ISSN No:-2456-2165

Software Reliability Growth Model with New Dynamic Learning Effects and Release Time Determination

[1]Shaik.Mohammad Rafi, [2]Dr. B. Srinivasa Rao, [3]Dr.Shaheda Akthar
[1]Research Scholar in the Department of Computer Science and Engineering, Acharya Nagarjuna University, Guntur
[2]Professor in the Department of Computer Science and Engineering
[3]Registrar, Dr.Abdul Haq Urdu University, Kurnool.

Abstract:- Main aim of software industry to produce software product which is of high quality and failure free. Software reliability growth models are helping the industries to produce required quality products by providing a mathematical models based on simulated testing oriented environment. Testing is the one of the costliest phase where several resources were consumed. In the history of software reliability several authors are proposed several research papers on this topic. Software reliability growth models are developed on the basis of data which was obtained during testing phase in the software engineering. In this paper we proposed a new software reliability growth model which is a enhanced model proposed by chiu. Model parameters are estimated through standard procedures. Evaluation criteria are used to analyze the performance of the proposed model and results obtained are compared with existing models. Proposed models seem to be best fit compared to all other models which are proposed earlier. software release time determination is also be done.

Keywords:- *Software Reliability, Software testing, testers learning, Non-homogeneous Poisson Process (NHPP), Software Cost.*

I. INTRODUCTION

Testing is the one of the precious and costliest phase of all the phases in the software industries. Every industry is struggling to produce quality product in the given time and environmental conditions. Testing consumes lots of resources which are like time, cost and many others. Fifty percent of cost of the software is consumed during testing which makes the industries to pay more attention towards it. If the testing is done in a qualified manner then quality of the product will improve if not it leads to bad impact on the product. Software reliability is defined as probability of failure free software for a specified period of time in a specified environment. Software development process is itself time consuming and time bound process. Every software industry wants to produce quality software , but it is very complex to measure the required quality. Quality of the software can be improved by reducing the number of errors. software reliability growth models are build on the mathematical and statistics concepts. After every successful software testing data is generated usually called as software failure data which is useful for estimating software reliability. The failure data are categorized as of two types one failure count data and another time between the failure data. These software failure data were very useful for estimating the reliability and quality of the product. In the history software engineering many papers were published to estimate the software reliability. Some authors assume failure detection to be constant. But failure detection is varies with time. Some researchers feels that software failure detection and correction to be an exponential process[10]. S shaped models are derived based on assumption testing will proceed slow at early stages and then later when they gain experience it proceeds normally[11]. Software reliability growth models are mimic mathematical and statistical models designed to reflect the real time testing environment. Some People feel that total number of initial errors is constant and no new errors are added during the software testing. Some believe that total number of errors varies with time based on chances of getting new errors during testing. .Testing effort based software reliability models developed by many authors. Some authors integrated testing efforts into their software growth models. Some have designed models based on constant failures and some assumes failures during testing will varies with time. Recently chiu and huang[5] proposed new software reliability growth model based on learning effects during testing. Learning factors can effects testing during software testing phase, will pay an important role as testers familiar with testing environment then it effects on failure identification. Learning and experience of testers can effects on defect identification and improves the testing environment. These experience and learning may also vary with time and effect dynamic effect on software testing environment. Javaid Iqbal and N. Ahmad and S.M.K. Quadri proposed an enhanced model by introducing a negligent factor in the software reliability growth models[1], they felt that testes are little negligent during testing. Recently Javid Iqbal[4] has integrated learning based functions into software reliability growth model ,by assuming both exponential and linear learning factors. In this paper we proposes a new enhancement by introducing a new dynamic learning phenomenon based on assumption that as the testing proceeds new failures are introduced on which each time the experience and learning phenomenon of testers also dynamically varies with time so we introduced new dynamic learning function which dynamically varies with time. Our proposed model has been fitted with real time datasets and parameters are estimated. Performance and validation is

done through comparing with other models . Proposed models fits better compared with other models.

