THE INTEGRATED SOFTWARE RELIABILITY GROWTH MODELS FOR VARIABLE POTENTIAL ERRORS USING TIME-VARYING LEARNING EFFECTS WITH CHANGE-POINTS

Kuei-Chen Chiu*

Abstract

Recently, various software reliability growth models (SRGMs) have been proposed to assess software reliability. Some important issues in these models include the S-curve or exponential testing behaviour, the change-points in process performance, estimation of the parameters, variations in potential errors, and learning effects in the testing process. This paper provides an integrated SRGM with time-varying learning effects to deal with multiple situations of software testing/debugging, based on the non-homogeneous Poisson process (NHPP) to satisfy both S-shaped and exponential-shaped types simultaneously. An exponential learning function is adopted to describe the learning effects varied with time, and a sine function is also adopted to point out the change-points for the testing environment and the various potential errors. The results show better fit than those of other models with actual data sets. This study also verifies the effectiveness of the proposed model with R^2, mean square error, and RRMS and LSE. criteria. The proposed model not only provides good numerical prediction performance for several different kinds of data but also explains the testing/debugging behaviour of the testing staff, the learning effects of the testing project itself, and the changes in the testing environment necessary to improve software system testing and management.

Key Words

Software reliability, non-homogeneous poisson process (NHPP), learning effects, time-varying learning effects, change-points, time-varying potential errors

1. Introduction

Software reliability growth is an important issue in the software industry as it can provide critical information for developers and testing staff during the testing/debugging phase. Over the last few decades, many software reliability growth models (SRGMs) have been developed that address some important major issues, such as S-shaped [32], [35] or exponential-shaped testing/debugging behaviour, constant or variable potential errors, and stable or unstable testing/debugging environments. Various methods have been used in these models, such as the stochastic differential equation, confidence interval estimation, soft computing, and Bayesian approaches. This study provides integrated models by which to assess the mean value functions of an SRGM that can simultaneously describe S-shaped and exponential-shaped testing/debugging behaviour, variable potential errors, both stable and unstable testing/debugging environments, and constant and variable learning effects. This research adopts Chiu's [1] models and employs a sine function [2] to discuss a testing/debugging environment with time-varying learning effects. This paper also examines the effectiveness of the proposed models with R^2, mean square error (MSE), and $RRMS$ criteria, and discusses when and what kinds learning effects occur and how these influence the software reliability. The results show good fit with the actual data, and these integrated models thus successfully explain many situations that can occur in the testing/debugging process.

2. Literature Review

SRGMs are mathematical functions that characterize the software testing/debugging process and explain how errors are removed. Various stochastic models have been proposed to assess software reliability some of which are based on the non-homogeneous Poisson process (NHPP), and they have been fairly effective in describing the error-detection process with a time-dependent error-detection rate.

Most SRGM functions assume that each time an error occurs, the fault that caused it can be immediately removed and that no new errors will be introduced, which is known

* Research Center for Health Data, National Cheng Kung University, Tainan, Taiwan, R.O.C.; e-mail: ckj0214@ms24.hinet.net

Recommended by Dr. L. Etzkorn
(DOI: 10.2316/Journal.213.2014.3.213-1060)

as perfect debugging, although imperfect debugging has also been considered to loosen this restrictive assumption [3]. Gokhale and Trivedi (1999) stated that the assumption of statistical independence for the number of events occurring in disjointed time intervals is constantly violated when SRGMs based on NHPP are used, and thus proposed an enhanced NHPP model to allow time-dependent failures to occur in the debugging process.

Learning effects usually occur when staff are involved in a production or service activity, and it is important to investigate how these learning effects will affect task time and cost. Several studies have noted that learning effects exist in the software testing/debugging process [4], and the use of an S-shaped mean value function in an SRGM implies their existence [5]–[7] (Yamada *et al.*, 1992). Learning effects have been shown to cause the failure function to have a decreasing curve in the initial testing period [8]. Some studies of scheduling have suggested that learning effects for planning different operating time for the same items [9]. Kapur *et al.* [10] adopted testing efforts and staff improvements to build an SRGM. Chiu *et al.* [11] constructed an SRGM with learning effects based on NHPP and estimated both linear and exponential learning effects, including these in a later SRGM [1]. The change-points in the testing/debugging process have been discussed in some studies [29], [36] and the change-points in the testing/debugging process with time-varying learning effects for actual data sets have also been discussed [12].

