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 Dongling School of Economics and Management, University of Science & Technology Beijing, Beijing, China
b Ecole Centrale Paris–SUPELEC, Paris, France
c Industrial & Systems Engineering, National University of Singapore, Singapore

Published online: 20 Aug 2013.

To cite this article: International Journal of Systems Science (2013): A risk-reduction approach for optimal software release time determination with the delay incurred cost, International Journal of Systems Science

To link to this article: http://dx.doi.org/10.1080/00207721.2013.827261
A risk-reduction approach for optimal software release time determination with the delay incurred cost

Rui Peng\textsuperscript{a}, Yan-Fu Lib\textsuperscript{b,*}, Jun-Guang Zhang\textsuperscript{a} and Xiang Li\textsuperscript{c}

\textsuperscript{a}Dongling School of Economics and Management, University of Science & Technology Beijing, Beijing, China; \textsuperscript{b}Ecole Centrale Paris–SUPELEC, Paris, France; \textsuperscript{c}Industrial & Systems Engineering, National University of Singapore, Singapore

\textsuperscript{*}Corresponding author. Email: yanfu.li@ecp.fr

(Received 13 January 2013; final version received 5 July 2013)

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 the modern society. Software reliability is of great importance for the stable operation of such systems. To ensure the reliability, the software needs to be systematically tested prior to its release in 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, Iannino, and Okumoto 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 advise 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, Narihisa, and Osaki 1984; Yamada and Osaki 1985; Pham 1996; Pham and Zhang 1999; Xie and Yang 2003; Huang and Lyu 2005; Boland and Ni Chuiv 2007; Liu and Chang 2007; Ho, Fang, and Huang 2008; Xie, Li, and Ng 2011; Lai, Garg, and Kapur 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, minimising 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 maximise the expected net gain in reliability. Pham (1996), Xie and Yang (2003) and Boland and Ni Chuiv (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) emphasised 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 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 to Yamada and Osaki (1985), 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 customers generally have a minimum reliability requirement, which can be specified in the contract. In order to check whether the reliability requirement is met, 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 a SRM are deterministic and the reliability estimate is accurate (Yamada and Osaki...
time-to-failure data. Parameter uncertainty arises since the estimated parameters are subject to the random variations (or noises) in the data (Dai, Xie, Long, and Ng 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), the resulting optimal release time $T$ given $R_0$ is asymptotically normal with mean $\hat{T}$ and variance $\text{Var}(\hat{T})$. Here, $\hat{T}$ is obtained from solving (1) with the estimated parameters, and $\text{Var}(\hat{T})$ is the variance of $\hat{T}$. Details about these quantities are presented in Appendix 1.

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

$$r_0(t) = P(\hat{R}(x|t) < R_0) = P(t < T)$$

$$= 1 - \Phi \left( \frac{t - \hat{T}}{\sqrt{\text{Var}(\hat{T})}} \right),$$

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

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

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 can be seen from Equation (3), the release time based on risk reduction requires a delay of $z_{r_0}\sqrt{\text{Var}(\hat{T})}$, 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 $C(t)$, then the delay incurred cost at time $t$ ($t > \hat{T}$) is obtained as

$$C_p(t) = C(t) - C(\hat{T})$$
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, 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 attractiveness of the conjoint outcome of the different attributes. The additive form of the multi-attribute utility function is given by

\[ U(d_1, d_2, \ldots, d_n) = \sum_{i=1}^{n} w_i u(d_i), \]

where each attribute is denoted by \( d_i \), \( i = 1, 2, \ldots, n \), 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 maximising 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 typical management scenarios can be appropriately represented within its structure. In the decision problem formulated, there are two competing objectives to be balanced: minimising the risk and minimising 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 the least, when managers have other requirements, i.e. the minimisation 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 optimised 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 in the following sections.

