A risk-reduction approach for optimal software release time determination with the delay incurred cost

Rui Peng, Yan-Fu Li, Jun-Guang Zhang, Xiang Li

A risk-reduction approach for optimal software release time determination with the delay incurred cost

Rui Peng[1], Yan-Fu Li[2*], Jun-Guang Zhang[1], Xiang Li[3]

[1]Dongling School of Economics and Management, University of Science & Technology Beijing, Beijing, China

[2]Ecole Centrale Paris - SUPELEC, Paris, France

(yanfu.li@ecp.fr, yanfu.li@supelec.fr)

[3]Industrial & Systems Engineering, National University of Singapore, Singapore

Abstract

Most existing research on software release time determination assumes that parameters of the software reliability model (SRM) are deterministic and the reliability estimate is accurate. In practice, however, there exists a risk that the reliability requirement cannot be guaranteed due to the parameter uncertainties in the SRM, and such risk can be as high as 50% when the mean value is used. It is necessary for the software project managers to reduce the risk to a lower level by delaying the software release, which inevitably increases the software testing costs. In order to incorporate the managers' preferences over these two factors, a decision model based on multi-attribute utility theory (MAUT) is developed for the determination of optimal risk-reduction release time.

Keywords: software reliability, software release time, parameter uncertainty, multi-attribute utility theory (MAUT)

1. Introduction

Software plays key roles in many industrial systems (e.g. power grid, telecommunication network, internet, etc) of modern society. Software reliability is of great importance for the stable operation of such systems. To ensure the reliability, software needs to be systematically tested prior to its release to the market. During the testing phase, the latent software faults are identified, isolated and removed. As a result, software reliability is improved. Based on the time-to-failure data obtained from the testing phase, software reliability can be measured and predicted using appropriate software reliability models (SRMs) (Musa et al., 1987, Xie, 1991, Lyu, 1996).

Besides measuring and predicting software reliability, the SRMs are often used to support the software project managers making important decisions. A typical application is to advice the managers when to release the software. Consequently, the optimal release time determination during the software testing phase has become an extensively researched topic (Okumoto and Goel, 1980, Yamada et al., 1984, Yamada and Osaki, 1985, Pham, 1996, Pham and Zhang, 1999, Xie and Yang, 2003, Huang and Lyu, 2005, Boland and Ní Chuív, 2007, Liu and Chang, 2007, Ho et al., 2008, Yang et al., 2008, Lai et al., 2011, Li et al., 2011). Okumoto and Goel (1980) originally formulated and studied the optimal software release time determination problem. Yamada et al. (1984) studied the optimum release policies minimizing the total expected software cost with a scheduled software delivery time. Yamada and Osaki (1985) developed a decision-making model, where both reliability and cost are considered. Pham and Zhang (1999) developed a software reliability-cost model to determine the optimal release policies that maximize the expected net gain in reliability. Pham (1996), Xie and Yang (2003), and Boland and Ní Chuív (2007) investigated the effect of imperfect debugging on release time determination. Huang and Lyu (2005) highlighted the importance of testing effort and testing efficiency in this decision problem. Ho et al. (2008) emphasized the learning effects for release time determination. Liu and Chang (2007) proposed a non-Gaussian Kalman filter model for a reliability-constrained software release policy. More recently, Yang et al. (2008) developed a new optimal software release time model, which can control the risk of the project being over-budgeting. Li et al. (2011) proposed a model for

the optimal version-updating time determination of open source software using multi-attribute utility theory (MAUT) to combine two decision attributes: development time and reliability. Similar as (Yamada and Osaki, 1985), Lai et al. (Lai et al., 2011) studied the software release time policy considering both reliability and software cost.

In the release time determination problem, meeting the reliability requirement is of great importance. This is because the customers generally have a minimum reliability requirement, which can be specified in the contract. In order to check whether the reliability requirement is satisfied, SRM is often adopted to predict the reliability of software after its release. Most existing research on release time determination assumes that the parameters of the software reliability model are deterministic and the reliability estimate is accurate (Yamada and Osaki, 1985, Xie and Yang, 2003, Huang and Lyu, 2005, Boland and Chuĭv, 2007, Liu and Chang, 2007, Ho et al., 2008, Yang et al., 2008, Li et al., 2011). In practice, however, there exists a risk that the reliability requirement cannot be guaranteed due to the uncertainties in the software testing process which are reflected in the parameters of SRMs, and such risk can be as high as 50% if the software is released at the moment the estimated reliability equals to the reliability requirement (shown in Section 2). It is necessary for managers to reduce the risk to a lower level, and thus the testing process is expected to be longer, which inevitably increases the costs of testing. In order to balance between reducing the risk of unfulfilling the reliability requirement and controlling the cost incurred by release delay, this paper develops a new decision model for software release time determination, using MAUT (Fishburn, 1970) to optimize the two conflicting objectives simultaneously.

The rest of the paper is organized as follows. In Section 2, the limitations of the existing research on software release time determination are further discussed, which motivate us to develop a new decision model. In addition, the attributes including risk and delay incurred cost are formulated. In Section 3, the decision model based on MAUT is presented, and the procedures of constructing it are described. In Section 4, the proposed decision model is illustrated by a case study. In Section 5, threats to validity are discussed. Finally, concluding remarks are presented in Section 6.

