IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received September 3, 2019, accepted October 20, 2019, date of publication October 23, 2019, date of current version November 7, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2949218

Tradeoff Analysis for Conflicting Software Non-Functional Requirements

XUAN ZHANG[1,2] AND XU WANG[3]
[1]School of Software, Yunnan University, Yunnan 650091, China
[2]Key Laboratory of Software Engineering of Yunnan, Yunnan 650091, China
[3]School of Economics, Yunnan University, Yunnan 650091, China

Corresponding author: Xuan Zhang (zhxuan@ynu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61862063, Grant 61502413, and Grant 61262025, in part by the National Social Science Foundation of China under Grant 18BJL104, in part by the Natural Science Foundation of Yunnan Province under Grant 2016FB106, in part by the Natural Science Foundation of Key Laboratory of Software Engineering of Yunnan Province under Grant 2015SE202, and in part by the Data Driven Software Engineering Innovative Research Team Funding of Yunnan Province under Grant 2017HC012.

ABSTRACT The need for particular software non-functional requirements (NFRs) leads a project team to use specific design strategies to fulfill these NFRs. However, some of the strategies may cause deficiencies in other software NFRs. These strategies are called conflicting strategies. Making the tradeoff decision of these conflicting strategies for NFRs is crucial. Therefore, a conflicting NFRs tradeoff framework (CNTF) and the corresponding method are proposed. Firstly, the NFRs are obtained from stakeholders. Fuzzy set theory is used to express stakeholder assessments on the importance of each NFR. The assessment results are ranked using a fuzzy ranking method. Then, adapting from production theory in microeconomics, the degree of satisfaction by which NFRs are affected by conflicting strategies are plotted as tradeoff curves. Borrowing the concept from the linear programming in management science, a tradeoff method is presented to help software project teams make the best tradeoff decision for conflicting NFRs. Finally, feasibility and limitations of the CNTF are elaborated in a case study.

INDEX TERMS Non-functional requirement, conflict, production theory, linear programming, tradeoff.

I. INTRODUCTION

Any software organization needs to consider non-functional requirements (NFRs) in order to deliver a system that complies with its stakeholder expectations [1]. Practical impact of NFRs has not only been acknowledged by the research community but also has been documented in many studies conducted in industry [1]–[3]. Neglecting NFRs during software development is a top-ten risk, and errors in considering them are the most expensive and difficult to correct [4], [5]. However, NFRs are often presented chaotically without a common standard and without sufficient analysis [6]. Moreover, the satisfactory degree of one NFR may be adversely affected by other NFRs. Software engineers and scholars have realized that balancing the NFRs among stakeholders, analyzing the conflicting relations among NFRs and finding their tradeoff are important for the quality of software and affect the success or failure of software.

The associate editor coordinating the review of this manuscript and approving it for publication was Resul Das.

To solve the conflicts of NFRs, Boehm provided a formal framework theory W for ''Make everyone winners'' [7], [8]. Based on the win conditions captured by the WinWin software [7], [8], Boehm and In developed quality attribute risk and conflict consultant (QARCC) to identify quality requirement conflicts [9], [10] and software cost option strategy tool (S-COST) to resolve cost-oriented NFRs conflicts [9], [10]. Almost at the same time, goal-oriented requirement engineering was developed for modeling and analyzing software requirements. For NFRs, NFR framework [11]–[14], Techne [15], and softgoal interdependency graph (SIG) [4], [12]–[14] are the basic methods and languages of goal-oriented NFRs engineering. Subsequently, more goal-oriented approaches and frameworks [16]–[19] have been formulated. The focus of this research has been mostly on modeling and reasoning, which aims to assist software engineers in analyzing NFRs relations qualitatively. These methods describe the interdependencies, e.g., ''BREAK, HURT, UNKNOWN, HELP, MAKE'' with some legends such as ''--, −, ?, +, ++'', and describe the degree of satisfaction with "×, √" [13], [20]. In our previous work [21], we also used goal-oriented modeling and reasoning for NFRs. Our proposed method was applied to a study of security infrastructure system (SIS) software and backward reasoning was used to find the conflicting NFRs. Generally speaking, goal-oriented modeling and reasoning mainly emphasize the importance of qualitative analysis [20]. The qualitative reasoning only helps to find the conflicting NFRs. When making tradeoff decision for the conflicting NFRs, the quantitative reasoning or evaluation becomes new trends in dealing with the tradeoff.