3.1 Elicitation of single utility function for each attribute

After the quantification of each attribute by Equations (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 \( r^1_0 \), respectively. In real applications, they provide the lowest and the highest satisfactions to the managers, respectively. Cukic, Gunel, Singh, and Lan (2003) suggest 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 \( r^1_0 = 0 \). At these boundary conditions, we have \( u(r^0_0) = 0 \) and \( u(r^1_0) = 1 \). The superscript of \( r^i_0 \), \( i \in [0, 1] \) is used to represent the corresponding utility value which is determined so that the management is indifferent towards the following two alternatives: (1) getting risk \( r^0_0 \) with certainty; (2) getting risk \( r^1_0 \) 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):

\[ u(r_0) = \alpha + \beta r_0 \text{ or } u(r_0) = \alpha + \beta \exp(\gamma r_0), \]

where \( \alpha \), \( \beta \) and \( \gamma \) are constants which ensure \( u(r_0) \in [0, 1] \).

To determine which form in Equation (6) should be used, we can compare the certainty equivalent \( u(50\%) \) and the expected value of 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

\[ u(r_0) = 1 - \frac{r^1_0 - r^0_0}{r^0_0 - r^1_0}, \]

Otherwise, the managers are not risk-neutral and the exponential form should be adopted.

In this case, the utility function can be written as

\[ u(r_0) = \frac{\exp(\gamma r^0_0) - \exp(\gamma r_0)}{\exp(\gamma r^0_0) - \exp(\gamma r^1_0)}, \]

where \( \gamma \) is the non-zero solution to \( \exp(\gamma r^1_0) + \exp(\gamma r^0_0) - 2 \exp(\gamma r^0_0^0.5) = 0 \).
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 $(r_0^0, C_p^0)$ comprising the lowest risk and the highest delay incurred cost; (2) the lottery consists of both attributes at their best levels $(r_1^0, C_p^1)$ 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 through the lottery having a 50–50 chance of occurrence. If the managers prefer the certain outcome, 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 Maximisation of the multi-attribute utility function

By maximising the multi-attribute utility function

$$\begin{align*}
U(r_0(t), C_p(t)) &= w_1 u(r_0(t)) + w_2 u(C_p(t)),
\end{align*}$$

the optimal risk-reduction release time is obtained as $T^* = \arg\max\{U(r_0(t), C_p(t))\}$. It is worth noting that Equation (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, De Almeida, and Cavalcante 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 Equation (2). For the cost attribute, the generalised cost model is used and it is quantified by Equation (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 Equation (7). The optimal risk-reduction release time is determined by maximising 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 analyse 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 SRMs are adopted. Moreover, future research can be carried out to analyse 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

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

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

$$R(x | t) = \exp[-\lambda(t)x]$$

and $R(x | t)$ 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

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

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 $T = 41.479$. Moreover, from the standard statistical analysis (Nelson
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

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

where \( c_1 \) is the software test cost per unit time, \( \kappa \) 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 \( \mu_y \) 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 as 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

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

**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 $15,000 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 \( C_p^0 = 15,000 \) and the highest satisfactory cost expectation is \( C_p^1 = 0 \). The single utility function for the delay incurred cost is determined as \( u(C_p) = 1 - C_p/15,000 \).

**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 consisting 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 1, \( w_2 \) is equal to 0.5 as well.

**Step 4: maximisation of multi-attribute utility function**

Based on the estimated single utility functions and scaling constants, the multi-attribute utility function can be obtained by Equation (7). Figure 1 shows this multi-attribute utility function as a function of the release time. This multi-attribute utility function is maximised when \( T^* = 50.586 \) and the corresponding risk and delay incurred cost at this time are \( r_0(T^*) = 0.18\% \) and \( C_p(T^*) = 5002.9 \), respectively. As a result, software should be released at the optimal risk-reduction release time \( T^* = 50.586 \) to appropriately compromise between reducing the risk and controlling the delay incurred cost.