In some cases, due to different testing/debugging strategies or resource allocations, software reliability may non-monotonically increase or decrease. Zhao [13] identified the change-points problem, and stated that its effects should be considered along with software reliability. Shyur [14] developed a generalized SRGM by incorporating imperfect debugging with change-points. Huang [7] incorporated both a generalized logistic testing-effort function and a change-points parameter into an SRGM and discussed SRGM with multiple change-points by using queuing models [15], [16]. Inoue *et al.* [17] provided the two-dimensional change-points SRGM. Kapur *et al.* [18] examined the severity of errors produced by an SRGM with change-points.

Recently, there has been a great deal of research examining imperfect debugging in which the testing/debugging process involves two parts: predicting failure time and removing errors [31], [33], [34] with programmers using debugger break points for the former (Horwitz *et al.*, 2010) as well as call stack-sensitive slicing to increase debugging effectiveness. Bai *et al.* [19] stated that remaining software defect estimation (RSDE) can describe the dynamic behaviour that occurs in the testing process and presented both simple and change-point models to trace the testing process. Imperfect debugging has also been widely discussed in the literature with the aim of overcoming this restrictive assumption [3], [14], [20]–[24] (Huang *et al.*, 2007). Dick *et al.* [25] compared two models for stochastic and non-linear processes that used a chaotic-time-series analysis to make predictions. They called the resulting approach radial basis function networks (RBFN), and showed that it had better results than other models.

3. The Integrated SRGM Model with Time-Varying Learning Effects

3.1 Model Definition

The proposed models consider multiple situations in the software testing/debugging process, which include fluctuating testing environments, the variable potential errors, the time-varying learning effects, and the adoption of time-varying learning functions [1] and the sine function [2] to construct an integrated SRGM model based on the NHPP. The explanations of the main expressions in the proposed model are described as follows.

3.1.1 Notations

The following notations will be used throughout this study:

a:	the initial number of all potential errors in the software system
α:	the autonomous errors-detection factor
η:	the learning factor
$\eta(t)$:	the learning effect function with exponential learning effects
ξ:	the accelerative factor with time of learning effect
$r(t)$:	the rate of extra variation to potential errors
ν:	the fluctuating factor
δ:	the project characteristic factor
θ_{j}:	the change-point
$m(t)$:	the mean value function of the software error-detection process, which is the expected number of errors detected within time $(0, t)$
$m_1(t)$:	the mean value function with exponential learning effects

The concept of this study is that with time, the potential errors will become the detected errors using the testing/debugging process and that the count depends on the autonomous errors-detection factor and the learning effects, the former related to the initial condition of testing staff and the latter related to learning effects from the testing/debugging process which vary with time. Sometimes, the potential errors will vary unexpectedly, and the count can be either positive or negative and may fluctuate with time but remain close to zero, progressively.

3.1.2 Time-Varying Learning Effects

Learning effects mean when staffs are involved in a production or service activity, the operation will be practiced. Many researchers have noted that learning effects exist in the software testing/debugging process. Chiu *et al.*, 2008 constructed an SRGM with constant learning effects. In practice, the learning effects will vary with time and will include both linear and exponential learning effects induced in later SRGMs [1]. In this study, the exponential learning function was adopted to represent the exponential growth of learning effects with time, and the function is given as (1). The factor ξ is the coefficient of the accelerative effect:

$$\eta(t) = \eta e^{\xi t} \tag{1}$$

3.1.3 Variable Potential Errors

In this study, perfect debugging means that each time an error occurs, the fault that caused it can be immediately removed and that no new errors will be introduced. Imperfect debugging means that an error being removed will deal with or induce another error unconsciously, which implies that potential errors will vary with testing/debugging time.

In practice, staff will either include new errors or deal with other errors unconsciously when they work, which will change the rate of potential errors. The rate of extra variation to potential errors should be considered in an imperfect debugging process, as wrongly fixing an error may cause more errors; while correctly debugging an error may resolve other errors, and this number will gradually converge as the testing staff becomes more familiar with the software system. Accordingly, an imperfect debugging method will have an impact on the rate of potential errors. Suppose that the rate of extra variation to potential errors can be modelled by a sine function [2] as it is able to describe the fluctuating nature of the debugging process within the software testing period. By employing a parameter of the fluctuating factor ν to exponentially adjust the decreasing amplitude and a parameter δ to describe the characteristic of the software project, the rate of variation can be given as (2):

$$r(t) = \mathrm{e}^{-\nu t} \cdot \sin\left(\frac{\pi}{2}\delta t\right) \tag{2}$$

