Confidence Interval Estimation of Software Reliability Growth Models based on Ohba's Inflection S-shaped Model

Tean-Quay Lee and Chih-Chiang Fang
Department of Information Management, Shu-Te University, Taiwan, R.O.C.
Email: tqlee@stu.edu.tw; ccfang@stu.edu.tw

Chun-Wu Yeh
Department of Information Management, Kun Shan University, Taiwan, R.O.C.
Email: davidyeh929@gmail.com

***Abstract*—Software reliability growth models with confidence intervals (SRGMs) are often utilized in software industry in that they provide useful information for software developers to decide the optimal software release time and to refine the quality of software testing tasks. Most SRGMs, nonetheless, do not have transparent explanations for the variance estimation of cumulative software errors. They might not be effective in deducing the confidence interval regarding the mean value function. In such cases, software developers cannot estimate the possible risk variation of software system by using the randomness of mean value function, and it might debase the practicability of applications. In this paper, we utilize Ohba's Inflection S-shaped model to build the SRGM with confidence intervals that can assist the software developers in determining the optimal release time.**

***Index Terms*—Software Reliability Growth Model; software reliability; confidence intervals; stochastic differential equations; Non-homogeneous Poisson process**

I. Introduction

Software reliability engineering is critical in software industry, since it can offer products with high performance and provide useful information for software developers and testing staff during the testing/debugging phase. How to improve the reliability and reduce the cost of software systems is the main issue of the software industry [1]. Software developers need to decide the optimal software release time, refine the quality of software testing tasks, and control related software testing/debugging cost by means of software reliability growth models (SRGMs) to ensure the software functionality. Generally, these software reliability growth models in previous literature are discussed in several kinds of probability distributions such as Exponential-shaped, S-shaped, and a mix of the two [2]. The software failure phenomena are mostly examined based on the Non-homogeneous Poisson Process (NHPP). The exponential SRGM that Goel and Okumoto [1] proposed is effective to portray the fault-detection process of software. In addition, quite a few of generalizations or modifications of SRGMs have been proposed in that the curve of cumulative number of faults detected is often associated with Exponential-shaped or S-shaped mean value functions. For instance, the delayed S-shaped reliability growth models proposed by Yamada *et al*. [3] and Yamada [4]. An inflection S-shaped reliability growth model was proposed by Ohba [5] and [6]. Pham and Zhang [7] proposed a software reliability model that combines testing coverage measures to assess the software reliability. Huang [8] joins a generalized logistic testing-effort function and the change-point parameter for analyzing system performance in their proposed model. These models may be applicable to specific types of software failure data but not considering the learning effect during the software-debugging process. If the software companies cannot afford more budget for testing and debugging, the raise of software reliability is rely on the learning effect of software developers. The learning effect here signifies that the debugging efficiency of software developers increases in the process due to the experience of detecting previous software errors without purchasing new equipments or introducing new techniques. Besides, the cost trade-off of fault detection and software release time affects the competitiveness of the software companies. How to correctly estimate the interval of mean value functions is the main focus of this study. Since the number of software errors is finite, there should be less software errors to be detected with time. The variance of cumulative software errors should be decreasing instead of increasing with time. Tamura and Yamada [9] proposed the concept that the variance of the mean value function stem from the error detection process and used stochastic differential equations method (SDE) to deduce the mean value function. These mentioned models still have some problems even though SDE is effective in evaluating the mean value function. For example, Yamada et al. [10] proposed a simple SRGM by applying an Itô type of SDE where the error detection rate is constant, but the error detection rate

Manuscript received June 17,2013; revised August 30, 2013.

doi: 10.12720/jiii.1.4.196-200

would vary over time in practice. Lee et al. [11] developed an SRGM by using an Itô type SDE based on the delayed S-shaped and the inflection S-shaped models respectively proposed by Yamada et al. [3] and Ohba [6], without considering the variance in the mean value function. This may be an important factor to measure testing costs during the testing phase. Tamura and Yamada [9] derived a flexible SDE model to describe the fault-detection process by using an inflection S-shaped SRGM and an Itô type SDE [12], but such a model improperly presents the expectation and variance of the mean value function, and lacks a complete process for obtaining the parameters of the model.