2. Model Formulation

Considering the minimum reliability requirement level R_0, the decision problem is typically formulated as

$$R(x \mid t) \geq R_0, \tag{1}$$

where $R(x \mid t)$ is the conditional software reliability, defined as the probability that the software will operate without failure within time interval $(t, t+x]$ given that it is released at t. The optimal release time T is then the minimum testing time required so that the software reliability reaches the level R_0. In most software reliability models, there are a set of parameters $\boldsymbol{\theta} = (\theta_1, \theta_2, ..., \theta_m)$ (where m is the number of parameters) used to represent the optimal release time T by a function $T = f(\theta_1, \theta_2, ..., \theta_m, R_0)$. To solve T, most existing research works assume that these model parameters are known without uncertainty, and $R(x \mid t)$ can model the actual software reliability exactly (Yamada and Osaki, 1985, Xie and Yang, 2003, Huang and Lyu, 2005, Boland and Chuĭv, 2007, Liu and Chang, 2007, Ho et al., 2008, Yang et al., 2008, Li et al., 2011).

2.1 Risk considerations

In reality, the exact values for these model parameters are often unknown. Instead, they are estimated from the collected time-to-failure data. Parameter uncertainty arises since the estimated parameters are subject to the random variations (or noises) in the data (Dai et al., 2007). Due to the uncertainty of parameters, the software reliability computed from SRM is no longer deterministic. Consequently, the optimal release time T given a reliability target is also a random variable.

When the SRM parameters are estimated by the maximum likelihood estimation (MLE) method (Nelson, 1982), it is shown that the optimal release time T given R_0 is asymptotically normally distributed with mean $\hat{T}$ and variance $Var(\hat{T})$. Here, $\hat{T}$ is obtained from solving (1) with the estimated parameters, and $Var(\hat{T})$ is the variance of $\hat{T}$. Details about these quantities are presented in the Appendix 1.

Considering the uncertainty in T, the risk that software cannot meet the reliability requirement when it is released at time t can be quantified as

$$r_0(t) = P(R(x \mid t) < R_0) = P(t < T) = 1 - \Phi\left(\frac{t - \hat{T}}{\sqrt{Var(\hat{T})}}\right), \tag{2}$$

where $\Phi(x)$ is the cumulative probability function (CDF) of standard normal distribution. It is seen that when the mean value of release time, $\hat{T}$ is used, there is $1 - \Phi(0) = 50\%$ chance that the reliability requirement cannot be guaranteed. Such risk is too high to be acceptable. As a result, reducing the risk to a lower level to improve the confidence on the software reliability becomes an important issue. To account for this, the risk-reduction release time T_R is introduced as,

$$T_R = \hat{T} + z_{r_0}\sqrt{Var(\hat{T})}, \tag{3}$$

where r_0 denotes the acceptable risk level of managers, and z_{r_0} is the (1-r_0) quantile of the standard normal distribution. As seen from (3), the release time based on risk reduction requires a delay of $z_{r_0}\sqrt{Var(\hat{T})}$, which often results in the increase of the testing costs. This is a useful approach if the managers are certain about the risk level required and are committed to achieve it at all costs. On the other hand, it is also easy to elicit a maximum tolerable risk value given the project budget (Nan and Harter, 2009).

2.2 Cost Considerations

From the managers' perspective, it is also important to control the cost incurred by release delay (Pham and Zhang, 1999). Let the testing cost function be denoted by $C(t)$, the delay incurred cost at time t ($t > \hat{T}$) is obtained as,

$$C_p(t) = C(t) - C(\hat{T}) \tag{4}$$

The discussions above indicate that reducing the risk and controlling the delay incurred cost are two important but conflicting criteria that should be considered simultaneously when determining the software release time. Therefore, it is natural to incorporate the managers' preference into the decision process to make a compromise between these two

criteria. In Section 3, the MAUT is adopted, and a decision model is developed for the determination of optimal release time.

3. The decision model based on MAUT

The application of MAUT is based on a one-dimensional multi-attribute utility function, which is the measure of the attractiveness of the conjoint outcome of the different attributes. The additive form of the multi-attribute utility function is given by

$$U(d_1, d_2, .. d_n) = \sum_{i=1}^{n} w_i u(d_i), \quad (5)$$

where each attribute is denoted by d_i, i=1,2,…n, the attractiveness of each attribute is represented by the single utility function $u(d_i)$ and w_i is the scaling constant which represents the importance weight for the utility $u(d_i)$. The sum of the weights is equal to 1 (von Winterfeldt and Edwards, 1986). By maximizing the multi-attribute utility function, the best alternative (i.e. the best set of values of the decision variables) is obtained, which gives the maximum attractiveness of the conjoint outcome of the attributes.

The main reason for using MAUT in our approach is that the typical management scenarios can be appropriately represented within its structure. In the decision problem formulated, there are two competing objectives to be balanced: minimizing the risk and minimizing the delay incurred cost. Given that the risk reduction and the cost control are both subjective, the single utility function is used to reveal managers' preference towards each attribute. By allocating different values of importance weights to the utilities of the attributes, managers can use the multi-attribute utility function to measure the total attractiveness of the conjoint outcome of the risk and the delay incurred cost given a specified release time.

Another reason for the selection of MAUT is that it has strong theoretical foundations due to the use of the expected utility theory. The utility theory takes managers' risk attitude into account, e.g. risk neutrality, risk aversion and risk proneness (Fishburn, 1970). Furthermore, MAUT provides a feasible approach for considering the continuous scale of the alternatives. Specifically, in our problem, the release time as the alternative

should be considered in a continuous scale. Last but not least, when managers have other requirements, i.e., the minimization of the total cost in the software development cycle (Sgarbossa and Pham, 2010), the control of the uncertainty in the total cost function (Yang et al., 2008), the optimized resource allocation (Ngo-The and Ruhe, 2009), our decision model can be extended by introducing more attributes in the framework of MAUT. The proposed MAUT procedure for our decision problem is discussed in detail below.

