Pricing and Contract Design for Software Products in the Presence of Requirements Change

Pandu Dwi Luhur Pambudi 1,2*, Cheng-Han Wu 1, Mahendrawathi ER 2

1 National Cheng Kung University, Tainan, Taiwan
2 Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia

* luhur.pandu@gmail.com, wuchan@mail.ncku.edu.tw, mahendra_w@is.its.ac.id

Abstract. The presence of software requirements change (RC) during project development is a critical challenge for the developer to offer software contract designs. Because under the presence of RC, the decisions toward the contract offer will impact to project’s price spent by the developer. Managers of software companies must decide what contract designs to offer to clients in the development of software. Abstracting from an example drawn from the software industry, we exhibit three designs of software contracts incorporating fixed price (FP) and time-and-materials (T&M) policies. Specifically, a software company offers a fixed-price and agree to RC with adjustment cost (SW contract), or initially provides a fixed price and then charges an additional fee based on the time-and-material in response to RC (MP contract). We examine the strategic choices of two contract designs in a two-stage game. We carry out a full analysis of distinct settings of monopoly model. We characterize the conditions under which the contracts can be the best decision for developers. As the result, we found that the MP contract performed better when accommodating the presence of RC instead of the SW contract.

1. Introduction

In every software development project, we use contracts to regulate project work and results. Throughout substantial evidence and proven theories over decades, some experts introduce and promote success variety of contract types, but two contract types have dominated and still dominate most work of software projects: fixed price (FP) and time-and-materials (T&M) types of contracts. The two contracts are frequently used in the software industry, as discussed in [1]–[3]. The FP type of contract is when the client agrees to pay the developer a specific price for a delivered software product. Meanwhile, A T&M type of contract is when the client agrees to pay for the effort spent by the developer, usually based on the price agreed by both sides per hour of work for different types of competence and other necessary expenditures [3]. In the FP contract, the price for completing the project is predetermined in advance. Conversely, T&M contract does not specify a price, but rather reimburse the vendor for its costs plus a predetermined profit [1]. Despite the extensive study in software contract, the issues related to contract choice and performance for software projects remain unclear. Those conditions explained by [4]; because of the high complexity in software engineering processes, software contracts pose many unique challenges that are usually not seen in contracts in other industries. We capture two main problems that challenge a developer to offer a software contract due to the presence of requirement change (RC).
Incomplete initial requirements and difficulty in accommodating the RC are two of these challenges. These problems could make software contract challenging to manage by often incurring significant cost overruns from RC. However, there is little research that provides a clear understanding and guidance on the choice and performance of contracts in terms of price perspective. Based on the problems, we are motivated to study the contract choice and design. We examine the adoption of the two contracts in the software industry: fixed-price (FP) contract and time-and-materials contract (T&M). Our research contributions are threefold, as explained below:

a. Formulation of the determinants of contract choices. This study is one of the papers to analytically study the determinants of contract choice in the software industry. Our paper is one of the papers to analytically study the determinants of contract choice in the software industry including [4]–[6] but of course we design our model differently with the two-stages game. Although there is some discussion of the contract types in the [1]–[3], [7], [8], they discussed their study in an empirical way.

b. Discussion on the impacts of competition. We discuss this study into one setting, which is monopoly model.

c. Investigation of the equilibrium pricing decisions and contract choices. We address the linkage between contract designs and project profits in the presence of RC. We aim to study this problem at a project level by modeling the characteristics of a software contract based on established theories in software development.

2. Literature Review

This study relates to three streams of literature: one examines the RC present analytically in the software industry, second provides a survey on software pricing, and the third are types of software contract design. We present the primary previous studies considerations, specifically the analytical and theoretical of the observed studies.

2.1. RC in The Software Industry

In any software development activity, RCs are inevitable and can occur due to changes in user requirements, increased understanding of the stakeholders’ needs, customer organizational restructure, and availability of new technologies [9], [10]. With every change in requirements, a developer can be affected by the change of the overall cost, quality, and schedule of the software, which is why the RCs as one of the significant causes of software failure [11]. However, dealing with RCs not only poses a risk to the successful delivery of software but also provide an opportunity for improved usability, value, and enhance software development process [12], [13]. Topics related to the first stream have been examined in the field of computer science and software development projects, e.g., [14]–[16]. These topics mainly discussed how RC can affect software development in theoretical and analytical ways.