Hence, discovering the confidence intervals of software reliability can enhance the decision of software releases and control the related expenditures for software testing. Yamada and Osaki [13] considered that the maximum likelihood estimates (MLE) concerning the confidence interval of the mean value function can be estimated by traditional NHPP method. Yin and Trivedi [14] obtained the confidence bounds for the model parameters via the Bayesian approach by quoting the estimation method of Yamada and Osaki [13]. Huang [8] also adopted the method of Yamada and Osaki [13] to draw a graph to illustrate the confidence interval of the mean value function.

The possible reason of existing improper variances lies in the misconception that merely the randomness of the mean value function should account for the final estimation error of software reliability. This turns out that the variance of mean value function will increase in time. However, since the number of software errors is finite, there should be less software errors to be detected with time. Therefore, the variance of cumulative software errors should be decreasing instead of increasing as time goes by. In this regard, it is assumed that the variance in the mean value function mainly stems from the error detection process in this study, and the confidence interval of the mean value function would thus diminish as remaining software errors decrease.

According to the above discussion, several SRGMs with confidence intervals by using the method of SDE will be proposed in this study. The proposed SRGMs with confidence intervals could assess the variance in the mean value function more reasonably, and assist software developers in properly dealing with the risks of software reliability estimation.

II. Model Development

Various software reliability growth models have been proposed in the last two decades, and some of these models were fairly effective in prediction of the average number of cumulative software errors during the testing task. These models based on non-homogeneous Poisson process include Goel and Okumoto's model (1979) [15], Yamada's delayed S-shaped model (1983) [3], Ohba's inflection S-shaped model (1984) [6], Musa exponential model (1985) [16], Chiu and Haung's learning effect model (2008) [17]. Table I shows the corresponding mean value functions and the error detection rate functions for the five classic models.

TABLE I. Summary of Mean Value and Error Detection Rate Functions for the Classic Models

Model	Mean value function and error detection rate function
Goel and Okumoto's model (1979)[15]	$m(t) = a(1-e^{-bt})$ $d(t) = b$
Yamada's Delayed S-shaped model (1983)[3]	$m(t) = a(1-(1+bt)e^{-bt})$ $d(t) = \frac{b^2 t}{(1+bt)}$
Ohba's Inflection S-shaped model (1984)[6]	$m(t) = a\left(\frac{1-e^{-bt}}{1+ce^{-bt}}\right)$ $d(t) = \frac{b}{(1+ce^{-bt})}$
Musa Exponential Model (1985)[16]	$m(t) = a\left(1-e^{-\left(\frac{ct}{nL}\right)}\right)$ $d(t) = \alpha\beta r e^{-\beta t}$
Chiu and Haung's learning effect model (2008)[17]	$m(t) = a\left(1-\frac{1+\frac{\eta}{\alpha}}{\frac{\eta}{\alpha}+e^{(\alpha+\eta)t}}\right)$ $d(t) = (\alpha+\eta)\left(1-\frac{\eta}{\alpha e^{(\alpha+\eta)t}+\eta}\right)$

The following notations will be used to deduce the model of this study:

Notations:

a: the expected number of all potential errors in the software system before the software testing begins

$m(t)$: the mean value function of the software error detection process, which is the expected number of errors detected within time $(0,t)$

$M(t)$: the function of the residual software errors at time t

$d(t)$: the error detection rate per error at time t

$W(t)$: the variable for Wiener process

σ: the standard deviation stems from testing process, and it is a positive constant to denote the magnitude of the irregular fluctuation.

Due to the fact that the above-mentioned models in Table I lack of the discussion about the corresponding confidence intervals, the decision makers cannot evaluate the possible variation in the mean value function itself during the testing task. Accordingly, we use the models in Table I as basis to develop their corresponding confidence intervals of the mean value function.

According to the behavior of testing and debugging, we assume the irregular fluctuation is influenced by the error detection rate, and it can be represented by the following stochastic differential equation:

Let $M(t)=a-m(t)$, $d(t)=\dfrac{b}{1+ce^{-bt}}$

$$\frac{\mathrm{d}M(t)}{\mathrm{d}t}=-\big(d(t)+\sigma dW(t)\big)M(t)$$

$$\frac{\dfrac{\mathrm{d}M(t)}{\mathrm{d}t}}{M(t)}=-\big(d(t)+\sigma dW(t)\big)$$