3.1 Elicitation of single utility function for each attribute

After the quantification of each attribute by (2) and (4), managers' preference towards the performance of each attribute should be assessed. To represent this, the single utility function is used. Suppose that the highest and lowest expected risks are first selected as r_0^0 and r_0^1, respectively. In real applications, they provide the lowest and highest satisfactions to the managers, respectively. Cukic et al. (Cukic et al., 2003) suggests that the reliability and the confidence of the reliability are usually application specific and predefined. For example, suppose that managers can only accept a risk level below 5%. Then, $r_0^0 = 5\%$ and $r_0^1 = 0$. At these boundary conditions, we have $u\left(r_0^0\right) = 0$ and $u\left(r_0^1\right) = 1$. The superscript of r_0^i, $i \in [0,1]$ is used to represent the corresponding utility value, which is determined so that the management is indifferent between the following two alternatives: 1) getting risk r_0^i with certainty; 2) getting risk r_0^0 with probability (1-*i*) and r_0^1 with probability *i* (Keeney and Raiffa, 1976, von Winterfeldt and Edwards, 1986). The single utility function is generally described by the linear or exponential function shown as follows (Keeney and Raiffa, 1976):

$$u(r_0) = \alpha + \beta \cdot r_0 \text{ or } u(r_0) = \alpha + \beta \exp(\gamma \cdot r_0), \quad (6)$$

where α, β and γ are constants which ensure $u(r_0) \in [0,1]$.

To determine which form in (6) should be used, we can compare the certainty equivalent *u*(50%) and the expected value of the 50-50 lottery (*u*(0%)+*u*(100%))/2. Specifically, if they are equal to each other, the managers are risk neutral and the linear form should be used. In this case, the utility function can be written as

$$u(r_0) = 1 - \frac{r_0 - r_0^1}{r_0^0 - r_0^1}$$

Otherwise, the managers are not risk neutral and the exponential form should be adopted. In this case, the utility function can be written as

$$u(r_0) = \frac{\exp(\gamma \cdot r_0^0) - \exp(\gamma \cdot r_0)}{\exp(\gamma \cdot r_0^0) - \exp(\gamma \cdot r_0^1)}$$

where γ is the non-zero solution to $\exp(\gamma \cdot r_0^1) + \exp(\gamma \cdot r_0^0) - 2\exp(\gamma \cdot r_0^{0.5}) = 0$.

3.2 Estimation of scaling constants

The following step is the estimation of the scaling constants w_1 and w_2=1-w_1, which correspond to the important weights of $u(r_0)$ and $u(C_p)$. There are two common methods to assess the scaling constants: certainty scaling and probabilistic scaling (von Winterfeldt and Edwards, 1986). Given that only two attributes are considered in our problem, the probabilistic scaling technique is used.

In the probabilistic scaling approach, the managers are asked to compare their preference over the two choices: 1) a deterministic joint outcome (r_0^1, C_p^0) comprising of the lowest risk and the highest delay incurred cost; 2) the lottery consists of both attributes at their best levels (r_0^1, C_p^1) with probability p and both attributes at their worst levels (r_0^0, C_p^0) with probability 1-p. The managers are first asked to compare the deterministic outcome with the lottery having a 50-50 chance of occurrence. If the managers prefer the certain outcome, the probability p is gradually increased until they are indifferent with these two choices. On the other hand, if the managers prefer the lottery, we decrease p. When the indifference is achieved, p is equal to the scaling constant w_1 for the risk attribute (von Winterfeldt and Edwards, 1986).

3.3 Maximization of multi-attribute utility function

By maximizing the multi-attribute utility function

$$U(r_0(t), C_p(t)) = w_1 u(r_0(t)) + w_2 u(C_p(t)), \quad (7)$$

the optimal risk-reduction release time is obtained as $T_R^* = \arg\max_t \{U(r_0(t), C_p(t))\}$. It is worth noting that (7) is based on certain independence assumptions. Interested readers can refer to (Keeney and Raiffa, 1976) for detailed theoretical discussions. In real-world applications, these assumptions are commonly accepted (Brito and de Almeida, 2008, Ferreira et al., 2009). Moreover, it has been shown that even when these assumptions are violated, the additive multi-attribute utility function can provide fairly good approximations (Edwards, 1977, Farmer, 1987).

3.4 Summary of the procedure

The first step of the implementation of the decision model is to quantify the attributes, i.e., the risk and the delay incurred cost. For the risk attribute, based on the standard statistical results, risk can be quantified by (2). For the cost attribute, the generalized cost model is used and it is quantified by (4). The following step is the elicitation of single utility functions for both attributes. After this, the scaling constants for each attribute are determined following procedures in Section 3.2. Finally, based on the single utility functions and the scaling constants, the multi-attribute utility function is obtained as shown in (7). The optimal risk-reduction release time is determined by maximizing it.

4. Case Study

In this section, the proposed optimal release time determination approach is applied onto the case study used in (Pham and Zhang, 1999). By considering the risk and the delay incurred cost simultaneously, the optimal release time is determined by incorporating the managers' preference into the decision process. In addition, sensitivity analysis is introduced to assist checking the robustness of the final decision.