In this study, we use the term of Requirements Change (RC) as a description from [15]. RC is the number of changes (addition, deletion, and modification) in each period of the development life cycle. It means that every addition, deletion, and modification of software requirements are referred to as RC.

2.2. Pricing in The Software Industry

Software companies often struggle with communicating the value of their solution to their customers and, as a result, lack of profits at the end. Many researchers tried to optimize and analyze the software pricing with some scenarios of strategy. They develop a pricing strategy that is tailored to customer segments that optimizes a company’s financial goals. Here we summarize the second stream surveys about pricing in the software industry. Studies related to this topic are broadly discussed in the field of information systems, operation management, and industrial management. e.g., [17]–[22].
2.3. Software Contract Policy
The third stream examines software contract policy. A software contract is a binding agreement between the software developer and the client in the outsourcing mechanism. The contract provides the development policy of the software to the client. A typical software development contract must deal with a variety of closely related issues such as the quality of the developed system, the timeliness of delivery, the effort and cost associated with the project, and the support [4]. The economists have studied contract design for at least 40 years (see McCall, 1970) in [23]. Meanwhile, [8] examined the use of fixed-fee and time-and-materials (T&M, or cost-plus) contracts and a hybrid contract that consisted of a T&M contract with a cap.

We resume our survey related to the previous studies into the table as seen in Table 1. We survey those studies based on the criteria set in the beginning through keywords searching from reputable journal databases. As we can see in Table 1, we highlight our study with a thick font that describes if this study has a novelty aspect compared to the previous studies. We consider the presence of RCs and duopoly market as the external competition which never discussed before in the contract of software development.

| Author(s) | RC Presence | External Competition | Software Pricing | Software Contract Type | Decision Variable |
|-----------|-------------|----------------------|------------------|------------------------|-------------------|
| This study | √           | √                    | √                | √                      | Price             |
| [17]      | -           | √                    | -                | -                      | Price             |
| [16]      | √           | -                    | -                | -                      | RC                |
| [5]       | √           | -                    | √                | √                      | Price             |
| [20]      | -           | -                    | √                | -                      | Price             |
| [19]      | -           | √                    | √                | -                      | Price             |
| [24]      | -           | √                    | √                | -                      | Price, quality    |
| [6]       | √           | -                    | -                | √                      | Work rate         |
| [21]      | -           | √                    | √                | -                      | Price             |
| [25]      | -           | √                    | √                | -                      | Price             |
| [4]       | √           | -                    | √                | √                      | Effort, release time |
| [26]      | -           | √                    | -                | -                      | Price, time       |

3. The Model

3.1. Model Setup
In this section, we analyze the two most common types of contracts, namely fixed-price (FP) contract and time and materials (T&M) contract. We extend the FP contract into one contract design. That is single price with second stage adjustment (SW) and T&M as the manual price (MP) contract design.

We build our model in a monopoly model between developer and client (see Figure 1). The developer will offer two different software contracts. As the model is built in a monopoly market, we do not consider the competition with other developers, because there is a case when the client chooses directly one software developer to execute their project. This condition usually happens when the developer is a loyal vendor for that client. We want to highlight under which software contract the developer as monopolist can obtain the best profit.

3.2. Two-stages model
We set this model into two stages of contract offers. At the first stage of the contract, the developer offers a price portion model to accommodate the main wage cost coming from the initial requirement of software in stage 1. Meanwhile, at the second stage of the contract, the developer offers a price portion model to accommodate the RC cost after stage 1 is completed.
3.3. Monopoly Model

In this section, we study contract designs decision for a monopoly selling a software product based on three contract designs. We derive the conditions under which the developer as a monopolist will offer SN, SW, or MP to the client. In this monopoly model, we model the contract designs to complete the initial requirements with the price portion of $\alpha$ in stage 1 and accommodate the RC with the price portion of $1 - \alpha$ in stage 2. These price portion models are developed for SN and SW models due to the explanation in Chapter 3.1. Otherwise, The MP contract does not consider the price portion, so we remove the portion of $\alpha$ in stage 1 and $1 - \alpha$ in stage 2. Following prior literature [2], [4], we also use contract design profit or surplus to measure the performance of the contract by maximizing the price and effort. But for this proposal, we firstly execute the price.