Let $Z(t)=\ln\big[M(t)\big]$

According Itô's calculus:

$$\mathrm{d}Z(t)=\left\{-d(t)-\frac{1}{2}(-\sigma)^2\right\}\mathrm{d}t-\sigma\mathrm{d}W(t)$$

$$\int_0^T\mathrm{d}Z(t)=\int_0^T-d(t)\mathrm{d}t-\int_0^T\frac{1}{2}\sigma^2\mathrm{d}t-\int_0^T\sigma\mathrm{d}W(t)$$

$$\because\ \int_0^T-d(t)\mathrm{d}t=\int_0^T\frac{-b}{1+ce^{-bt}}\mathrm{d}t=\ln\left[\frac{(1+c)e^{-bT}}{1+ce^{-bT}}\right]$$

$$\therefore\int_0^T\mathrm{d}Z(t)=\ln\left[\frac{(1+c)e^{-bT}}{1+ce^{-bT}}\right]-\int_0^T\frac{1}{2}\sigma^2\mathrm{d}t-\int_0^T\sigma\mathrm{d}W(t)$$

$$\ln\big[M(T)\big]=$$

$$\ln\left[\frac{(1+c)e^{-bT}}{1+ce^{-bT}}\right]-\int_0^T\frac{1}{2}\sigma^2\mathrm{d}t-\int_0^T\sigma\mathrm{d}W(t)+\text{Constant}$$

$$M(T)=\frac{(1+c)e^{-bT}}{1+ce^{-bT}}e^{-\frac{1}{2}\sigma^2T-\sigma w(T)+\text{Constant}}$$

$$E\big[M(T)\big]=E\left[\frac{(1+c)e^{-bT}}{1+ce^{-bT}}e^{-\frac{1}{2}\sigma^2T-\sigma W(T)+\text{Constant}}\right]$$

$$=\frac{(1+c)e^{-bT}}{1+ce^{-bT}}e^{\text{Constant}-\frac{1}{2}\sigma^2T}E\left[e^{-\sigma W(T)}\right]$$

$$\because\ E\left[e^{-\sigma W(T)}\right]=\int_{-\infty}^{\infty}e^{-\sigma x}\frac{1}{\sqrt{2\pi T}}e^{-\frac{x^2}{2T}}\mathrm{d}x=e^{\frac{1}{2}\sigma^2T}$$

$$\therefore E\big[M(T)\big]=\frac{(1+c)e^{-bT}}{1+ce^{-bT}}e^{\text{Constant}-\frac{1}{2}\sigma^2T}e^{\frac{1}{2}\sigma^2T}$$

$$=\frac{(1+c)e^{-bT}}{1+ce^{-bT}}e^{\text{Constant}}$$

$\because$ Initial condition: $M(0)=a$

$\therefore$ Constant=$\ln\big[a\big]$

$$\therefore E\big[M(T)\big]=a\left(\frac{(1+c)e^{-bT}}{1+ce^{-bT}}\right)$$

$$\therefore E\big[m(T)\big]=E\big[a-M(T)\big]=a\left(\frac{1-e^{-bT}}{1+ce^{-bT}}\right)$$

$$Var\big[m(T)\big]=Var\big[M(T)\big]=E\big[M(T)^2\big]-E\big[M(T)\big]^2$$

$$E\big[M(T)^2\big]=E\left[a^2\left(\frac{(1+c)e^{-bT}}{1+ce^{-bT}}\right)^2e^{-\sigma^2T-2\sigma W(T)}\right]$$ (It is deduced by Itô's calculus under the initial condition: $M(0)^2=a^2$)

$$\because\ E\left[e^{-2\sigma W(t)}\right]=\int_{-\infty}^{\infty}e^{-2\sigma x}\frac{1}{\sqrt{2\pi t}}e^{-\frac{x^2}{2t}}\mathrm{d}x=e^{2\sigma^2T}$$

$$E\big[M(T)^2\big]=a^2\left(\frac{(1+c)e^{-bT}}{1+ce^{-bT}}\right)^2e^{\sigma^2T}\tag{1}$$

$$Var\big[m(T)\big]=Var\big[M(T)\big]$$

$$=E\big[M(T)^2\big]-E\big[M(T)\big]^2$$

$$=a^2\left(\frac{(1+c)e^{-bT}}{1+ce^{-bT}}\right)^2\left(e^{\sigma^2T}-1\right)\tag{2}$$

According to the above Equations, we have got the general forms of the expectation $E\big[m(T)\big]$ and the variance $Var\big[m(T)\big]$ of the mean value function.