II. SOFTWARE RELIABILITY GROWTH MODELS WITH LEARNING FACTORS

A. *Chiu , Huang and Lee learning model [5]*

In this model authors proposed a imperfect debugging environment based software reliability model based on casual loop diagram. They incorporated learning and experience of software testers in their models. They feel that learning and experience of software during software testing can effect on software testing during defect identification in constant environment. they feel that learning and experience factors are constant.

$$f(t) = (\alpha + \eta * F(t)) * (1 - F(t)) \quad (1)$$

above equation solved by assuming F(0)=0 then

$$F(t) = \frac{e^{(\alpha+\eta)*t} - 1}{\frac{\eta}{\alpha} + e^{(\alpha+\eta)*t}} \quad (2)$$

B. *Kuei-Chen Chiu [2] and chiu , Kuei Chen 2013[7]*

In this paper author proposed new model based on time varying learning phenomenon by introducing new learning factors where they introduced two new time varying learning factors into their model. $\eta(t) = (1 + \xi * t)$ and $\eta(t) = e^{\xi * t}$ where ξ represents coefficient of accelerating factor.

$$F(t) = \frac{e^{1+\frac{\eta+\xi*t}{\alpha'}} - 1}{\frac{\eta}{\alpha} + e^{1+\frac{\eta+\xi*t}{\alpha'}}} \quad (3)$$

$$F(t) = \frac{e^{1+\frac{\eta+e^{\xi*t}}{\alpha'}} - 1}{\frac{\eta}{\alpha} + e^{1+\frac{\eta+e^{\xi*t}}{\alpha'}}} \quad (4)$$

here learning and experience factors varies with time.

C. *Javid Iqbal , N.Ahmad and S.M.K Quadri [1][3]*

In this paper authors assumes that software testers ate little negligent during testing process where it has adverse effect on software testing. They incorporated an negligent factor into their model.

$$\alpha'(t) = \eta_1 * \alpha - \tau \quad (5)$$

Now the

$$F(t) = \frac{e^{1+\frac{\eta_2}{\alpha'(t)}} - 1}{\frac{\eta}{\alpha} + e^{1+\frac{\eta_2}{\alpha'(t)}}} \quad (6)$$

D. *Proposed Model*

In this paper we proposed a new dynamic learning factor which can capable to give different effects on learning experience of testers. we assume that software learning phenomenon depending on the environmental conditions of testing phase. Testing phase environment is dynamic in nature so we have introduced learning phenomenon as one dynamic function instead constant factor .

$$\frac{dF(t)}{dt} = ([\alpha + \eta(t) * F(t)]) * ([1 - F(t)]) \quad (7)$$

as we have integrated $\eta(t)$ as dynamic time oriented function which can varies with time. depending on testing environment the learning function also varies with time.

$$z(t) = \int_0^t \frac{dF(t)}{1-F(t)} = \int_0^t (\alpha + \eta(t) * F(t))dt \quad (8)$$

$$F(t) = (1 - e^{-(\alpha*t+\phi(t))}) \quad (9)$$

$$\theta * F(t) = m(t) = \theta * (1 - e^{-(\alpha*t+\phi(t))}) \quad (10)$$

$$\lambda(t) = (\alpha + \phi'(t)) * e^{-(\alpha*t+\phi(t))} \quad (11)$$

$$\phi(t) = \int_0^t \eta(t) * F(t)dt \quad (12)$$

from above equation $z(t)$ represents Hazard Rate function which completely depending on testing and performance of the testers. it is observed that hazard rate function z(t) depends on two other functions learning function $\eta(t)$ and *a* distribution function *F(t).* By assigning suitable distribution functions into these functions can give dynamic effects into deriving models. For that we assumed two different functions represented in eq.(13)

In learning function $\eta(t)$ r represents learning accelerating factor, η learning factor , F(t) an adjustment distribution function which adjusts the given testing environment, in this paper we assumed adjustment function as exponential in nature. β represents distribution parameter.

$$\eta(t) = \eta * e^{r*t} \ and \ F(t) = (1 - e^{-\beta*t}) \quad (13)$$

substituting (13) into (12) and assuming F(0)=0 we derived the following equation

$$m_1(t) = \theta * \left\{ 1 - e^{-\alpha*t-\eta*\left[\frac{(e^{r*t}-1)*(1-e^{-\beta*t})-\beta*\left(\frac{e^{(-\beta*t+r*t)}}{r-\beta}+\frac{1}{\beta*e^{\beta*t}}\frac{1}{r-\beta}\frac{1}{\beta}\right)}{r}\right]} \right\} \quad (14)$$

$$\eta(t) = \eta * (1 + r * t) \ and \ F(t) = (1 - e^{-\beta*t}) \quad (15)$$

substituting (13) into (10) and assuming F(0)=0 we derived the following equation