The client’s willingness to pay (WTP) as $S$ increases with the quality of the software $q$ and consumer valuation $\theta$. Thus, the client’s WTP can be expressed $S = \theta q$. Given the software quality $q$ as a function of effort as $e$ and time to complete effort as $\tau$. We describe the model as $q = e \tau$. We assume that the effort $e$ and the software quality $q$ are private information of the developer and are not directly observable by the client. However, in all settings, software quality may be either higher or lower in WTP. So, in our model, all types of clients coexist (1) low quality, low WTP, and (2) high quality, high WTP. But in our model, we assume that the consumer valuation $\theta$ is always positive. Let $\theta$ denote the consumer valuation to quality that closely related to the features of the software development product. Even though some clients with greater knowledge and experience in software development usually understand the contract designs in more detailed terms and conditions. That is why we assume both the client and the developer are risk-neutral.

We consider the benchmark case where the developer offers three contract designs. We build the model step by step by solving the utility function, demand function, and developer’s problem by solving the profit, after that, we derive the optimal price for the developer’s problem. We explain the steps as follows:

3.3.1. Utility and Demand Function

Let $N$ be the number of clients with a positive valuation for the software contract designs. We normalize $N$ to 1 for the sake of simplicity and to capture the real event. Client’s valuation for the software is denoted by $\theta$ and uniformly distributed over $[0,1]$. In our model, we assume clients always in positive valuation. Figure 1 shows the demand for the software contract designs.

![Figure 1. Demand for software contract designs](image)

We assume the demand $D_A$ as a possible demand for the client to choose the contract designs. The client’s valuation for the software is denoted by $\theta$ and uniformly distributed over $[0,1]$. Therefore, it corresponds to the demand $D_A$ for all of the software contract, a possible demand can be expressed as $D_A = 1 - \theta$.

The monopoly offers one software development product under SN contract with software utility $U = S - \alpha P_A$ and SW contracts with software utility $U = (1 + \delta)S - P_A$ and under MP contract with software utility $U = (1 + \delta)S - P_A1 - P_A2$. We model WTP as $S = \theta q$, let $\theta$ as client’s positive valuation for the software and $q$ as quality at a price $P_A$ this function applied in the stage 1. Meanwhile, in the stage 2 when the RC is acted, the $\delta$ added as primary value addition due to RC.
this phase, we assume that the project is always delivered on time to the client, so there is no variable to decrease the utility of the software due to the project delay. Thus, the expected software utility of the client for SN contracts who have zero net utility can be expressed as $S - P_A = 0$. The software utility of the client for SW contracts who have zero net utility can be expressed as $(1 + \delta)S - P_A = 0$ while software utility of the client for MP contract who has zero net utility can be expressed as $(1 + \delta)S - P_{A1} - P_{A2} = 0$. After that, we solve $\theta$ and get $\theta = \alpha P_A/e \tau$ for SN, $\theta = P_A/e(1 + \delta)\tau$ for SW and $\theta = P_{A1} + P_{A2}/e(1 + \delta)\tau$ for MP contract. A client with a higher primary valuation of quality $q$ than the consumer valuation $\theta$ has higher WTP $S$, that leads to positive utility and will buy the software. Otherwise, a client with a lower primary valuation of quality $q$ than the consumer valuation $\theta$ will not buy the software. In SW and MP contract we have $\delta$ as the addition of primary valuation due to RC. We solve the demand $D^S_A$ for SN contract $D^S_A = 1 - \alpha P_A/(e \tau)$, demand $D^S_A$ for SW contract $D^S_A = 1 - P_A/e(1 + \delta)\tau$ and demand $D^M_A$ for MP contract $D^M_A = 1 - P_{A1} + P_{A2}/e(1 + \delta)\tau$.