III. Estimation of Parameters

Since the above model involve unknown parameters which can be estimated from the observed data, the method of maximum likelihood is utilized to estimate the values of unknown parameters. The set of paired data (T_i,m_i) can be collected in practice, where m_i is the total number of errors detected until T_i. Also, suppose that the unknown parameters of the specified SRGM are determined by the $n+1$ observed paired data: $(T_0,m_0),(T_1,m_1),(T_2,m_2),...,(T_n,m_n)$, then the likelihood function for the SRGMs can be expressed as

$$L=\Pr\{N(T_1)=m_1,N(T_2)=m_2,N(T_3)=m_3,...,N(T_n)=m_n\},$$

$$=\prod_{i=1}^{n}\frac{\big(m(T_i)-m(T_{i-1})\big)^{(m_i-m_{i-1})}\left(e^{-(m(T_i)-m(T_{i-1}))}\right)}{(m_i-m_{i-1})!}\tag{3}$$

Taking a logarithm of the likelihood function, the MLE of the unknown parameters can be obtained by numerically solving the simultaneous equation

$$\frac{\partial\ln(L)}{\partial(\text{unknown parameter 1})}=\frac{\partial\ln(L)}{\partial(\text{unknown parameter 2})}$$

$$=\ldots=\frac{\partial\ln(L)}{\partial(\text{unknown parameter k})}=0.$$

IV. Estimation of Confidence Intervals

In this section, the relevant confidence intervals of the SRGMs are developed; the confidence interval of the mean value function can be developed using Equations (1) and (2), which is given by

$$E\big[m(T)\big]\pm t_{CR/2,n-k}\sigma[m(T)]\sqrt{1+\frac{1}{n}+\frac{\big(T-\bar{t}\big)^2}{\sum_{i=1}^{n}\big(t_i-\bar{t}\big)^2}},\tag{4}$$

where CR denotes the critical region, $t_{CR/2,n-k}$ denotes the value providing an area $CR/2$ of the Student-t distribution with $n-k$ degrees of freedom.

However, in the classical model, owing to the different expression of $\hat{\sigma}^2$, the confidence intervals that are correspondent with Equation (4) would be somewhat dissimilar. For instance, in the classical model, the confidence interval of the mean value function is expressed as

$$E\left[m(T)\right] \pm t_{CR/2,n-k} \sqrt{\left(\frac{\sum_{i=1}^{n}\left(m_i - m(t_i)\right)^2}{n-k}\right)\left(1+\frac{1}{n}+\frac{\left(T-\bar{t}\right)^2}{\sum_{i=1}^{n}\left(t_i-\bar{t}\right)^2}\right)}. \quad (5)$$

Since the classical model assumes that the variance in mean value functions stems from $m(T)$, whereas the proposed model assumes that the variance in mean value functions stems from $d(T)$, confidence intervals in the classical model would be divergent in the late testing phase, but convergent for the proposed model. This can be explained by the fact that the possibility of finding new software errors becomes lower due to less software errors remaining in the late testing phase. In such a case, the variance in mean value functions turns out to be fewer in the proposed model, which is opposite to the case in the classical model.

V. Decision for Optimal Software Release

In practice, software developers would like to know when the software testing should be stopped so that the related costs can be minimized and the requirement of software quality can be met. In general, the longer the testing time the more reliable the software. However, a longer testing time will increase costs and lead to the loss of commercial opportunities. Therefore, optimal release policies are of practical importance for software developers.

In order to minimize the total testing cost and meet the minimal requirement of software reliability, the optimal software release model can be formulated as:

$$\begin{aligned} &Min\ E\left[C(T)\right] = C_0 + C_1 T + C_2 E\left[m(T)\right] u_y \\ &\qquad + C_3\left(1 - R(x/T)\right) + C_4\left(v_1 + T\right)^{v_2}, \quad (6) \\ &Subject\ to:\ R(x/T) \geq R_0 \end{aligned}$$

where C_0 is the set-up cost for software testing, C_1 is the software routine cost per unit time, C_2 is the cost of removing an error per unit time during testing, C_3 is the loss due to software failure, $C_4\left(v_1+T\right)^{v_2}$ is the loss of commercial opportunities due to postpone software release. R_0 is the minimum requirement of software reliability for avoiding a premature software release.

