.54(T_9*)	83.3(T_9*)	52.4(T_7*)	77.6(T_7*)
87.65(T_7*)	88.7(T_7*)	65.6(T_7*)	75.7(T_7*)
87.4 (T_7*)	85.2(T_7*)	57.5(T_8*)	74.1(T_7*)
111.02(T_7*)	85.2(T_7*)	57.3(T_8*)	63.6(T_{10}*)

Table 1 shows the results for each case with fixing g(T). Here the parenthesized T_i* shows the time which achieves max $\{T_o, T_s^*, T_1, \ldots, T_*\}$ for each problem.

Next, the test problems are runned by varying g(T) to observe the dependence of optimal release time on the form of g(T).

Table 2 shows the computational results from which we can make following observations.

Table 2. Optimal Release Time with Various g(T)

g(T)	$\tau = 0$	$\tau \neq 0$		
	simple	simple	degenerative	simple+degenerative
$T^{0.5}$	111.4(T_0)	79.8(T_0)	58.2(T_0)	67.8(T_0)
$T^{0.75}$	42.6(T_0)	44.01(T_5^*)	35.4(T_3^*)	40.1(T_3^*)
T	33.3(T_5^*)	44.01(T_5^*)	35.4(T_3^*)	40.1(T_3^*)
$T^{1.25}$	33.3(T_5^*)	44.01(T_5^*)	35.4(T_3^*)	40.1(T_3^*)
$T^{1.5}$	33.3(T_5^*)	44.01(T_5^*)	35.4(T_3^*)	40.1(T_3^*)

Comparing with Table 1, one can see that some of the optimal release times are minimal points of the objective function, i.e. T_0 is the largest time among T_0, T_S^*, $T_1^*,\ldots,T_w^*$. T_0 is apt to be the optimal release time in case of $g(T) = T^{0.5}$ or $T^{0.75}$, since the testing time for minimizing the total cost is generally larger than the times satisfying constraints.

Table 3 shows the minimum points of the objective function in case of no constraints, which can be solved uniquely by Newton's method.

Table 3. The minimum cost time for various g(T)

g(T)	$\tau = 0$	$\tau \neq 0$		
	simple	simple	degenerative	simple+degenerative
$T^{0.5}$	97.61	72.64	54.47	64.04
$T^{0.75}$	39.62	29.9	22.68	26.48
T	20.13	15.22	11.48	13.43
$T^{1.25}$	11.68	8.8	6.6	7.74
$T^{1.5}$	7.48	5.64	4.24	4.96

Observe that for every case the minimum cost time decreases exponentially as increasing the power of T in g(T) so that the minimum cost times in case of $g(T)=T^{1.5}$ are very small compared with those of $g(T)=T^{0.5}$, where the former represents the case of insufficient time and the latter case has enough time from the viewpoint of benefit obtained by releasing a software timely. Accordingly, one can recognize that the benefit function g(T) plays an important role in deciding the optimal release time of a software system.

8. Conclusions

In this paper, a cost-reliability optimal release time model for a software system has been studied. Faults existing in a software system are firstly classified into three types, and formulas for reliabilites, failure rate, and mean number of faults in the system are obtained regarding these three types of faults. This model considers the constraints on both modules and whole system as well as the mixture of fault types, and incooperates a benefit function for the release time. Thus it can be considered as an extension to existing models which considers only simple faults and constraints on whole system without benefit function. Consequently, it is expected that the model developed here would give more realistic solution comparing with the existing models.

References

[1] Catuneanu, V.M. and A.N. Mihalache, "Improving the Accuracy of the Littlewood - Verall Model", IEEE Trans., Vol. R-34, No. 5, 1985.

[2] Currit, P.A., M. Dyer and H.D. Mills, "Certifying the Reliability of Software", IEEE Trans., Vol SE-12, No. 1, 1986.

[3] Geol,A.L, "Forward software Reliability", IEEE Trans., Vol. SE-12, No. 1, 1986.

[4] Geol,A.L, and K.Okumoto, "Time-Dependent Error-Dection Rate Model for Software Reliability and Other Performance Measuers", IEEE Trans., Vol. R-28, PP. 206-211, 1979.

[5] Joe, H and N.Reid, "On the Software Reliability Models of Jelinoki-Moranda and Littlewood", IEEE Trans., Vol. R-34, No. 3, 1985

[6] Littlewood, B., "How to Measure Software Reliability and How not to Measure Software Reliability", IEEE Trans., Vol. R-28, PP. 103-110, 1979.

[7] ———————— , "Software Rrliability Model for Modular Program Structure", IEEE Trans., Vol. R-28, PP. 241-247, 1979.

[8] ———————— , "A Bayesian Differential Debugging Model for Software Reliability", Proc.COMPAS, PP. 511-519, 1980.

[9] ———————— , "Stochastic Reliability-Growth : A Model for Fault-Removal in Computer-Programs and Hardware-Designs", IEEE Trans., Vol. R-30, No. 4, 1981.

[10] Musa, J., "Validity of Execution-Time Theory of Software",IEEE Trans., Vol. R-28, PP. 181-191, 1979.

[11] Pressman, R.S, <u>Software Engineering:A Practitioner's Approach,</u> McGrow-Hill, 1982.

[12] Schick, G.T. and R.W. Wolverton, "An Analysis of Compeying Software Reliability Models", IEEE Trans., Vol. SE-4, No. 2, 1987.

[13] Schnidewind, N.F. and H.M. Hoffman, "An Experiment in Software Error Data Collection and Analysis", IEEE Trans., Vol. SE-5, No. 3, 1979.

[14] Yamada, S. and S. Osaki, " Cost-Reliability Optimal Release Polices for Software System", IEEE Trans., Vol. R-34, No. 5, 1985.

[15] Zelkowitz, M.V., "Perspectives on Software Engineering", Computer Surveys, Vol. 10, PP. 197-210, 1978.