4.1 The determination of optimal risk-reduction release time

Step 1: quantification of attributes

The Goel-Okumoto (GO) model (Goel and Okumoto, 1979) was adopted in (Pham and Zhang, 1999) to analyze the failure data for reliability assessment. In this work, we use this model as well. It is noted that the procedures are similar if other software reliability

models are adopted. Moreover, future research can be carried out to analyze the impact of parameter uncertainties when different models are used for software release time determination. The mean value function and the failure intensity function of the GO model are given by

$$m(t) = a(1-e^{-bt}) \text{ and } \lambda(t) = abe^{-bt} \tag{8}$$

where a denotes the total number of expected faults in the software and b represents the fault detection rate. Furthermore, the reliability of the software system during its operational phase is obtained as

$$R(x \mid t) = \exp[-\lambda(t)x] \tag{9}$$

and $R(x \mid t)$ represents the conditional software reliability, which is defined as the probability that the software will not fail given a specified time interval $(t, t+x]$ in the operational phase (Yang and Xie, 2000). Since x is usually set to 1 without loss of generality, the release time given the reliability target R_0 is

$$T = \frac{1}{b}\ln\left[\frac{ab}{\ln(1/R_0)}\right] \tag{10}$$

Suppose that customer has indicated a reliability requirement of R_0=0.95. Based on the maximum likelihood estimates $\hat{a} = 139.862$ and $\hat{b} = 0.144$ (see Appendix 2 for details), the mean value of the release time is $\hat{T} = 41.479$. Moreover, from the standard statistical analysis (Nelson, 1982), as shown in the appendix, the variance of the release time is obtained $Var(\hat{T}) = 9.758$. Accordingly, the attribute risk can be quantified by substituting these estimated values into (2).

The cost model proposed by Pham and Zhang (1999) consists of two parts, i.e., the expected general testing cost $C_1(t)$ and the expected cost of removing errors during testing phase $C_2(t)$ as

$$C_1(t) = c_1 t^{\kappa},\ C_2(t) = c_2 m(t)\mu_y, \tag{11}$$

where c_1 is the software test cost per unit time, κ is the discount rate of the testing cost due to the learning effect, c_2 is the cost of removing an error per unit time during the

testing phase and μ_y is the expected time of removing an error during this period. According to Pham and Zhang (1999), the coefficients in the cost model can be determined by empirical data and previous experiences of the staff members. We set the parameters as: c_1=700, $\kappa = 0.95$, c_2=60 and $\mu_y = 0.1$, same to the assignments in (Pham and Zhang, 1999). It is worth noting that risk is expected to be less than 50% from managers' point of view. Therefore the software should be released after $\hat{T} = 41.479$ in order to reduce the risk, that is, we just need to consider $t \in [\hat{T}, +\infty)$. Accordingly, the delay incurred cost at the time t is obtained as

$$C_p(t) = c_1\left[(t)^{\kappa} - \left(\hat{T}\right)^{\kappa}\right] + c_2\left[m(t) - m\left(\hat{T}\right)\right]\mu_y . \tag{12}$$

Step 2: Elicitation of single utility functions

The following step is to assess managers' preference towards the performance of the risk and the delay incurred cost. Interviews with the managers are needed to elicit reasonable single utility functions. Suppose that management scenarios are as follows:

(1) Managers are risk neutral towards both attributes.

(2) Managers indicate that they can only accept up to a risk level of 5%, and the smaller the risk the better, until the risk can be eliminated.

(3) Managers have an incurred cost budget of $15000 and they are completely unsatisfied when all the money is spent; their satisfaction increases when the expense decreases, and the highest satisfaction level is achieved when no money is spent.

According to the management scenarios above, corresponding explanations on the determination of single utility functions are shown as follows:

(1) Since managers are risk neutral towards both attributes, the linear form of the single utility function is used.

(2) The lowest risk requirement is $r_0^0 = 5\%$ and the highest risk expectation is $r_0^1 = 0$. The single utility function for risk is obtained as $u(r_0) = 1 - 20r_0$.

(3) The maximum cost budget is $C_p^0 = 15000$ and the highest satisfactory cost expectation is $C_p^1 = 0$. The single utility function for delay incurred cost is determined as $u(C_p) = 1 - C_p / 15000$.

Step 3: Estimation of scaling constants

At this stage, the scaling constant w_1 is estimated first by comparing the deterministic joint outcome (r_0^1, C_p^0) with the lottery consists of (r_0^1, C_p^1) with probability p and (r_0^0, C_p^0) with probability 1-p. Suppose managers claim that they are indifferent between these two choices when p is equal to 0.5, then w_1=0.5. Since the sum of scaling constants is equal to one, w_2 is equal to 0.5 as well.

Step 4: Maximization of multi-attribute utility function

Based on the estimated single utility functions and scaling constants, the multi-attribute utility function can be obtained by (7). Figure 1 shows this multi-attribute utility function as a function of the release time. This multi-attribute utility function is maximized when $T_R^* = 50.586$ and the corresponding risk and delay incurred cost at this time are $r_0(T_R^*) = 0.18\%$ and $C_p(T_R^*) = 50029$ respectively. As a result, software should be released at the optimal risk-reduction release time $T_R^* = 50.586$ to appropriately compromise between reducing the risk and controlling the delay incurred cost.

Figure 1 Multi-attribute utility function given different release times

4.2 Illustration of the proposed decision model

In Figure 3, we denote $\hat{T} = 41.479$ as the mean value of release time without consideration of parameter uncertainty. If we release the software at this time, no cost is incurred and $C_p^1 = 0$. However, at this release time, the 50% risk is too high to be acceptable by the managers because the lowest risk requirement $r_0^0 = 5\%$ is not satisfied.

At this point, the manager has to make a compromise between reducing the risk and controlling the delay incurred cost.

With this consideration, the software testing time is expected to increase. We denote $T(r_0^0)=46.618$ and $T(C_p^0)=69.026$ as the release times when the lowest risk requirement $r_0^0=5\%$ and the maximum cost budget $C_p^0=15000$ are satisfied respectively. When *t* increases between these two time points, the single utility function associated with risk increases and the single utility function associated with cost decreases. As the weighted sum of the two single utility functions, the multi-attribute utility function increases first and then decreases. The optimal risk-reduction release time which maximizes the multi-attribute utility function is $T_R^*=50.586$, and the corresponding risk and delay incurred cost are $r_0(T_R^*)=0.18\%$ and $C_p(T_R^*)=50029$ respectively.