$$m_2(t) = \theta * \left\{ 1 - e^{-\alpha*t-\eta*(t+\frac{1}{2}*r*t^2)*(1-e^{-\beta*t})+\left[\beta*\left(-\frac{1}{2}*\frac{e^{-(\beta*t)}*(\beta^2t^2+2\beta t+2\beta+2)}{\beta^3}\right)+\frac{\beta+1}{\beta^3}\right]} \right\} \quad (16)$$

difference between Kuei-Chen Chiu (2012) and chiu , Kuei Chen 2013 and our model is we are incorporated dynamic factor in to software reliability growth model where as they assumed and substituted terms related to learning factors into their respective models.

III. PARAMETER ESTIMATION

In this paper we used standard procedure as least squate estimation to validate our proposed. As the equation

ISSN No:-2456-2165

is little complex in nature we used numerical approximations.

$$SSE = \left\{(m_i - \theta * (1 - e^{-(\alpha * t_i + \phi(t_i))}))\right\}^2 \quad (17)$$

$$\frac{d(SSE)}{d\theta} = -2 * \sum_{i=1}^{n} \left\{(m_i - \theta * (1 - e^{-(\alpha * t_i + \phi(t_i))}) * \left\{(1 - e^{-(\alpha * t_i + \phi(t_i))})\right\}\right\} = 0 \quad (18)$$

$$\frac{d(SSE)}{d\alpha} = -2 * \sum_{i=1}^{n} \left\{(m_i - \theta * (1 - e^{-(\alpha * t_i + \phi(t_i))}) * \left\{((\theta * t_i) e^{-(\alpha * t_i + \phi(t_i))})\right\}\right\} = 0 \quad (19)$$

$$\frac{d(SSE)}{d\eta} = -2 * \sum_{i=1}^{n} \left\{(m_i - \theta * (1 - e^{-(\alpha * t_i + \phi(t_i))}) * \left\{\theta * (e^{-(\alpha * t_i + \phi(t_i))} * \frac{d(-(\phi(t_i)))}{d\eta})\right\}\right\} = 0 \quad (20)$$

$$\frac{d(SSE)}{dr} = -2 * \sum_{i=1}^{n} \left\{(m_i - \theta * (1 - e^{-(\alpha * t_i + \phi(t_i))}) * \left\{\theta * (e^{-(\alpha * t_i + \phi(t_i))} * \frac{d(-(\phi(t_i)))}{dr})\right\}\right\} = 0 \quad (21)$$

$$\frac{d(SSE)}{d\beta} = -2 * \sum_{i=1}^{n} \left\{(m_i - \theta * (1 - e^{-(\alpha * t_i + \phi(t_i))}) * \left\{\theta * (e^{-(\alpha * t_i + \phi(t_i))} * \frac{d(-(\phi(t_i)))}{d\beta})\right\}\right\} = 0 \quad (22)$$

IV. EVALUATION CRITERIA

A. *Coefficient of multiple determinations (R^2)*

Which measures the percentage of tells total variation about mean accounted for the fitted model and us how well a curve fits the data. It is frequently employed to compare model and access which model provides the best fit to the data. The best model is that which proves higher R^2. that is closer to 1.

$$R^2 = 1 - \frac{(Residual\ Sum\ of\ Squares)}{(Corrected\ Sum\ of\ Squares)} \quad (23)$$

$$R^2 = 1 - \sum_{i=1}^{n} \frac{(m_i - m(t_i))^2}{\sum_{i=1}^{n} (m_i - \sum_{j=1}^{n} \frac{m_j}{n})^2} \quad (24)$$

V. MODEL PERFORMANCE ANALYSIS

A. *DATA SETS*

In this paper we used standard datasets used by various authors in their research paper. we have taken the reference of datasets 1 and 2 from research paper proposed by Chiu (2008) .[5]

S.No	Reference	Datasets
1	Zhang and Pham (1998)	Failure data of misra system
2	Pham (2003)	Failure data real time control system

Table 1

model comparisons are done through R^2 .