### 4.2 Illustration of the proposed decision model

In Figure 3, we denote \( \hat{T} = 41.479 \) as the mean value of the release time without the 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 maximises the multi-attribute utility function is \( T^* = 50.586 \), and the corresponding risk and delay incurred cost are \( r_0(T^*) = 0.18\% \) and \( C_p(T^*) = 5002.9 \), respectively.
Finally, it should be noted that during the time periods $[\hat{T}, T(\tau_0)]$ and $[T(C_0^p), +\infty)$, 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 the release time is greater than $T(C_0^p)$. It implies that the available cost budget $C_0^p = 15,000$ 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 maximising 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 fixed levels (Xie and Hong 1998; Li, Xie, and Ng 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).

$$S_{q,x} = \frac{T^*_R(x + qx) - T^*_R(x)}{T^*_R(x)}$$

A large value of $S_{q,x}$ indicates that parameter $x$ has a significant impact on 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 $c_2$
Table 1. Sensitivity analysis results according to different parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>-30%</th>
<th>-20%</th>
<th>-10%</th>
<th>10%</th>
<th>20%</th>
<th>30%</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S_{q,w_1}$</td>
<td>-1.36%</td>
<td>-0.88%</td>
<td>-0.43%</td>
<td>0.42%</td>
<td>0.84%</td>
<td>1.27%</td>
</tr>
<tr>
<td>$S_{q,c_2}$</td>
<td>-0.77%</td>
<td>-0.48%</td>
<td>-0.22%</td>
<td>0.20%</td>
<td>0.38%</td>
<td>0.55%</td>
</tr>
<tr>
<td>$S_{q,\kappa}$</td>
<td>0.74%</td>
<td>0.47%</td>
<td>0.22%</td>
<td>-0.20%</td>
<td>-0.39%</td>
<td>-0.57%</td>
</tr>
<tr>
<td>$S_{q,c_1}$</td>
<td>0.74%</td>
<td>0.47%</td>
<td>0.22%</td>
<td>-0.20%</td>
<td>-0.39%</td>
<td>-0.57%</td>
</tr>
<tr>
<td>$S_{q,\kappa}$</td>
<td>2.91%</td>
<td>1.96%</td>
<td>0.99%</td>
<td>-0.53%</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

and $\mu_y$ 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 $\mu_x = 0.1$, compared with the estimated value of $c_1$, the product of $c_2$ and $\mu_x$ is too small to have any impact on the delay incurred cost function in Equation (12). Another example is the determination of $r_1$ and $C_0$, 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 $\kappa$ are much more important since they dominate the change of the delay incurred cost over time. Similarly, $r_0$ and $C_0$ are of importance as shown in Figure 1, where $T(r_0)$ and $T(C_0)$ are the changing points of the 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 the attributes, which directly affect the optimal solution of $T_R$. Since the sum of these two weights is equal to 1, investigating one factor is sufficient. Results of sensitivity analysis with regard to these parameters are summarised in Table 1. Specially, since parameter $\kappa$ 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 $\kappa$.

It can be seen that these parameters do not significantly influence the final solution on $T_R$ since all the absolute values of $S_{q,\kappa}$ 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 $C_0$, and negatively correlated with $r_0$, $c_1$ and $\kappa$. 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 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 the 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 the optimal release time can be obtained as $T_{\text{real}} = 52.73$. According to the general procedures discussed in Lyu (1996), 10,000 failure data-sets are generated, and each failure data-set is composed of 90 time-to-failure data points. Since each failure data-set can produce an estimate of the optimal release time denoted by $\hat{T}$, the risk that the software cannot meet the reliability requirement can be easily estimated by comparing these $\hat{T}$ values with $T_{\text{real}}$, and such risk is estimated as $r_0 = 60.21\%$. 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) with 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 modelled 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 limited 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 might not be sufficient for release time determination. In reality, managers can also have other requirements, which may include the minimisation of the total cost in the software development cycle (Sgarbossa and Pham 2010), the control over the uncertainty in the total cost function (Yang et al. 2008), the optimised 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 parameter uncertainty in the SRM. However, reducing such a risk inevitably increases 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 optimise 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 analyse the effect of choosing different SRMs on the estimated optimal release time and the risk of not fulfilling the reliability requirement.

Acknowledgements
The research is partially supported by Natural Science Foundation of China (NSFC) through grant number 71231001, China Postdoctoral Science Foundation funded project, under grant number 2013MS30531, and the fundamental research funds for the central universities through grant numbers FRF-TP-13-009A and FRF-TP-13-026A.

Notes on contributors
Rui Peng obtained his PhD from Department of Industrial & Systems Engineering of 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 defence.