Finally, it should be noted that during the time periods $[\hat{T}, T(r_0^0)]$ and $[T(C_p^0), +\infty)$, the multi-attribute utility function is dominated by only one of the attributes. More specifically, for the first period, since the lowest risk requirement $r_0^0=5\%$ has not been satisfied, the delay incurred cost is the only attribute contributing to the multi-attribute utility function. Given that the delay incurred cost is increasing over time and managers' satisfaction level is decreasing with it, the multi-attribute utility function is decreasing during this time period. While for the second time period, the multi-attribute utility function is dominated by the risk attribute and it equals to $0.5u(r_0)$. Figure 1 shows that the multi-attribute utility function remains at 0.5 level when release time is greater than $T(C_p^0)$. It implies that the available cost budget $C_p^0=15000$ is sufficient for the managers to reduce the risk to the best level $r_0^1=0$.

4.3 Sensitivity analysis

As shown in the Sections above, the optimal risk-reduction release time can be determined by maximizing the multi-attribute utility function. However, since most parameters in the MAUT are obtained based on the subjective assessments of the managers, the optimal risk-reduction release time obtained may not be accurate. In practice, the managers have to know how robust the optimal decision is, and thus sensitivity analysis is needed. More specifically, sensitivity analysis can help to investigate the relative variation of the optimal solution when a specific parameter changes, i.e., the change of cost parameters, scaling constants, etc. The results from sensitivity analysis reveal the stability of the optimal solution.

Sensitivity analysis is generally done by changing one parameter and setting the other parameters at the fixed levels (Xie and Hong, 1998, Li et al., 2010). The sensitivity of the optimal decision to one parameter x can be quantified by $S_{q,x}$, defined as the relative change of the optimal risk-reduction release time when x is changed by $100q\%$ (Xie and Hong, 1998, Li et al., 2010).

$$S_{q,x} = \frac{T_R^*(x+qx) - T_R^*(x)}{T_R^*(x)} \tag{13}$$

A large value of $S_{q,x}$ indicates that parameter x has significant impact onto the determination of T_R^*, and T_R^* is regarded as sensitive to the change of x. Normally, managers should pay special attention to the important parameters as the optimal decision T_R^* is heavily dependent on the accurate estimates of them (Xie and Hong, 1998, Li et al., 2010).

From a practical point of view, it may not be necessary to conduct sensitivity analysis for all the parameters in this optimal release time problem. For instance, parameters c_2 and μ_y are expected to be insignificant, because the expected cost to remove errors from time t to $\hat{T}$ is negligible. More specifically, given a high reliability requirement such as R_0=0.95, there will be few faults detected from $\hat{T}$ to t. Additionally, as c_1=700, c_2=60 and $\mu_y = 0.1$; compared with the estimated value of c_1, the product of c_2 and μ_y is too

small to have any impact on the delay incurred cost function in (12). Another example is the determination of r_0^1 and C_p^1, which represents the highest risk reduction expectation and highest cost control expectation, respectively. Since managers always prefer less risk and less cost, setting them to zero can properly describe the best cases for risk reduction and cost control respectively.

In contrast, parameters c_1 and κ are much more important since they dominate the change of the delay incurred cost over time. Similarly, r_0^0 and C_p^0 are of importance as shown in Figure 1, where $T\left(r_0^0\right)$ and $T\left(C_p^0\right)$ are the changing points of multi-attribute utility function. Furthermore, scaling constants w_1 and w_2 are also important since they represent the different importance weights allocated to both attributes, which directly affect the final solution of T_R^*. Since the sum of these two weights is equal to one, investigating one factor is sufficient. Results of sensitivity analysis with regard to these parameters are summarized in Table 1. Specially, since parameter κ represents the learning effect of the testing team which is not greater than 1, the value of $S_{q,\kappa}$ when $\kappa = 1$ is used for the positive change of κ.

Table 1 Sensitivity analysis results given different parameters

q	-30%	-20%	-10%	10%	20%	30%
S_{q,w_1}	-1.36%	-0.88%	-0.43%	0.42%	0.84%	1.27%
S_{q,C_p^0}	-0.77%	-0.48%	-0.22%	0.20%	0.38%	0.55%
S_{q,r_0^0}	0.74%	0.47%	0.22%	-0.20%	-0.39%	-0.57%
S_{q,c_1}	0.74%	0.47%	0.22%	-0.20%	-0.39%	-0.57%
$S_{q,\kappa}$	2.91%	1.96%	0.99%		-0.53%	

It can be seen that these parameters do not significantly influence the final solution on T_R^* since all the absolute values of $S_{q,x}$ are below 3%. In other words, the optimal risk-reduction release time obtained is robust to the changes in the parameters. Moreover,

results in Table 1 indicate that T_R^* is positively correlated with w_1 and C_p^0, and negatively correlated with r_0^0, c_1 and κ. Physical meanings of these parameters can actually explain these results. For instance, when w_1 increases, it means that more importance is allocated for the control of risk. As a result, T_R^* increases as well.

5. Threats to validity

Based on the standard statistical analysis (Nelson, 1982), there is 50% chance that the software will not meet its reliability requirement when the mean value of the release time, $\hat{T}$ is used. However, it should be noted that the standard statistical analysis is an approximation. It is still an open question whether the risk is really as high as 50%. To investigate this problem, an empirical case study is conducted by the Monte Carlo simulation using MATLAB tool.