B. *Results*

Following *Table 2* indicates parameters of our proposed models 1 and 2. model parameters are estimated through least square estimation with numerical approximations. *Table 3* indicates all fitted results of comparisons of different models based on R^2 values. table 4 shows the results of various models fitted on Zhang and Pham 1998 model data set. as from the given table 4 it seems proposed models better predicts the software failures. hence a good fit model.

datasets	proposed model $\boldsymbol{m_1(t) from\ eq.14}$	proposed model $\boldsymbol{m_2(t) from\ eq.16}$
Zhang and Pham (1998)	$\alpha = 0.146, \beta = 0.00306, \eta = 0.02118, r = 0.4, \theta = 131.6$	$\alpha = 0.2096, \beta = 0.03814, \eta = 0.2852, r = 1, \theta = 133.7$
Pham (2003)	$\alpha = 0.05266, \beta = 0.06165, \eta = 0.0000296, r = 0.1238, \theta = 121.6$	$\alpha = 0.05958, \beta = 0.003089, \eta = 0.3206, r = 1, \theta = 132.3$

Table 2

Models	Sources of datasets	
	Zhang and Pham (1998)	Pham(2003)
Pham and Zhang(2003)	0.966	0.975
Huang (2005)	0.973	0.982
Chiu (2008)	0.966	0.975
Chiu and Kuei -Chen linear model(2013)	0.975	0.987
Chiu and Kuei -Chen exponential model(2013)	0.986	0.989
Javaid Iqbal, N. Ahmad and S.M.K Quadri (2013)	0.966	0.978
Proposed Model $\boldsymbol{m_1(t)}$	0.976	0.988
Proposed Model $\boldsymbol{m_2(t)}$	0.997	0.996

Table 3

Total defects predicted by the following models based on Zhang and Pham (1998)							
Testing time (per hour)	Defects found	Pham and Zhang(2003)	Huang (2005)	Chiu (2008)	Chiu and Huang and Lee 2013	Chiu and Huang and Lee2013	Proposed model $m_2(t)$
1	27	17.515178	18.753639	17.527226	17.527305	17.527226	24.03182
2	43	32.789511	34.611824	32.795171	32.795691	32.795171	41.99699
3	54	46.105543	48.073478	46.095057	46.096522	46.095057	55.42135
4	64	57.711199	59.544218	57.680570	57.683470	57.680571	65.49290
5	75	67.823776	69.354962	67.772696	67.777430	67.772700	73.11840
6	82	76.633546	77.776637	76.563933	76.570776	76.563948	78.99073
7	84	84.306973	85.031825	84.221968	84.231067	84.222016	83.64890
8	89	90.989586	91.304011	90.892873	90.904255	90.893015	87.52462
9	92	96.808530	96.744969	96.703889	96.717483	96.704300	90.97340
10	93	101.874820	101.480660	101.765860	101.781512	101.767016	94.29050
11	97	106.285370	105.615980	106.175330	106.192839	106.178534	97.71190
12	104	110.124690	109.238530	110.016420	110.035532	110.025130	101.40280
13	106	113.466510	112.421750	113.362390	113.382835	113.385768	105.43820
14	111	116.375100	115.227370	116.277050	116.298563	116.339097	109.78500
15	116	118.906460	117.707460	118.816010	118.838314	118.978846	114.30026
16	122	121.109420	119.906050	121.027700	121.050527	121.449567	118.74723
17	122	123.026490	121.860500	122.954300	122.977408	124.023841	122.84940
18	127	124.694700	123.602620	124.632560	124.655724	127.209285	126.35618
19	128	126.146320	125.159520	126.094480	126.117512	131.484050	129.10792
20	129	127.409420	126.554440	127.367970	127.390682	135.255654	131.07247
21	131	128.508460	127.807290	128.477290	128.499548	135.971130	132.33747
22	132	129.464730	128.935240	129.443630	129.465296	135.974000	133.06583
23	134	130.296760	129.953060	130.285400	130.306378	135.974000	133.43760
24	135	131.020680	130.873580	131.018670	131.038874	135.974000	133.60442
25	136	131.650520	131.707870	131.657420	131.676788	135.974000	133.66964

Table 4

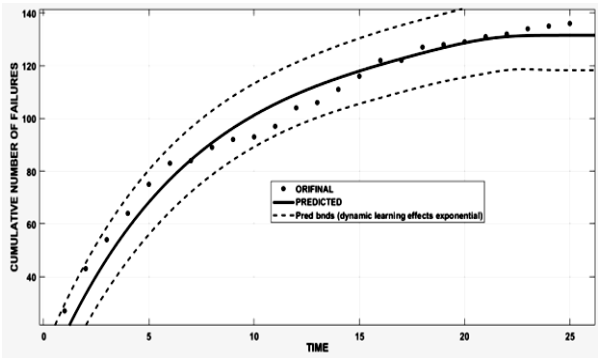

Fig 1

Figure 1 and Figure 2 indicates the estimated model data with original dataset 1and Figure 3 indicates the proposed model fitted with the original dataset 2