3.1.4 Change-Point

Sometimes, a software testing/debugging project will experience changes in the testing condition that will cause variations in the testing performance, and these situations are considered with change-point in this research. Accordingly, the time points while the testing performance goes through a critical change will be the change-points that will be judged by variations in the value of $r(t)$, as in (2). For example, the parameters $\alpha, \eta, \xi, \nu, \delta$, and a in this model can be obtained by using numerical methods with the historical data in the data set [1] and with the curve of the $r(t)$ drawn as shown in Fig. 1. As we can see, the curve of the value of the rate of extra variation to potential errors was diverted from increasing to decreasing at the 6th week and diverted to increasing at the 11th week. We can therefore assume the software testing/debugging project must experience changes at the 6th and 11th weeks, then modified the parameters α_i, η_i, and ξ_i in these three periods: before the 6th week, during the 6th week through the 11th week, and after the 11th week.

3.2 Model Development

According to Chiu's (2008) model, the mean value function of the software error-detection process can be written as (3), which assumes the learning effects are constant:

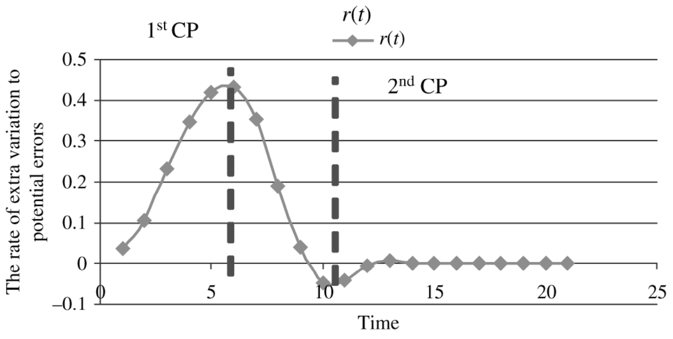

Figure 1. The variations in the value of $r(t)$ to judge the change-points.

$$m(t) = aH(t) = a\left(1 - \frac{1+\dfrac{\eta}{\alpha}}{\dfrac{\eta}{\alpha} + \mathrm{e}^{(\alpha+\eta)t}}\right) \tag{3}$$

In practice, the learning effects will vary with time, and this function is given as (1), respectively.

The mean value function (3) can be improved with this learning style as given by:

$$m_1(t) = a\left(1 - \frac{1+\dfrac{\eta e^{\xi t}}{\alpha}}{\dfrac{\eta e^{\xi t}}{\alpha} + \mathrm{e}^{(\alpha+\eta e^{\xi t})t}}\right) \tag{4}$$

The learning effects will be constant while $\xi = 0$, and the mean value function (4) will degenerate to (3) flexibly.

Further, the staff removing an error will potentially induce new errors or deal with other errors unconsciously, and this will change the rate of potential errors, which will cause additional variations in the cumulative function. The rate of extra variations in the cumulative function can be given as (2), as explained previously. Accordingly, the mean value functions of the software error-detection process can be improved with (3) as given by (5):

$$\begin{aligned} m_2(t) \;=\; aH(t)\,(1+r(t)) = a\cdot &\left(1 - \frac{1+\dfrac{\eta e^{\xi t}}{\alpha}}{\dfrac{\eta e^{\xi t}}{\alpha} + e^{(\alpha+\eta e^{\xi t})t}}\right) \\ &\times \left(1 + \left(\mathrm{e}^{-\nu t}\cdot \sin\left(\frac{\pi}{2}\delta t\right)\right)\right), \end{aligned} \tag{5}$$

When the fluctuating factor $\nu = \infty$ or the project characteristic factor $\delta = 0$, the number of potential errors will be a constant, and (5) will degenerate to (4).

Table 1
Data Set References

Label	Reference	Data Sets
[1]	[26]	Failure data of the space program
[2]	Huang and Hung (2011)	The DS 1