Tradeoffs are important in engineering. A tradeoff is a situational decision that involves diminishing or losing one quality, quantity or property of a design in return for gains in other aspects. The concept of a tradeoff suggests a decision made with full comprehension of the advantages and disadvantages of each choice. Based on our previous work [21], the conflicting NFRs were found by using goal-oriented modeling and reasoning. This paper proposes an efficient tradeoff framework called conflicting NFRs tradeoff framework (CNTF) to make tradeoff decision for these conflicting NFRs. The major contributions of this paper are the following:

(1) For the characteristics of NFRs that cannot be defined precisely, we used fuzzy set theory for stakeholders to assess the importance of each NFR. Trapezoidal fuzzy numbers were used to collect the stakeholder fuzzy assessments and a fuzzy ranking method was used to rank these fuzzy assessment results.

(2) A conflicting NFRs tradeoff framework (CNTF) was proposed to compensate for the reasoning in goal-oriented requirements engineering. Quantitative measurements of the strategies for the conflicting NFRs were used for the tradeoff. The production theory in microeconomics and linear programming in management science were adapted to provide a visualization method to analyze the tradeoff for the conflicting NFRs.

The remainder of this paper is structured as follows. In section II, a motivation scenario along with our previous work is presented. Section III details the framework and workflow of CNTF. A case study followed by our previous work [21] and the discussions of the method are presented in Section IV. Section V describes the related work and compares with our work. Section VI concludes our work and presents the future extension of our study.

II. MOTIVATION SCENARIO

The conflicts between NFRs are essentially caused by the conflicting design strategies. Table 1 shows some examples of the conflicting NFRs.

In Table 1, strategy "Store traceable information" causes the conflict between traceability and performance. Similarly, strategy "Encryption" causes the conflict between security and performance of speed. Furthermore, when we choose the specific design actions for these strategies, different design options can be used. For instance, "Store traceable information" for traceability includes options such as: "Store all states", "Store last state", "Store all names", "Store time", and "Store last change" et al. [17]. "Encryption" for security includes a set of different cryptographic algorithm options, such as data encryption standard (DES), Advanced Encryption Standard (AES), Rivest-Shamir-Adleman (RSA), and Elliptic curve cryptography (ECC) et al. The different options in a strategy are alternatives that create different level of NFRs satisfaction, and also cause different conflicting degree to the conflicting NFRs. Therefore, the tradeoff for these conflicting NFRs is transferred to make tradeoff decision for the corresponding conflicting strategies. In the paper, a design strategy describes how a NFR will be fulfilled by a set of design options. A design option is a specific design action that is taken to implement the strategy.

TABLE 1. Conflicting NFRs.

NFRs	Design strategies	Conflicting NFRs
Traceability	Store traceable information	Performance [17]
Security	Encryption	Performance (Speed) [16] [17]
	Authorized access	Usability[16], Performance (Speed) [4]
	Multifactor authentication	Usability [4]
Portability	Run on multiple operating systems	Maintainability [16]
Privacy	Manage cookies	Usability [17]
	Use smart cards	Usability [17]
	Use anonymous methods	Security [17]
Performance (Speed)	Use uncompressed format	Performance (Space) [4]
Extensibility	Ability to create new components	Maintainability [16]
	Runtime reconfiguration	Performance (Speed) [16]
Reliability	Data redundancy	Interoperability [22]
	Hardware redundancy	Cost [22]

In our previous work [21], we proposed an approach to modeling trustworthiness requirement-oriented software process. In our definition, NFRs of trustworthy software are divided into trustworthiness requirements and quality requirements. Trustworthiness requirements are part of the NFRs and are circumstance-dependent requirements that satisfy the stakeholders set of trustworthiness expectations. To satisfy multiple trustworthiness requirements, goal-oriented modeling and reasoning methods were used in a case study of security infrastructure system (SIS) software. SIS is a trustworthy third-party certification authority software system. It provides identity authentication services and secure connections over the Internet. SIS certification authority (SISCA) and SIS user agent (SISUA) are two subsystems of SIS. SISCA is a server system that manages users, keys, certificates, and cross authorization. SISUA is a client system that helps users encrypt, decrypt, sign, and verify. Because the SIS software had been running for years, new evolution requirements have been proposed continually.

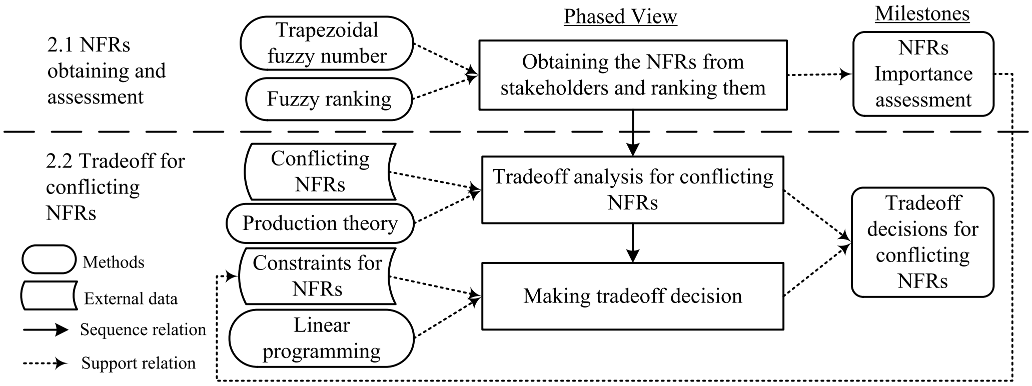


FIGURE 1. Conflicting NFRs tradeoff framework (CNTF).