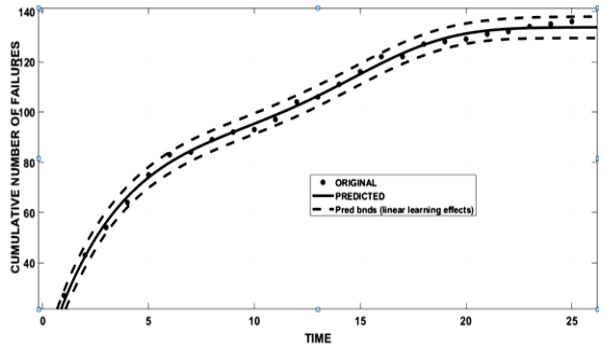

Fig 2

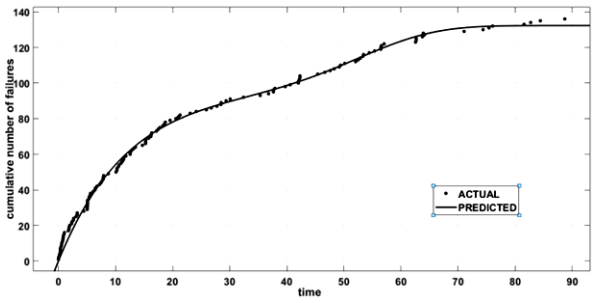

Fig 3

VI. OPTIMAL SOFTWARE RELEASE POLICY

Software release time determination is an important concern to much software development process. Software release time determination is concerned with time at which software has to be delivered to the customer such that released software product should have quality and error free. in order to determine the exact release time we must know its reliability and concerned cost of testing of the product. Once they have determined reliability and cost we can predict the release time based on cost and reliability which are predicted.

A. Software Release-Time Based on Reliability Criteria

Software reliability can be estimated based on the change in a mean value function over a period of time. for that following equations represents the concerned reliability expression

ISSN No:-2456-2165

$$R(t) = e^{[m(t+\Delta t)-m(t)]} \quad (25)$$

Lets consider the required R_0 reliability to release the software product. The expression 18 changed as

$$R_0 = e^{[m(t+\Delta t)-m(t)]} \; and \; [m(t+\Delta t) - m(t)] = \ln(R_0) \quad (26)$$

$$\theta * [\left(1 - e^{-(\alpha*(t+\Delta t)+\phi(t+\Delta t))}\right) - (1 - e^{-(\alpha*(t)+\phi(t))})] = ln(R_0) \quad (27)$$

$$[e^{-(\alpha*(t)+\phi(t))} - e^{-(\alpha*(t+\Delta t)+\phi(t+\Delta t))}] = \frac{ln(R_0)}{a} \quad (28)$$

Solving the above equation we will optimal time T_{R_0} at which the reliability could reach R_0. *Figure 4and table 5* indicates the reliability of dataset 1 through second model with mean value function $m_2(t)$. The concern reliability $R_0 = 0.95$ at.

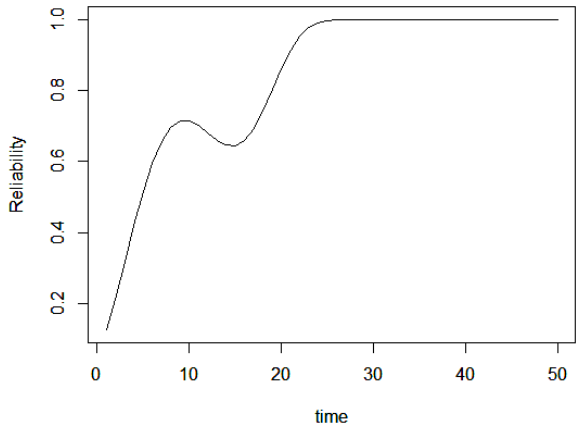

Fig 4

Time	Reliability	Time	Cost
18	0.7424	17	558.64
19	0.8018	18	540.06
20	0.8623	19	526.23
21	0.9147	20	518.52
22*	0.9534	21*	515.11
23	0.9778	22	515.28
24	0.9908	23	517.82