In particular, the GO model is adopted, where the preset parameters are given by $a = 100$ and $b = 0.1$. Suppose that the reliability requirement is $R_0 = 0.95$, then the real value of optimal release time can be obtained as $T_{real} = 52.73$. According to the general procedures discussed in (Lyu, 1996), 10000 failure data sets are generated, and each failure data set is composed of ninety time-to-failure data points. Since each failure data set can produce an estimate of the optimal release time denoted by $\hat{T}$, risk that software cannot meet the reliability requirement can be easily estimated by comparing these $\hat{T}$ values with T_{real}, and such risk is estimated as $\hat{r}_0 = 60.21\%$. Although this result is different from the estimated risk based on the standard statistical analysis, it is another piece of evidence that the risk due to parameter uncertainty cannot be neglected, because the risk can be even higher than 50%.

In addition, the normal distribution is used to quantify the uncertainty of optimal release time. Although this type of approximation technique is widely applied to reliability engineering, it may not be accurate. In this case, incorporating experts' opinion and past experience could be a choice. For example, experts could probably know the distributions

of some model parameters based on their past experience on similar software projects. Based on this type of information, parameter uncertainty can be effectively quantified by combining the Maximum-Entropy Principle (MEP) into the Bayesian approach (Dai et al., 2007). By incorporating this quantification of parameter uncertainty into the simulation of optimal release time, the uncertainty of optimal release time can be modeled more sufficiently.

Besides the consideration of risk, the delay incurred cost is incorporated into our decision problem. This is because the risk cannot be overlooked due to the limit cost budget of the project (Nan and Harter, 2009). Management needs to strike a balance between reducing the risk and controlling the delay incurred cost. In other words, given a reliability requirement, we introduce two new important dimensions for the determination of optimal release time: the risk that software cannot meet the reliability requirement due to parameter uncertainty, and the delay incurred cost associated with such risk. However, it should be noted that the formulation here may not be sufficient for release time determination. In reality, managers can also have other requirements, which may include the minimization of the total cost in the software development cycle (Sgarbossa and Pham, 2010), the control of the uncertainty in the total cost function (Yang et al., 2008), and the optimized resource allocation (Ngo-The and Ruhe, 2009), etc. When these requirements are considered, our decision model can be extended by introducing more attributes into the framework of MAUT.

6. Conclusions

The software release problem during the testing phase is of great importance in the software development cycle. This paper discusses in detail when to release software given a reliability constraint. In particular, we highlight the risk in the reliability estimate due to the parameter uncertainty in the SRM. However, reducing such risk inevitably increases the testing costs. Thus, from the management's point of view, a compromise should be made between reducing the risk and controlling the delay incurred cost associated with it. To account this issue, a decision model based on MAUT is developed for the determination of optimal risk-reduction release time. The proposed model

provides project managers with a boarder view of the release time determination problem. It not only allows managers to optimize two criteria simultaneously, but also incorporates managers' preference to the decision process. In this paper, the risk of not fulfilling the software reliability requirement is studied from the aspect of parameter uncertainties in the SRM. Future work can be conducted to analyze the effect of choosing different SRMs on the estimated optimal release time and the risk of not fulfilling the reliability requirement.

Acknowledgment

The research is partially supported by the China NSFC under grant number 71231001，China Postdoctoral Science Foundation funded project under grant number 2013M530531, and the Fundamental Research Funds for the Central Universities under Grant FRF-MP-13-009A and FRF-TP-13-026A.

Appendix 1

Parameters in the software reliability model are estimated on the basis of the recorded time-to-failure data. Maximum likelihood estimation (MLE) technique is generally adopted for such estimation. Based on the standard statistical analysis (Nelson, 1982), the optimal release time T given a reliability target is asymptotically normally distributed with mean $\hat{T}$ and variance $Var\left(\hat{T}\right)$.

Suppose that there are totally m model parameters to be estimated, $\theta_1, \theta_2, \cdots, \theta_m$. Let n_i denote the number of failures observed within each time interval $[t_{i-1}, t_i)$, where $0 = t_0 < t_1 < \cdots < t_k = t$ and t is the time at which the testing process has expired. The likelihood function for a non-homogeneous Poisson process (NHPP) model with mean value function $m(t)$ is

$$L = \prod_{i=1}^{k} \frac{\left[m(t_i) - m(t_{i-1})\right]^{n_i} \exp\left\{-\left[m(t_i) - m(t_{i-1})\right]\right\}}{n_i!} \tag{14}$$

By maximizing the likelihood function above, point estimates of model parameters can be determined. The variances of these estimators can be calculated following the

asymptotic theory for maximum likelihood estimation (Nelson, 1982). In particular, the Fisher information matrix is obtained as

$$I(\theta_1,\cdots,\theta_m)=\begin{bmatrix} -E\left[\frac{\partial^2 \ln L}{\partial\theta_1^2}\right] & -E\left[\frac{\partial^2 \ln L}{\partial\theta_2\partial\theta_1}\right] & \cdots & -E\left[\frac{\partial^2 \ln L}{\partial\theta_m\partial\theta_1}\right] \\ -E\left[\frac{\partial^2 \ln L}{\partial\theta_1\partial\theta_2}\right] & -E\left[\frac{\partial^2 \ln L}{\partial\theta_2^2}\right] & \cdots & -E\left[\frac{\partial^2 \ln L}{\partial\theta_m\partial\theta_2}\right] \\ \cdots & \cdots & \cdots & \cdots \\ -E\left[\frac{\partial^2 \ln L}{\partial\theta_1\partial\theta_m}\right] & -E\left[\frac{\partial^2 \ln L}{\partial\theta_2\partial\theta_m}\right] & \cdots & -E\left[\frac{\partial^2 \ln L}{\partial\theta_m^2}\right] \end{bmatrix} \quad (15)$$