3.3.2. SW Contract Design
We model the developer’s problem under SW contract, as follows:

$$\max_{P_A} \pi^{SW}_{A} = [(\alpha P_A - C_1)D^S_A + ((1 - \alpha)P_A - C_2)]D^S_A$$  \hspace{1cm} (1)

The direct development cost incurred by the developer denoted as $C_1 = c_1 e^2$. The support cost incurred by the developer denoted as $C_2 = c_2 \Delta^2$. $c_2$ is the adjustment cost and $\Delta$ is the level of RC, where $\Delta \leq 1$ is always less than equal with 1. The support cost captures the fact that higher quality software will incur a higher support cost and vice versa. This condition also related to the level of RC, the higher level of RC $\Delta$ the higher support cost $C_2$ and the higher software quality obtained. First, we simply set the two different costs without discounting them even though they are typically incurred at different time points. This is because we assume that the cost parameters ($c_1$, and $c_2$) in the model already account for the necessary discounting.

**Proposition 1.** The optimal price for the developer’s problem under SW is

$$P^{SW}_A = \frac{1}{2} \left( c_2 \Delta^2 + e(c_1 e + \tau + \delta \tau) \right)$$  \hspace{1cm} (2)

**Proof of Proposition 1.** The first-order condition for the developer’s problem in SW contract is

$$\frac{\partial \pi^{SW}_{A}}{\partial P_A} = [(\alpha P_A - C_1)D^S_A + ((1 - \alpha)P_A - C_2)]D^S_A$$  \hspace{1cm} (3)

and from Eq. (3), we get optimal price $P^{SW}_A$

$$P^{SW}_A = \frac{1}{2} \left( c_2 \Delta^2 + e(c_1 e + \tau + \delta \tau) \right)$$  \hspace{1cm} (4)

Because we only consider price as our decision variable, we derive second profit to see whether the objective function in $P^{SW}_A$ is concave or convex, so we get the result of second derivation, as follows:

$$\frac{\partial^2 \pi^{SW}_{A}}{\partial P_A^2} = - \frac{2(1 - \alpha)}{e(1 + \delta)\tau} - \frac{2\alpha}{e(1 + \delta)\tau}$$  \hspace{1cm} (5)

Therefore, the objective function is concave because $\frac{\partial^2 \pi^{SW}_{A}}{\partial P_A^2} < 0$ and is the optimal solution.
3.3.3. MP Contract Design

We model the developer’s problem under the MP contract by solving it using a backward induction. We firstly solve the price 2 as \( P_{A2}^{MP} \) in stage 2 by solving the Eq. (6). After that, we calculate both the profit 1 and profit 2 as \( \pi_A^{MP} = \pi_{A1}^{MP} + \pi_{A2}^{MP} \) and we derive the price 1 as \( P_{A1}^{MP} \) in stage 1 by solving Eq. (7). Then, we backward the \( P_{A2}^{MP} \) by solving \( \pi_A^{MP} = \pi_{A1}^{MP} + \pi_{A2}^{MP} |_{P_{A2}^{MP}} \) in Eq. (8). We express our model, as follows:

\[
\begin{align*}
\text{Max} \; \pi_{A2}^{MP} &= [(P_{A2} - C_2)]D_{A2}^{MP} \quad (6) \\
\pi_A^{MP} &= \pi_{A1}^{MP} + \pi_{A2}^{MP} |_{P_{A2}^{MP}} \quad (7) \\
\text{Max} \; \pi_{A1}^{MP} &= [(P_{A1} - C_1)]D_{A1}^{MP} \quad (8)
\end{align*}
\]

In this MP contract design, we don’t include the price portion \( \alpha \) both in stages 1 and 2, but we include \( \delta \) as addition towards primary valuation due to the level of RC.

**Proposition 2.** The optimal prices for the developer’s problems under MP are

\[
\begin{align*}
P_{A2}^{MP^*} &= \frac{1}{2}(\Delta^2 c_2 + e(c_1 e + \tau + \delta \tau)) \quad (9) \\
P_{A1}^{MP^*} &= e^2 c_1 \quad (10)
\end{align*}
\]

**Proof of Proposition 2.** The first-order condition for the developer’s problem in MP contract is using standard backward induction. We first examine the first-order condition for the developer’s problem in Eq. (7,8, and 9). The first-order condition is

\[
\frac{\partial \pi_{A2}^{MP}}{\partial P_{A2}^{MP}} = [(P_{A2} - C_2)]D_{A2}^{MP} = 0 \quad (5)
\]

and from Eq. (11), we get

\[
P_{A2}^{MP} = \frac{1}{2}(c_2\Delta^2 + e(1 + \delta)\tau - P_{C1}) \quad (6)
\]