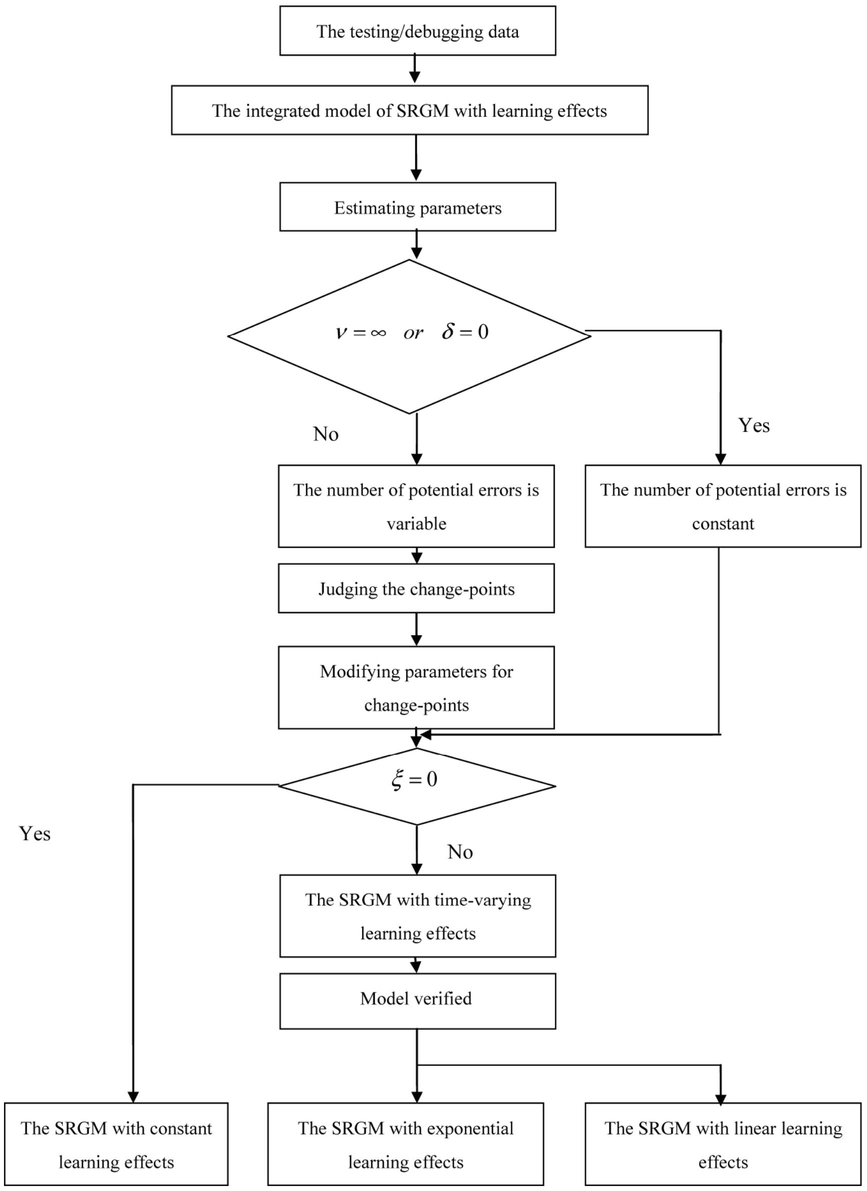
The testing/debugging data
The integrated model of SRGM with learning effects
Estimating parameters
ν = ∞ or δ = 0
No
Yes
The number of potential errors is variable
The number of potential errors is constant
Judging the change-points
Modifying parameters for change-points
ξ = 0
Yes
No
The SRGM with time-varying learning effects
Model verified
The SRGM with constant learning effects
The SRGM with exponential learning effects
The SRGM with linear learning effects

Figure 2. The concept of the integrated models.

As the learning effects can change over time, the proposed model can provide more flexibility in regard to the graphic mapping of the mean value function.

The parameters $\alpha, \eta, \xi, \nu, \delta$, and a can be obtained using either least squares estimation (LSE) or numerical methods.

3.3 Model Verification

This study evaluates the effectiveness of the proposed models by using the data sets in [26] and Huang and Hung (2010) (Table 1). This study investigates the effectiveness of the proposed models by using the ***MSE***, ***R2,*** and relative root mean square error (RRMSE) comparison criteria. The equations as follows:

$$MSE = \frac{\sum_{i=1}^{n} (m_i - m(t_i))^2}{n-k} \tag{6}$$

$$RRMSE = \frac{\sqrt{\frac{1}{n}\sum_{i=1}^{n} (m_i - m(t_i))^2}}{\frac{1}{n}\sum_{i=1}^{n} (m_i)} \tag{7}$$

Least Squares Estimation (LSE):

$$\min M_2(a, \nu, \delta, \alpha, \eta, \xi) = \sum_{i=1}^{n} (m_i - m_2(t_i))^2$$
$$= \sum_{i=1}^{n} \left(m_i - a \cdot \left(1 + e^{-\nu t} \cdot Sin\left(\frac{\pi}{2}\delta t\right)\right) \cdot \left(1 - \frac{1+E}{E+F}\right)\right)^2 \tag{8}$$

$$\text{Note that} \begin{cases} E = \frac{\eta e^{\xi t}}{\alpha} \\ F = e^{(\alpha t + \eta t e^{\xi t})} \end{cases}$$