Note that the optimal software release time can be determined using Equation (6), but it only fits for an average case. However, in the worst case scenario, software reliability might not reach the expected level if the software debugging is not effective. Therefore, software developers should give a conservative estimation of software reliability. Consequently, the mean value function $E\left[m(T)\right]$ and the software reliability function $R(x/T)$ in Equations (6) should be changed to expressions in terms of lower boundaries $m_{LB}^{CR}(T)$ and $R_{LB}^{CR}(x/T)$ given a specified critical region CR, and therefore the expected total software testing cost in the worst case can be rewritten as:

$$\begin{aligned} &Min\ E_{LB}^{CR}\left[C(T)\right] = C_0 + C_1 T + C_2 m_{LB}^{CR}(T) u_y \\ &\qquad + C_3\left(1 - R_{LB}^{CR}(x/T)\right) + C_4\left(v_1 + T\right)^{v_2}, \quad (7) \\ &Subject\ to:\ R_{LB}^{CR}(x/T) \geq R_0 \end{aligned}$$

Decision makers can set an appropriate confidence level to determine the optimal release time with consideration of both software quality and costs during the testing phase.

VI. Application

In this section, a numerical example is provided to illustrate the determination of the optimal release policies. Suppose that a software technology company acquires a contract of developing a new work flow system. After the end of the coding phase, the manager of the application service provider has to identify an appropriate time to release the software.

According to the evaluation from historical data and domain experts, it is determined that Ohba's inflection S-shaped model would fit this case. The related parameters of the model are as follows: the potential errors *a* are about 4500 and the standard deviation of the detection rate σ is 0.3112. The parameters *b* and *c* are 1.5 and 0.005, respectively.

Besides, the staff in the testing department work a total of 10 hours a day, 30 days a month. Other related cost parameters are as follows: C_0 =\$2,000, C_1 =\$6,000, C_2 =\$12,000, C_3 =\$250000, C_4 =\$4,000, x =1 hours, v_1 =2, v_2 =1.6, and u_y =1 hour. In addition, the manager would like to assure that the developed business application can satisfy the minimal requirement of software reliability (R_0=0.9) with the 95% confidence level, and therefore they need to identify the optimal software release times for the average and worst cases.

By Equations (6) and (7) and spectrum analysis can be used to obtain the optimal release time, the expected testing cost, and expected reliability. As shown in Fig. 1, in the average case, the optimal release time *T** is 1.8 months after the testing/debugging work has started. The expected testing cost is about \$332505 and the expected reliability can reach 0.9381, which meets the minimal requirement of software reliability (R_0=0.9). However, in order to satisfy the software reliability at the confidence level of 95%, it would be necessary to extend the testing/debugging period and postpone the software release. In order to ensure that the software quality meets the requirement in the worst case situation, the decision-

maker should extend the time of testing and debugging time from 1.8 months to 2.15 months, and the expected testing cost is $342,595. The expected reliability reaches 0.92651.

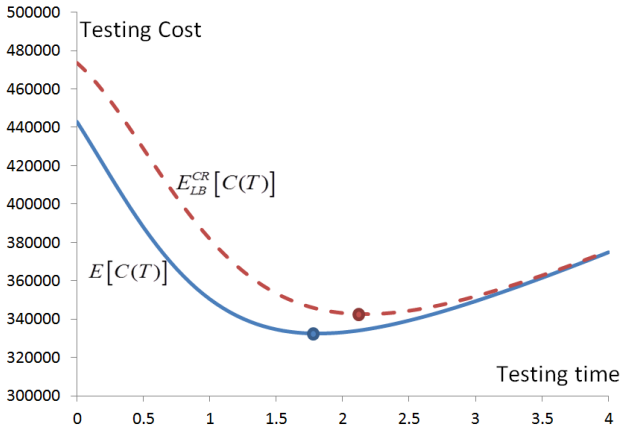

Figure 1. Expected testing costs versus testing time in the average case and the worst case

VII. Conclusion

In this paper, a software reliability model is presented using the SDE method to construct confidence intervals regarding the mean value function whose variance is assumed to stem from the error detection rate. It can assist software developers in determining optimal release times at different confidence levels. From the inference procedure,, the effects of time on the error detection rate and the variance in the mean value function would be crucial to the shapes and confidence intervals regarding the mean value function.

Future research may estimate the unknown parameters in the proposed model through expert opinions due to the lack of historical data, and this can be conducted by using a Bayesian decision approach coupled with the proposed model.