After that, we calculate the profit 1 and profit 2 as \( \pi_A^{MP} = \pi_{A1}^{MP} + \pi_{A2}^{MP} \) and get the result, as follows:

\[
\pi_A^{MP} = (-c_2 e^2 + P_{A1}) \left(1 - \frac{P_{A1} + P_{A2}}{e(1 + \delta)\tau}\right) + (-c_2 \Delta^2 + P_{A2}) \left(1 - \frac{P_{A1} + P_{A2}}{e(1 + \delta)\tau}\right) \quad (7)
\]

and derive the price 1 as \( P_{A1}^{MP^*} \) by substituting Eq. (13) into Eq. (12). We get the optimal price 1 \( P_{A1}^{MP^*} \), as follows:

\[
P_{A1}^{MP^*} = e^2 c_1 \quad (8)
\]

After that, we backward the price 2 \( P_{A2}^{MP} \) by solving \( \pi_A^{MP} = \pi_{A1}^{MP} + \pi_{A2}^{MP} |_{P_{A2}^{MP}} \). We get the optimal price 2 \( P_{A2}^{MP^*} \), as follows:

\[
P_{A2}^{MP^*} = \frac{1}{2}(\Delta^2 c_2 + e(c_1 e + \tau + \delta \tau)) \quad (9)
\]

Now, we look at the Hessian matrix, which is given by

\[
\text{det}(H^{MP}) = \begin{bmatrix}
\frac{\partial^2 \pi_A^{MP}}{\partial P_{A2}^2} & \frac{\partial \pi_A^{MP}}{\partial P_{A2}} \\
\frac{\partial \pi_A^{MP}}{\partial P_{A1}} & \frac{\partial^2 \pi_A^{MP}}{\partial P_{A1}^2}
\end{bmatrix} = 0 \geq 0.
\]
It is easy to see that $H^{MP}$ is negative semidefinite for any value of $P_{A2}^{MP}$ and $P_{A1}^{MP}$. This is because $\det(H^{MP}) \geq 0$. We derive second profit to see whether the objective function in $P_{A2}^{MP}$ and $P_{A1}^{MP}$ are concave or convex, so we get the result of second derivation, as follows:

$$\frac{\partial^2 \pi^{MP}}{\partial P_{A2}^2} = -\frac{2}{e(1 + \delta)\tau} < 0$$  \hspace{1cm} (16)$$

and

$$\frac{\partial^2 \pi^{MP}}{\partial P_{A1}^2} = -\frac{2}{e(1 + \delta)\tau} < 0$$  \hspace{1cm} (17)$$

Therefore, the objective functions in $P_{A2}^{MP}$ and $P_{A1}^{MP}$ are concave, and the equilibrium results are the optimal solutions.

4. Model Analysis and Result

Proposition 1 suggests that under SW, the developer does accommodate development cost $C_1$ in stage 1 and support cost $C_2$ in stage 2. This SW contract is designed to solve the problem found in the SN contract. This means after completing the initial requirement in stage 1, the developer will continue to perform stage 2 for executing the RC. The cost of mainly wages $c_1$ will increase when the effort $e$ spent to do the initial requirement in stage 1 is high. Meanwhile, when the level of RC increases $\Delta$, the adjustment cost $c_2$ will increase. This is because the developer requires to do the additional RC proposed by the client.

We can also see the quality of software $q$. When the effort $e$ increases, the time to complete the effort $\tau$ will also increase. At the end it will leads to the higher mainly wages $c_1$ and adjustment cost $c_2$. This indicates when the initial requirement proposed by the client is high, the developer will set up the higher effort $e$ and the higher mainly wages $c_1$ in advance. Meanwhile, the higher level of RC $\Delta$, means the higher adjustment cost $c_2$. This indicates when the additional RC proposed by the client is high, the developer will set up a higher level of RC $\Delta$ by working on that as well as the primary value will increase due to the addition of RC $\delta$. That is why it reflects the higher adjustment cost $c_2$. We suggest these contracts for a client who wants to improve their software quality by adding the RC.