3.4 The Integrated Model of SRGM

Above of all, this study includes a sine function to construct an integrated model of SRGM with time-varying learning effects for considering variable potential errors and multiple change-points. The operating process of this model is shown in Fig. 2. This study estimates parameters for testing/debugging data by the proposed model, when the fluctuating factor $\nu = \infty$ or the project characteristic factor $\delta = 0$, (13) will degenerate to (9) and (14) will degenerate to (10), where the number of potential errors will be a constant, judges the numbers and the time point of the change-points with the trace of $r(t)$, modifies parameters for change-points, when the accelerative factor with time of learning effect $\xi = 0$, (9) and (10) will degenerate to (7) where the learning effects will be a constant.

4. Results

This study includes three actual data sets to verify the proposed integrated models. The results show good fitting for the actual data sets and describe the software testing process reasonably.

4.1 The Fitting Results for Data Set [1]

According to the proposed integrated models, the change-points for the data sets are judged by the proposed sine function, $r(t)$ which is the rate of extra variation to potential errors. As the unstable testing environment and the level of debugging skill will cause extra variations in potential errors unconsciously during the initial testing process, the sine function $r(t)$ can describe this phenomenon and estimate parameters α_j, η_j, ξ_j for each phase using the partial data between change-points, respectively, and then can modify the parameters and change-points.

Figure 3 shows the $r(t)$ curve and the predicted change-points. Figure 4 shows the fitting results, and Table 2 provides the parameters for data set [1] with the proposed model built with exponential learning effects.

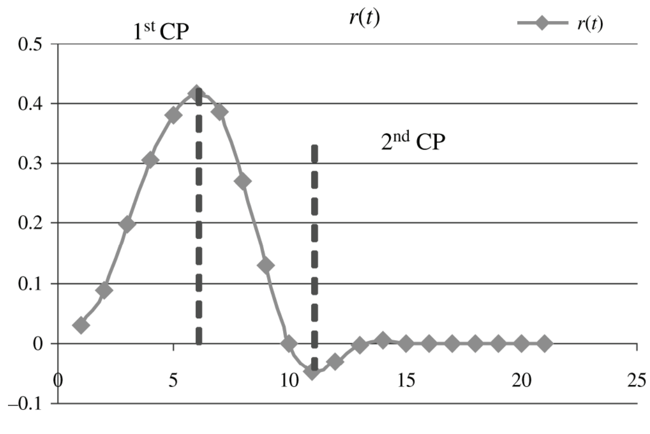

Figure 3. The variations of $r(t)$ with predicted change-points for data set [1] using the exponential learning model.

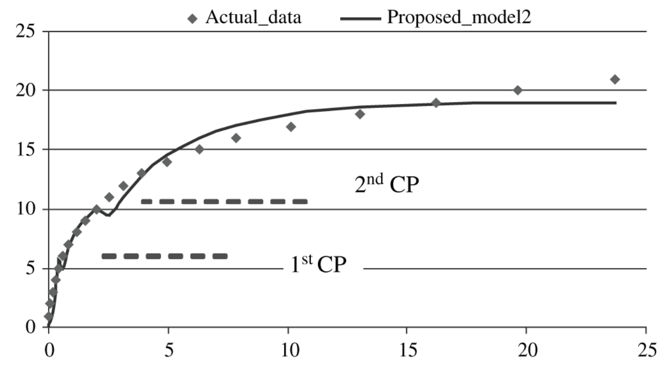

Figure 4. Fitting results for data set [1] using the exponential learning model.

Table 2
The Parameters for Data Set [1] with the Proposed Model

Parameters		1st Change-Point ($\theta_1 = 6$)	2nd Change-Point ($\theta_2 = 11$)
$a = 19$	$\alpha_0 = 0.32$	$\alpha_1 = 0.35$	$\alpha_2 = 0.29$
$\nu = 1.1$	$\eta_0 = 2.8$	$\eta_1 = 0.07$	$\eta_2 = 0.000001$
$\delta = 1$	$\xi_0 = 0$	$\xi_1 = 0.01$	$\xi_2 = 0$

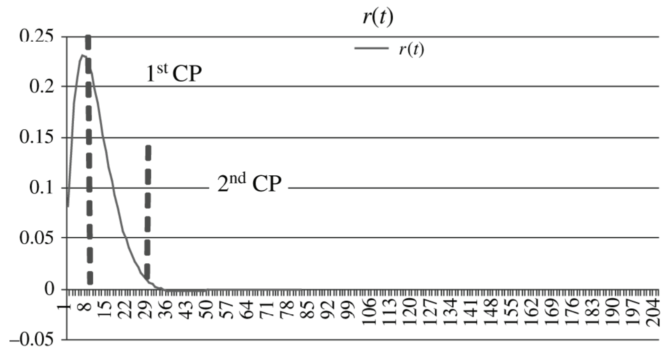

Figure 5. The variations of $r(t)$ with predicted change-points for data set [2] using the exponential learning model.