References

[1] H. Pham, *Software Reliability*, Springer, New York, 2000.
[2] P. K. Kapur, S. Anand, S. Yamada, and V. S. S. Yadavalli, "Stochastic differential equation based flexible software reliability growth model," *Mathematical Problems in Engineering*, 2009.
[3] S. Yamada, M. Ohba, and S. Osaki, "S-Shaped software reliability modeling for software error detection," *IEEE Transactions on Reliability*, vol. 32, pp. 475-484, 1983.
[4] S. Yamada, "Software quality/reliability measurement and assessment: software reliability growth models and data analysis," *Journal of Information Processing*, vol.14, no.3, pp. 254-266, 1991.
[5] M. Ohba, "Inflexion S-shaped software reliability growth models, Stochastic Models," *Reliability Theory*, Osaki, S. and Hatoyama, Y., Eds. Berlin, Germany: Springer-Verlag, pp. 144-162, 1984.
[6] M. Ohba, "Software reliability analysis models," *IBM Journal of Research and Development*, vol.28, pp. 428-443, 1984.
[7] H. Pham and X. Zhang, "NHPP software reliability and cost models with testing coverage," *European Journal of Operational Research*, vol. 145, no. 2, pp. 443-454, 2003.
[8] C. Y. Huang, "Performance analysis of software reliability growth models with testing-effort and change-point," *Journal of Systems and Software*, vol. 76: pp. 181-194, 2005.
[9] Y. Tamura and S. Yamada, "A flexible stochastic differential equation model in distributed development environment," *European Journal of Operational Research*, vol. 168: pp. 143-152, 2006.
[10] M. Yamada, H. Kimura, and S. Osaki "Software reliability measurement and assessment with stochastic differential equations," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E77-A, pp. 109-116, 1994.
[11] C. H. Lee, Y. T. Kim, and D. H. Park, "S-shaped software reliability growth models derived from stochastic differential equations," *IIE Transactions*, vol. 36, pp. 1193-1199, 2004.
[12] I. Karatsas and S. Shreve, *Brownian Motion and Stochastic Calculus*, 2nd ed., New York: Springer-Verlag, 1997.
[13] S. Yamada and S. Osaki, "Software reliability growth modeling: Models and applications," *IEEE Transactions on Software Engineering*, vol. 11, pp. 1431-1437, 1985.
[14] L. Yin and K. S. Trivedi, "Confidence interval estimation of nhpp-based software reliability models," in *Proceedings 10th International Symposium on Software Reliability Engineering*, 1999, pp. 6-11.
[15] A. L. Goel and K. Okumoto, "Time-dependent fault detection rate model for software and other performance measures," *IEEE Transactions on Reliability*, vol. 28, pp. 206-211, 1979.
[16] J. D. Musa, "Software engineering: The future of a profession," *IEEE Software*, vol. 2, no. 1, pp. 55-62, 1985.
[17] .K. C. Chiu, Y. S. Huang, and T. Z. Lee, "A study of software reliability growth from the perspective of learning effects," *Reliability Engineering and Systems Safety*, vol. 93, no. 10, pp. 1410-1421, 2008.

Tean-Quay Lee is currently a lecturer in the Department of Information Management at Shu-Te University, Taiwan, R.O.C. He earned his M.S. degrees from the Department of Computer Science at Stevens Institute of Technology, U.S. His research interests include decision analysis, database, and soft computing.

Chih-Chiang Fang is currently an assistant professor in the Department of Information Management at Shu-Te University, Taiwan, R.O.C. He earned both his M.S. and Ph.D. degrees from the Department of Industrial and Information Management at National Cheng Kung University. His research interests include decision analysis, data mining, and software engineering. Related papers have appeared in such professional journals as Naval Research Logistics, IEEE Transactions on Engineering Management, Software Testing, Verification and Reliability, Computers & Industrial Engineering, International Journal of Production Economics, International Journal of Production Research, Decision Support Systems and others.

Chun-Wu Yeh is currently an assistant professor in the Department of Information Management, Kun Shan University, Taiwan. He earned his Ph.D. degree from the Department of Industrial and Information Management at National Cheng Kung University. His research interests include operations management, data mining, machine learning, and time series data analysis. Related papers have appeared in such professional journals as Computers & Industrial Engineering, Engineering Applications of Artificial Intelligence, Expert Systems, Expert Systems with Applications and others.