Meanwhile, Proposition 2 suggests that under MP, the developer accommodates the support cost $C_2$ in stage 2 first and continue with the development cost $C_2$ in stage 1. The cost of mainly wages $c_1$ will increase when the effort $e$ spent to do the initial requirement in stage 1 is high. Meanwhile, when the level of RC increases $\Delta$, the adjustment cost $c_2$ will increase. This is because the developer requires to do the additional RC proposed by the client. We can also see the quality of software $q$. When the effort $e$ increases, the time to complete the effort $\tau$ will also increase. At the end it will leads to the higher mainly wages $c_1$ and adjustment cost $c_2$. This indicates when the initial requirement proposed by the client is high, the developer will set up the higher effort $e$ and the higher mainly wages $c_1$ in advance. Meanwhile, the higher level of RC $\Delta$, means the higher adjustment cost $c_2$. This indicates when the additional RC proposed by the client is high, the developer will set up a higher level of RC $\Delta$ by working on that as well as the primary value will increase due to the addition of RC $\delta$.

We design this contract under backward induction by solving stage 2 first and obtain the price 2 and following stage 1 and obtain the price 1. We intend if the client will propose the additional RC during the project that will influence the initial requirement in stage 1. In this MP contract, we do not require the client to complete stage 1 first and continue to stage 2, but we want to approach the condition when the client can flexibly adding RC as much as they want and capture the initial requirement after it. That is why we will charge the adjustment cost $c_2$ in MP contract always be higher than the adjustment cost $c_2$ in SW contract $c_2^{MP} \geq c_2^{SW}$ as a support cost by the developer to deal with the increasing level of RC $\Delta$. We also do not include the price portion $\alpha$ for both stages since
this MP contract is designed for a client who wants a flexible setting to accommodate the RC. We illustrate the result of Proposition 1 and Proposition 2 using parameter values $e = 0.4, c_1 = 0.2, c_2 = 0.2, \tau = 0.2, \alpha = 0.3, \delta = 0.3$. Meanwhile, for MW contract we set $c_2 = 0.4$ or higher than the SW contract due to the developer’s additional consultant for RC.

![Figure 2. Developer’s profit with the presence of RC](image)

We can see in the figure 2. When level of RC $\Delta$ increased, the profit received by developer will also increase. We see that in MP contract the developer will get more profit with the same level of RC $\Delta$ in SWC contract.

5. Conclusion

In this study we model a monopoly model to know which contract would perform better under the presence of RC. From the result we can see, if the MP contract performed better when accommodating the RC instead of the SW contract. The higher level of RC will lead to the higher profit that developer get in the project.

Our future study will focus on the analysis of model and explore another potential of decision variable. We also want to see this model in a competitive market when dealing with another developers.

References

[1] L. Fink, Y. Lichtenstein, and S. Wyss, “Ex post adaptations and hybrid contracts in software development services,” Appl. Econ., vol. 45, no. 32, pp. 4533–4544, 2013.

[2] A. Gopal, K. Sivaramakrishnan, M. S. Krishnan, and T. Mukhopadhyay, “Contracts in Offshore Software Development: An Empirical Analysis,” Manage. Sci., vol. 49, no. 12, pp. 1671–1683, 2003.

[3] M. Jørgensen, P. Mohagheghi, and S. Grimstad, “Direct and indirect connections between type of contract and software project outcome,” Int. J. Proj. Manag., vol. 35, no. 8, pp. 1573–1586, 2017.

[4] D. Dey, M. Fan, and C. Zhang, “Design and analysis of contracts for software outsourcing,” Inf. Syst. Res., vol. 21, no. 1, pp. 93–114, 2010.

[5] S. Oh, J. Rhodes, and R. Strong, “Impact of cost uncertainty on pricing decisions under risk aversion,” Eur. J. Oper. Res., vol. 253, no. 1, pp. 144–153, 2016.

[6] H. Dharma Kwon, S. A. Lippman, and C. S. Tang, “Optimal time-based and cost-based coordinated project contracts with unobservable work rates,” Int. J. Prod. Econ., vol. 126, no.
2, pp. 247–254, 2010.