4.2 The Fitting Results for Data Set [2]

Figure 5 shows the $r(t)$ curve and the predicted change-points. The fitting results are shown in Fig. 6, and Table 3 gives the parameters for data set [2] with the proposed model built with exponential learning effects.

4.3 Comparison of the Results

This paper compares its fitting results for data set [1] with those of the models in published papers by using the MSE and R^2 comparison criteria (Table 4). The proposed models are more concise and effective, and the exponential learning model is more suitable for data set [1] with two change-points. The parameters show that the learning effects exhibit exponential growth from time 6 to 11; otherwise, they remain constant.

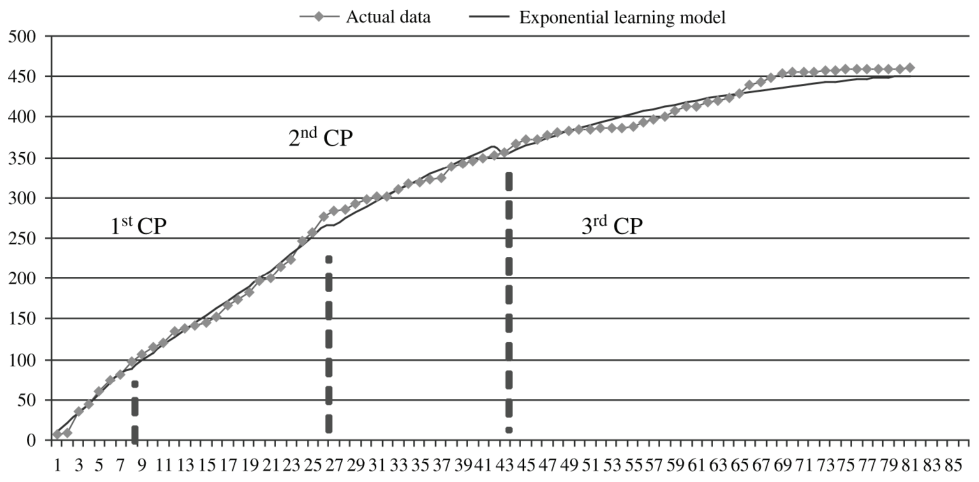

Figure 6. Fitting results for data set [2] using the exponential learning model.

Table 3
The Parameters for Data Set [2] with the Proposed Model

Parameters		$\theta_1 = 8$	$\theta_2 = 27$	$\theta_3 = 43$
$a = 465$	$\alpha_0 = 0.02$	$\alpha_1 = 0.02$	$\alpha_2 = 0.02$	$\alpha_2 = 0.02$
$\nu = 0.14$	$\eta_0 = 0.01$	$\eta_1 = 0.01$	$\eta_2 = 0.038$	$\eta_2 = 0.029$
$\delta = 0.06$	$\xi_0 = 0.19$	$\xi_1 = 0.054$	$\xi_2 = 0$	$\xi_2 = 0.0025$

Table 4
Fitting Results Using the MSE and R^2 Comparison Criteria for Data Set [1]

Criteria	Goel and Okumoto [27]	Yamada (1983)	Ohba [6]	Yamada (1992)	Huang [7]	Pham and Zhang [28]	Chiu (2008)	The Proposed Model
MSE	2.685	6.593	3.437	1.946	2.428	4.122	3.010	1.380
R^2	0.934	0.837	0.920	0.955	0.953	0.914	0.930	0.973

Table 5
Fitting Results Using the MSE and $RRMSE$ Comparison Criteria for Data Set [2]

Model	MSE	$RRMSE$
Huang and Hung (2011)	87.09	0.030
ISS with two change-points	92.56	0.030
Yamada delayed S-shaped (DSS) model	426.93	0.066
Goel–Okumoto (G–O) model	102.39	0.032
Goel–Okumoto (G–O) model (with two change-points)	94.44	0.030
The proposed model	87.94	0.029

This study compares its fitting results for data set [2] with those of other models in published papers by using the MSE and $RRMSE$ comparison criteria (Table 5). The proposed models are more concise and effective, and the exponential learning model is more suitable for data set [2] with three change-points. The parameters show the learning effects to be constant from week 27 to 43, and to otherwise exhibit exponential growth.