Table 5

B. Optimal release time based on cost criterion

Software development cost can be estimated from following expression where C1 and C2 and C3 are cost associated with correcting the errors during testing , error correction during operational use of software and miscellaneous cost during entire software development process.

$$COST(t) = C1 * m(t) + C2 * [m(t_{LC}) - m(t)] + C3 * t \quad (29)$$

now from $\frac{dCOST(t)}{dt} = 0$ then $\lambda(t) = \frac{C3}{C2-C1}$ find the T_C at which cost the software to be minimized. Let us consider the various cost related with C1=3 , C2=10, and C3=5 applied through second model $m_2(t)$ on dataset 1. *Figure 5 and table 5* show the relation between cost and time .

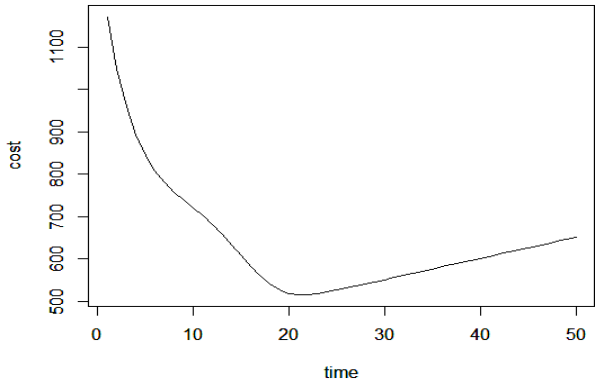

Fig 5

C. Optimal release time based on Cost and Reliability Criterion [9]

Based on above equations 20and 21 software release time can be determined based on $max\{T_{R_0}, T_C\} = \{21,21.6\}$. So the release time can be 21.6.

VII. CONCLUSIONS

This paper mainly we integrated the dynamic learning function into software reliability growth models. As testing is the one important phase where 50% of resources are being consumed. Testing phase itself is a dynamic environment where finding actual error are some difficult process same time testers need lots of experience and learning capacity to adopt the current environment fluctuations. This paper we adopted a dynamic learning function integrated into software reliability growth model. Results have shown that our proposed model fits good compared with other models. In future we want to develop some more rigorous models which can capable to adopt the fluctuating environment.

REFERENCES

[1]. Javaid Iqbal and N. Ahmad and S.M.K. Quadri " A Software Reliability Growth Model with Two Types of Learning" Proceedings of the 2012 IEEE IEEM.

[2]. Kuei-Chen Chiu " A Study of Software Reliability Growth Model for Time-dependent Learning Effects" Proceedings of the 2012 IEEE IEEM.

[3]. Javid Iqbal "Analysis of Some Software Reliability Growth Models with Learning Effects" I.J.Mathematical Sciences and Computing 2016.

[4]. Javid Iqbal " Software reliability growth models: A comparison of linear and exponential fault content functions for study of imperfect debugging situations" Cogent Engineering (2017).

[5]. Kuei-ChenChiuYeu-ShiangHuang and Tzai-ZangLee"A study of software reliability growth from the perspective of learning effects" 2008.

[6]. Huang, C.Y., Kuo, S.Y. and Lyu, M.R. (2000), "Effort-index based software reliability growth models and performance assessment", in Proceedings of the 24th IEEE Annual International Computer Software and

Applications Conference (COMPSAC'2000), pp. 454-9.
[7]. Chiu ,Kuei Chen " A discussion of software reliability growth models with time varying learning effects" American Journal of Software engineering and Applications 2013.
[8]. Pham, H. Software Reliability, Springer-Verlag,New York, NY.2000.
[9]. Yamada, S. and Osaki, S. (1985b), "Cost-reliability optimal release policies for software systems", IEEE Transactions on Reliability, Vol. R-34 No. 5, pp. 422-4.
[10]. Amrit L Goel, Kazu Okumoto, "Time dependent error detection rate model for software reliability and other performance measures", IEEE transactions on reliability, vol R-28, pp. 206-211, 1979.
[11]. Shigeru Yamada ; Mitsuru Ohba ; Shunji Osaki s-Shaped Software Reliability Growth Models and Their Applications" IEEE Transactions on Reliability (Volume: R-33, Issue: 4, Oct. 1984).