[7] M. Suprapto, H. L. M. Bakker, H. G. Mooi, and M. J. C. M. Hertogh, “How do contract types and incentives matter to project performance?,” Int. J. Proj. Manag., vol. 34, no. 6, pp. 1071–1087, 2016.

[8] A. Kalnins and K. J. Mayer, “Relationships and hybrid contracts: An analysis of contract choice in information technology,” J. Law, Econ. Organ., vol. 20, no. 1, pp. 207–229, 2004.

[9] N. Ali and R. Lai, “A method of requirements change management for Global software development,” Inf. Softw. Technol., vol. 70, pp. 49–67, 2016.

[10] S. Basri, N. Kama, F. Hanecem, and S. A. Ismail, “Predicting effort for requirement changes during software development,” pp. 380–387, 2016.

[11] N. Nurmuliani, D. Zowghi, and S. P. Williams, “Requirements Volatility and Its Impact on Change Effort: Evidence Based Research in Software Development Projects.”, Aust. Work. Requir. Eng., 2006.

[12] A. AlSanad and A. Chikh, “The Impact of Software Requirement Change – A Review,” in Á. Rocha et al. (eds.), New Contributions in Information Systems and Technologies, Advances in Intelligent Systems and Computing 353, 2015, vol. 353.

[13] S. McGee and D. Greer, “Towards an understanding of the causes and effects of software requirements change: Two case studies,” Requir. Eng., vol. 17, no. 2, pp. 133–155, 2012.

[14] C. Gaebert, “Dilemma structures between contracting parties in software development projects,” ICSOFT-EA 2014 - Proc. 9th Int. Conf. Softw. Eng. Appl., pp. 539–548, 2014.

[15] C. Mao, Y. Lu, and X. Wang, “A study on the distribution and cost prediction of requirements changes in the software life-cycle,” Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 3840 LNCS, pp. 136–150, 2006.

[16] Z. Tong, X. Su, L. Cen, and T. Wang, “Greening Software Requirements Change Management Strategy Based on Nash Equilibrium,” Wirel. Commun. Mob. Comput., vol. 2017, pp. 1–10, 2017.

[17] R. Bala and S. Carr, “Usage-based pricing of software services under competition,” J. Revenue Pricing Manag., vol. 9, no. 3, pp. 204–216, 2010.

[18] H. K. Cheng, S. Li, and Y. Liu, “Optimal software free trial strategy: Limited version, time-locked, or hybrid?,” Prod. Oper. Manag., vol. 24, no. 3, pp. 504–517, 2015.

[19] V. Choudhary, A. Ghose, T. Mukhopadhyay, and U. Rajan, “Personalized pricing and quality differentiation,” Manage. Sci., vol. 51, no. 7, pp. 1120–1130, 2005.

[20] Y. Liu, H. K. Cheng, Q. C. Tang, and E. Eryarsoy, “Optimal software pricing in the presence of piracy and word-of-mouth effect,” Decis. Support Syst., vol. 51, no. 1, pp. 99–107, 2011.

[21] A. Mehra, R. Bala, and R. Sankaranarayanan, “Competitive behavior-based price discrimination for software upgrades,” Inf. Syst. Res., vol. 23, no. 1, pp. 60–74, 2012.

[22] G. Nan, F. Shi, R. Dou, and M. Li, “Duopoly pricing of software products under free strategy: Limited-feature vs. seeding,” Comput. Ind. Eng., vol. 100, pp. 13–23, 2016.

[23] S. A. Lippman, K. F. McCardle, and C. S. Tang, “Using Nash bargaining to design project management contracts under cost uncertainty,” Int. J. Prod. Econ., vol. 145, no. 1, pp. 199–207, 2013.

[24] H. K. Cheng and Q. C. Tang, “Free trial or no free trial: Optimal software product design with network effects,” Eur. J. Oper. Res., vol. 205, no. 2, pp. 437–447, 2010.

[25] G. Nan, F. Shi, R. Dou, and M. Li, “Duopoly pricing of software products under free strategy: Limited-feature vs. seeding,” Comput. Ind. Eng., vol. 100, pp. 13–23, 2016.

[26] H. K. Cheng, S. Li, and Y. Liu, “Optimal software free trial strategy: Limited version, time-locked, or hybrid?,” Prod. Oper. Manag., vol. 24, no. 3, pp. 504–517, 2015.