5. Conclusion

Recently, various SRGMs have been proposed to assess software reliability. Some important issues in such models include the S-curve or exponential testing data, the change-points in the process, estimation of the parameters, variations in potential errors, and the learning effects occurring in the testing process. However, the models which were proposed in the past may be suitable only for a specific S-curve or exponential testing data, may determine the change-points but not describe the change, and may discuss the learning effects but not evaluate them, thus narrowing the scope of applications. This study includes three actual data sets to verify the proposed integrated model, and the results show good fitting for the actual data sets and describe the software testing process reasonably. The proposed integrated model describes the software testing process accurately, which helps in regard to managing the testing project effectively. Equation (3) can simultaneously describe both S-shaped and exponential-shaped types of testing behaviour with constant learning effects, and (4) illustrates software testing behaviour with time-varying learning effects that help to effectively determine the learning effects in the testing process. Equation (2) judges the umber and the time point of change-points and illustrates the variations of the parameters on the change-points that help to master the software testing project. Equation (5) traces the change-points in the testing project and explains both S-shaped and exponential-shaped types of testing behaviour with time-varying learning effects by considering variable potential errors that help staff become acquainted with the software testing environment. The proposed model not only provides good numerical prediction performance for several different kinds of data but also explains the testing/debugging behaviour of testing staff, the learning effects of the testing project, and the changes in the testing environment, all of which are important considerations of software system testing and management.

References

[1] K.C. Chiu, An improved model of software reliability growth under time-dependent learning effects, *Proc. 2011 IEEE ICQR*, 2011, Bangkok, 191–194.

[2] I.-C. Huang, *The software reliability growth model with imperfect debugging*, The Master Dissertation of Department of Industrial and Information Management, National Cheng Kung University, 2011.

[3] M. Ohba, Software reliability analysis models, *IBM Journal of Research and Development, 28*, 1984b, 428–443.

[4] K. Maxwell, L.V. Wassenhove, and S. Dutta, Performance evaluation of general and company specific models in software development effort estimation, *Management Science, 45*, 1999, 787–803.

[5] S. Yamada, M. Ohba, and S. Osaki, S-shaped reliability growth modeling for software error detection, *IEEE Transactions on Reliability, R-32*(5), 1983, 475–479.

[6] M. Ohba, Inflexion S-shaped software reliability growth models, in S. Osaki and Y. Hatoyama (eds.), *Stochastic models in reliability theory*, (Berlin, Germany: Springer-Verlag, 1984a), 144–162.

[7] C.-Y. Huang, Performance analysis of software reliability growth models with testing-effort and change-point, *Journal of Systems and Software, 76*, 2005, 181–194.

[8] M. Xie and B. Yang, A study of the effect of imperfect debugging, *Journal of Systems and Software, 29*(5), 2003, 471–473.

[9] G. Mosheiov, Scheduling problems with a learning effect, *European Journal of Operational Research, 132*, 2001, 687–693.

[10] P. Kapur, D. Goswami, A. Bardhan, and O. Singh, Flexible software reliability growth model with testing effort dependent learning process, *Applied Mathematical Modelling, 32*(7), 2008, 1298–1307.

[11] K.C. Chiu, Y.S. Huang, and T.Z. Lee, A study of software reliability growth from the perspective of learning effects, *Reliability Engineering & System Safety, 93*(10), 2008, 1410–1421.

[12] K.C. Chiu, A study of software reliability growth model for time-dependent learning effects, *Proc. 2012 IEEE IEEM*, Hong Kong, 2012, 1015–1019.

[13] M. Zhao, Change-point problems in software and hardware reliability, *Communications in Statistics Theory and Methods*, 22, 1993, 757–768.

[14] H. Shyur, A stochastic software reliability model with imperfect-debugging and change-point, *Journal of Systems and Software*, *66*(2), 2003, 135–141.

[15] C.-Y. Huang and T.-Y. Hung, Software reliability analysis and assessment using queuing models with multiple change-points, *Computers and Mathematics with Applications*, *60*, 2010, 2015–2030.

[16] C.-Y. Huang and M.R. Lyu, Optimal testing resource allocation, and sensitivity analysis in software development, *IEEE Transactions on Reliability*, *54*(4), 2005, 592–603.