According to the standard theory of MLE, when the data size is large, $[\theta_1,\theta_2\cdots,\theta_m]$ converges to *m*-variate normal distribution with mean $[\hat{\theta}_1,\hat{\theta}_2\cdots,\hat{\theta}_m]$ and variance $[Var(\hat{\theta}_1),Var(\hat{\theta}_2),\cdots,Var(\hat{\theta}_m)]$. The asymptotic covariance matrix, which is the inverse of the Fisher information matrix, is given by

$$V=I^{-1}=\begin{bmatrix} Var(\hat{\theta}_1) & Cov(\hat{\theta}_1,\hat{\theta}_2) & \cdots & Cov(\hat{\theta}_1,\hat{\theta}_m) \\ Cov(\hat{\theta}_2,\hat{\theta}_1) & Var(\hat{\theta}_2) & \cdots & Cov(\hat{\theta}_2,\hat{\theta}_m) \\ \cdots & \cdots & \cdots & \cdots \\ Cov(\hat{\theta}_m,\hat{\theta}_1) & Cov(\hat{\theta}_m,\hat{\theta}_2) & \cdots & Var(\hat{\theta}_m) \end{bmatrix} \quad (16)$$

Subsequently, based on the covariance matrix, the uncertainty of other quantities, which are functions of parameters $[\theta_1,\theta_2\cdots,\theta_m]$, can be quantified as well. In our release time determination problem, we denote $T=f(\theta_1,\theta_2,...,\theta_m)$ as the optimal release time given the reliability target. The variance of it is estimated by

$$Var(\hat{T})=\sum_{i=1}^{m}\left(\frac{\partial T}{\partial\theta_i}\right)^2 Var(\hat{\theta}_i)+\sum_{i=1}^{m}\sum_{\substack{j=1\\ i\neq j}}^{m}\left(\frac{\partial T}{\partial\theta_i}\right)\left(\frac{\partial T}{\partial\theta_j}\right)Cov(\hat{\theta}_i,\hat{\theta}_j) \quad (17)$$

where $\partial T/\partial\theta_i$ is evaluated at $[\hat{\theta}_1,\hat{\theta}_2\cdots,\hat{\theta}_m]$. Based on the standard statistical analysis (Nelson, 1982), T is asymptotically normally distributed with mean $\hat{T}$ and variance $Var(\hat{T})$.

Appendix 2

Table 2 shows the software testing data used in (Pham and Zhang, 1999), which are summarized as the number of failures per one-hour interval of execution time.

Table 2 Failure in 1 hour intervals and cumulative failures

Hour (i)	Number of failures (n_i)	Cumulative failures
1	27	27
2	16	43
3	11	54
4	10	64
5	11	75
6	7	82
7	2	84
8	5	89
9	3	92
10	1	93
11	4	97
12	7	104
13	2	106
14	5	111
15	5	116
16	6	122
17	0	122
18	5	127
19	1	128
20	1	129
21	2	131
22	1	132
23	2	134
24	1	135
25	1	136

According to (14), the likelihood function can be written as

$$L = \prod_{i=1}^{k} \frac{[m(t_i) - m(t_{i-1})]^{n_i} \exp\{-[m(t_i) - m(t_{i-1})]\}}{n_i!}$$
$$= \frac{[m(1) - m(0)]^{27} [m(2) - m(1)]^{16} \cdots [m(25) - m(24)]^{1} \exp\{-[m(25) - m(0)]\}}{27!16! \cdots 1!} \quad (18)$$

According to (8), we have

$$\ln(L)$$
$$= 27\ln(m(1) - m(0)) + 16\ln(m(2) - m(1)) + \ldots + 1\ln(m(25) - m(24)) - m(25) - \ln(27!16! \cdots 1!)$$
$$= 136\ln(a) + 27\ln(1 - e^{-b}) + 16\ln(e^{-b} - e^{-2b}) + \ldots + 1\ln(e^{-24b} - e^{-25b}) - a(1 - e^{-25b}) - \ln(27!16! \cdots 1!)$$

Furthermore, we have

$$\frac{\partial \ln(L)}{\partial a} = \frac{136}{a} - 1 + e^{-25b}$$

$$\frac{\partial \ln(L)}{\partial b} = \frac{27e^{-b}}{1 - e^{-b}} + \frac{16(2e^{-2b} - e^{-b})}{e^{-b} - e^{-2b}} \ldots + \frac{25e^{-25b} - 24e^{-24b}}{e^{-24b} - e^{-25b}} - 25abe^{-25b}$$

Solving $\frac{\partial \ln(L)}{\partial a} = \frac{\partial \ln(L)}{\partial b} = 0$ gives $\hat{a} = 139.862$ and $\hat{b} = 0.144.$

References

Boland, P. J. & Chuĭv, N. N. (2007). Optimal times for software release when repair is imperfect. *Statistics & Probability Letters,* 77**,** 1176-1184.

Boland, P. J. & Ní Chuív, N. (2007). Optimal times for software release when repair is imperfect. *Statistics & Probability Letters,* 77**,** 1176-1184.

Brito, A. J. & De Almeida, A. T. (2008). Multi-attribute risk assessment for risk ranking of natural gas pipelines. *Reliability Engineering & System Safety,* 94**,** 187-198.

Cukic, B., Gunel, E., Singh, H. & Lan, G. (2003). The theory of software reliability corroboration. *IEICE TRANSACTIONS on Information and Systems,* 86**,** 2121-2129.

Dai, Y. S., Xie, M., Long, Q. & Ng, S. H. (2007). Uncertainty analysis in software reliability modeling by Bayesian approach with maximum-entropy principle. *IEEE Transactions on Software Engineering,* 33**,** 781-795.

Edwards, W. (1977). Use of multiattribute utility measurement for social decision making. *In:* BELL, D. E., KEENEY, R. L. & RAIFFA, H. (eds.) *Conflicting Objectives in Decision.* New York: Willey.

Farmer, P. C. (1987). Testing the robustness of multiattribute utility theory in an applied setting. *Decision Sciences,* 18**,** 178-193.