References
Ho, J.W., Fang, C.C., and 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.

Appendix 1

Parameters in a 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 \( \text{Var}(\hat{T}) \).

Suppose that there are total \( m \) model parameters to be estimated, \( \theta_1, \theta_2, \ldots, \theta_m \). Let \( n_i \) denote the number of failures observed within each time interval \([t_{i-1}, t_i)\), where \( 0 = t_0 < t_1 < \cdots < t_k = t \) 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

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

By maximising 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 MLE (Nelson 1982). In particular, the Fisher information matrix is obtained as

\[
I(\theta_1, \ldots, \theta_m) = \begin{bmatrix}
\frac{\partial^2 \ln L}{\partial \theta_1^2} & \cdots & \frac{\partial^2 \ln L}{\partial \theta_1 \partial \theta_m} \\
\vdots & \ddots & \vdots \\
\frac{\partial^2 \ln L}{\partial \theta_m \partial \theta_1} & \cdots & \frac{\partial^2 \ln L}{\partial \theta_m^2}
\end{bmatrix}
\]

\[
= \begin{bmatrix}
-\frac{\partial}{\partial \theta_1} \frac{\partial^2 \ln L}{\partial \theta_1^2} & \cdots & -\frac{\partial}{\partial \theta_1} \frac{\partial^2 \ln L}{\partial \theta_1 \partial \theta_m} \\
\vdots & \ddots & \vdots \\
-\frac{\partial}{\partial \theta_m} \frac{\partial^2 \ln L}{\partial \theta_1 \partial \theta_m} & \cdots & -\frac{\partial}{\partial \theta_m} \frac{\partial^2 \ln L}{\partial \theta_m^2}
\end{bmatrix}
\]

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

\[
V = \left( I^{-1} \right)
= \begin{bmatrix}
\text{Var}(\hat{\theta}_1) & \text{Cov}(\hat{\theta}_1, \hat{\theta}_2) & \cdots & \text{Cov}(\hat{\theta}_1, \hat{\theta}_m) \\
\text{Cov}(\hat{\theta}_2, \hat{\theta}_1) & \text{Var}(\hat{\theta}_2) & \cdots & \text{Cov}(\hat{\theta}_2, \hat{\theta}_m) \\
\vdots & \vdots & \ddots & \vdots \\
\text{Cov}(\hat{\theta}_m, \hat{\theta}_1) & \text{Cov}(\hat{\theta}_m, \hat{\theta}_2) & \cdots & \text{Var}(\hat{\theta}_m)
\end{bmatrix}
\tag{16}
\]

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

$$\text{Var}(\hat{T}) = \sum_{i=1}^{m} \left( \frac{\partial T}{\partial \hat{\theta}_i} \right)^2 \text{Var}(\hat{\theta}_i)$$

$$= \frac{[m(1) - m(0)]^2 + \cdots + [m(24) - m(25)]^2 \exp [-m(25) - m(0)]}{27!16! \cdots 1!}$$

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

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

**Appendix 2**

Table 2 shows the software testing data used in Pham and Zhang (1999) that are summarised as number of failures per 1-hour interval of execution time.

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

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

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

$$= \frac{[m(1) - m(0)]^7 \cdots [m(25) - m(24}]^1 \exp \left[ -m(25) - m(0) \right]}{27!16! \cdots 1!} \quad (18)$$

According to Equation (8), we have

$$\ln(L) = 136 \ln(m(1) - m(0)) + 27 \ln(m(2) - m(1))$$

$$+ \cdots + 1 \ln(m(25) - m(24)) - m(25) - \ln(27!16! \cdots 1!)$$

$$= 136 \ln(a) + 27 \ln(1 - e^{-b}) + 16 \ln(e^{-b} - e^{-2b})$$

$$+ \cdots + 1 \ln(e^{-24b} - e^{-25b}) - a(1 - e^{-25b})$$

$$- \ln(27!16! \cdots 1!)$$

Furthermore, we have

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

$$\frac{\partial \ln(L)}{\partial b} = 27 e^{-b} + 16(2 e^{-2b} - e^{-b})$$

$$+ \cdots$$

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