[17] S. Inoue, K. Fukuma, and S. Yamada, Two-dimensional change-point modeling for software reliability assessment, *International Journal of Reliability, Quality and Safety Engineering*, *17*(6), 2010, 531–542.

[18] P. Kapur, A. Kumar, and K. Yadav, Software reliability growth modeling for errors of different severity using change point, *International Journal of Reliability, Quality and Safety Engineering*, *14*(4), 2007, 311–326.

[19] C.G. Bai, Bayesian network based software reliability prediction with an operational profile, *The Journal of Systems and Software*, *77*, 2005, 103–112.

[20] X. Zhang, X. Teng, and H. Pham, Considering fault removal efficiency in software, *IEEE Transactions on Systems, Man, and Cybernetics – Part A: Systems and Humans*, *33*(1), 2003, 114–120.

[21] X. Zhang and H. Pham, Software field failure rate prediction before software deployment, *Journal of Systems and Software*, *79*(3), 2006, 291–300.

[22] J. Lo and C. Huang, An integration of fault detection and correction processes in software reliability analysis, *Journal of Systems and Software*, *79*(9), 2006, 1312–1323.

[23] M. Xie, Q.P. Hu, Y.P. Wu, and S.H. Ng, A study of the modeling and analysis of software fault-detection and fault-correction processes, *Quality and Reliability Engineering International*, *23*(4), 2007, 459–470.

[24] N. Ahmad, M.G.M. Khan, and L.S. Rafi, A study of testing-effort dependent inflection S-shaped software reliability growth models with imperfect debugging, *International Journal of Quality & Reliability Management*, *27*(1), 2010, 89–110.

[25] S. Dick, C.L. Bethel, and A. Kandel, Software-reliability modeling the case for deterministic behavior, *IEEE Transactions on Systems, Man, and Cybernetics – Part A: Systems and Humans*, *37*(1), 2007, 106–119.

[26] C.G. Bai, Q.P. Hu, M. Xie, and S.H. Ng, Software failure prediction based on a Markov Bayesian network model, *The Journal of Systems and Software*, *74*, 2005, 275–282.

[27] A.L. Goel and K. Okumoto, Time dependent fault detection rate model for software and other performance measures, *IEEE Transactions on Reliability*, *28*(2), 1979, 206–211.

[28] H. Pham and X. Zhang, NHPP software reliability and cost models with testing coverage, *European Journal of Operational Research*, *145*, 2003, 443–454.

[29] C.-Y. Huang and C.T. Lin, Analysis of software reliability modeling considering testing compression factor and failure-to-fault relationship, *IEEE Transactions on Computers*, *59*(2), 2010, 283–288.

[30] C.-Y. Huang and M.R. Lyu, Estimation and analysis of some generalized multiple change-point software reliability models, *IEEE Transactions on Reliability*, *60*(2), 2011, 498–513.

[31] D.R. Jeske, X. Zhang, and L. Pham, Adjusting software failure rates that are estimated from test data, *IEEE Transactions on Reliability*, *54*(1), 2005, 107–115.

[32] C. Lee, Y. Kim, and D. Park, S-shaped software reliability growth models derived from stochastic differential equations, *IIE Transactions*, *36*(12), 2004, 1193–1199.

[33] H. Pham, An imperfect-debugging fault-detection dependent-parameter software, *International Journal of Automation and Computing*, *4*(4), 2007, 325–328.

[34] H. Pham and X. Zhang, A general imperfect-software-debugging model, *IEEE Transactions on Reliability*, *48*(2), 1999, 169–175.

[35] S. Yamada, M. Ohba, and S. Osaki, s-shaped software reliability growth models and their applications, *IEEE Transactions on Reliability*, *R-33*(4), 1984, 289–293.

[36] J. Zhao, H. Liu, G. Cui and X. Yang, Software reliability growth model with change-point and environmental function, *Journal of Systems and Software*, *79*(11), 2006, 1578–1587.

Biography

Kuei-Chen Chiu is a researcher of National Cheng Kung University. She earned both her first M.S. and Ph.D. degrees from Industrial and Information Management of National Cheng Kung University. Her research interests include software reliability, operations management, human factor, human resource management, and performance assessment. Related papers have appeared in professional journals as such as *Reliability Engineering and System Safety* and *Software Quality Journal.* She has also earned the second M.S. degree from the Department of Counseling, National Chiayi University and now has the second Ph.D. degree grogram proceeding in the Institute of Allied Health Sciences, College of Medicine, National Cheng Kung University. Her interdisciplinary researches include cognitive psychology, industrial and organizational psychology, psychological assessment, and psychological testing.