Ferreira, R. J. P., De Almeida, A. T. & Cavalcante, C. a. V. (2009). A multi-criteria decision model to determine inspection intervals of condition monitoring based on delay time analysis. *Reliability Engineering & System Safety,* 94**,** 905-912.

Fishburn, P. C. (1970). *Utility Theory for Decision Making,* New York, Wiley.

Goel, A. L. & Okumoto, K. (1979). Time-dependent error-detection rate model for software reliability and other performance measures. *IEEE Transactions on Reliability,* R28**,** 206-211.

Ho, J. W., Fang, C. C. & Huang, Y. S. (2008). The determination of optimal software release times at different confidence levels with consideration of learning effects. *Software Testing, Verification and Reliability,* 18**,** 221-249.

Huang, C. Y. & Lyu, M. R. (2005). Optimal release time for software systems considering cost, testing-effort, and test efficiency. *IEEE Transactions on Reliability,* 54**,** 583-591.

Keeney, R. L. & Raiffa, H. (1976). *Decisions with Multiple Objectives: Preferences and Value Tradeoffs,* New York, Wiley.

Lai, R., Garg, M. & Kapur, P. K. (2011). A Study of When to Release a Software Product from the Perspective of Software Reliability Models. *Journal of Software,* 6**,** 651-661.

Li, X., Li, Y. F., Xie, M. & Ng, S. H. (2011). Reliability analysis and optimal version-updating for open source software. *Information and Software Technology,* 53**,** 929-936.

Li, X., Xie, M. & Ng, S. H. (2010). Sensitivity analysis of release time of software reliability models incorporating testing effort with multiple change-points. *Applied Mathematical Modelling,* 34**,** 3560-3570.

Liu, C. T. & Chang, Y. C. (2007). A reliability-constrained software release policy using a non-Gaussian Kalman filter model. *Probability in the Engineering and Informational Sciences,* 21**,** 301-314.

Lyu, M. R. (1996). *Handbook of Software Reliability Engineering,* New York, McGraw Hill.

Musa, J. D., Iannino, A. & Okumoto, K. (1987). *Software Reliability: Measurement, Prediction, Application,* New York, McGraw Hill.

Nan, N. & Harter, D. E. (2009). Impact of budget and schedule pressure on software development cycle time and effort. *IEEE Transactions on Software Engineering,* 35**,** 624-637.

Nelson, W. (1982). *Applied Life Data Analysis,* New York, John Wiley & Sons.

Ngo-The, A. & Ruhe, G. (2009). Optimized resource allocation for software release planning. *IEEE Transactions on Software Engineering,* 35**,** 109-123.

Okumoto, K. & Goel, A. L. (1980). Optimum release time for software systems, based on reliability and cost criteria. *Journal of Systems and Software,* 1**,** 315-318.

Pham, H. (1996). A software cost model with imperfect debugging, random life cycle and penalty cost. *International journal of systems science,* 27**,** 455-463.

Pham, H. & Zhang, X. M. (1999). A software cost model with warranty and risk costs. *IEEE Transactions on Computers,* 48**,** 71-75.

Sgarbossa, F. & Pham, H. (2010). A Cost Analysis of Systems Subject to Random Field Environments and Reliability. *IEEE Transactions on Systems, Man, and Cybernetics, Part C-Applications and Reviews,* 40**,** 429-437.

Von Winterfeldt, D. & Edwards, W. (1986). *Decision Analysis and Behavioral Research,* Cambridge, Cambridge University Press.

Xie, M. (1991). *Software reliability modelling,* Singapore, World Scientific.

Xie, M. & Hong, G. Y. (1998). A study of the sensitivity of software release time. *Journal of Systems and Software,* 44**,** 163-168.

Xie, M. & Yang, B. (2003). A study of the effect of imperfect debugging on software development cost. *IEEE Transactions on Software Engineering,* 29**,** 471-473.

Yamada, S., Narihisa, H. & Osaki, S. (1984). Optimum release policies for a software system with a scheduled software delivery time. *International journal of systems science,* 15**,** 905-914.

Yamada, S. & Osaki, S. (1985). Cost-reliability optimal release policies for software systems. *IEEE Transactions on Reliability,* R-34**,** 422-424.

Yang, B., Hu, H. & Jia, L. (2008). A study of uncertainty in software cost and its impact on optimal software release time. *IEEE Transactions on Software Engineering,* 34**,** 813-835.

Yang, B. & Xie, M. (2000). A study of operational and testing reliability in software reliability analysis. *Reliability Engineering & System Safety,* 70**,** 323-329.

Biography

Rui PENG obtained his Ph. D in Industrial & Systems Engineering Department in National University of Singapore in 2011. Currently he is an assistant professor in Dongling School of Economics & Management, University of Science & Technology Beijing. His research interests include software reliability, network reliability, and system defense.

Yan-Fu LI is an assistant professor at Ecole Centrale Paris (ECP) & Ecole Superieure d'Electricite (SUPELEC), Paris, France. Dr. Li completed his PhD research in 2009 at National University of Singapore, and went to the University of Tennessee as a research associate. His current research interests include system reliability modeling, uncertainty modeling and analysis, evolutionary computing, and Monte Carlo simulation. He is the author of more than 40 publications, all in refereed international journals, conferences, and books. He is an invited reviewer of 20 international journals. He is a member of IEEE.

Jun-Guang ZhANG obtained his Ph. D in School of Economics and Management, Beijing University of Posts and Telecommunications. He is currently an associate professor in

Dongling School of Economics & Management, University of Science & Technology Beijing. His research focuses on software project management.

Xiang LI obtained his Ph. D in Industrial & Systems Engineering Department in National University of Singapore in 2011. Currently he is working at OCBC, Singapore.