For the aim of the evolution of the SIS software, a project team was formed involving the corresponding stakeholders. The original functional requirements and NFRs were summarized from the software requirement specifications. New functional requirements and NFRs were acquired from the project team. To satisfy the new NFRs, goal-oriented NFRs modeling and reasoning were used. In the process of modeling and reasoning, the design strategies were discussed in the first four meetings and continued for many times during the evolution development. Only 44 strategies were chosen because too many additional strategies may extend the duration of the project and cause complexity. A modified propositional satisfiability (SAT) solver was applied in the reasoning. The results of the reasoning found five conflicting strategies. One of these conflicting strategies was removed and compensated by the other two strategies which did not cause NFR conflicts. The other four conflicting strategies are listed in Table 2.

TABLE 2. Conflicting NFRs in [21].

NFRs	Design strategies	Conflicting NFRs
Compatibility	Interface design	Maintainability, Security
Reliability	Redundancy design	Maintainability, Security, Performance
Security	Define minimum cryptographic design	Performance, Usability
	Define minimum security criteria	Performance, Usability

As mentioned above in section I, goal-oriented modeling and reasoning are mainly used in qualitative analysis. When the quantitative analysis was used in goal models to manage tradeoffs, the experts' or users' subjective opinions were the main source of quantitative values. When we used prototyping in making design decision in our previous work [21], we obtained some objective measurement values from prototyping and we thought they could be used to make tradeoff decision. All design strategies in Table 2 can be measured in the prototype and the objective measurement values can be used in the tradeoff. Therefore, the following research focus is to provide a tradeoff method for these conflicting strategies. The advantage of our method is to fill the gap between NFRs elicitation and NFRs implementation.

III. CNTF FRAMEWORK

To find the best tradeoff for the conflicting NFRs, in this paper, we propose a conflicting NFRs tradeoff framework (CNTF), as shown in Figure 1.

There are two phases in the CNTF.

(1) NFRs obtaining and assessment

The stakeholders are the providers and evaluators of the requirements. They have a deep effect on the quality of software. Recent studies by Mckinsey [23] indicated that the top factors challenging project success are the lack of stakeholder involvement and poor convergence performance [24]. Therefore, in the first phase of the CNTF, we try to involve the stakeholders in the process. We ask them to indicate their assessments about NFRs and use their assessment results for the following tradeoff. The details are described in the following Section A.

(2) Tradeoff for conflicting NFRs

In the second phase, borrowing from the concept of production theory in microeconomics, a quantitative tradeoff curve for conflicting NFRs is drawn. Then, the tradeoff decision is made by adopting the method from linear programming in management science. A detailed description is provided in the following Section B.

A. NFRS OBTAINING AND ASSESSMENT

As NFRs specify how well software should perform its functions, imprecisely defined values of NFRs are better defined in fuzzy numbers [25]. In the following, trapezoidal fuzzy numbers (TrFNs) in the fuzzy set theory are used to collect stakeholder fuzzy assessment of their opinions on the importance of the NFRs, and a fuzzy ranking method is used to rank these fuzzy assessment results. The basic concepts of fuzzy set theory are introduced in Appendix A.

When the stakeholders are asked to express their individual opinions on the importance of each NFR, linguistic variables such as "high" and "low" are better to be used. We adapted from Zhu's linguistic variables and corresponding TrFNs [20]

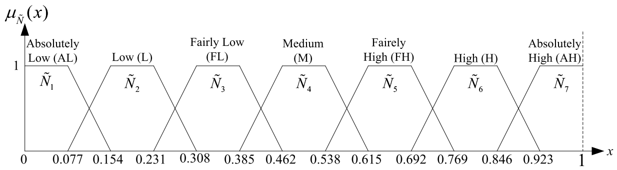


FIGURE 2. Linguistic variables and TrFns.

to collect the stakeholders fuzzy assessment. Figure 2 shows the linguistic variables set and the corresponding TrFNs.

In Figure 2, a linguistic variable can be represented as a fuzzy number, for example, "Absolutely High" can be represented as (0.846, 0.923, 1, 1). The same span length for the linguistic variables $\tilde{N}_2$ (Low), ..., $\tilde{N}_6$ (High) in Figure 2 signifies the same probability for stakeholders to choose. The linguistic variables $\tilde{N}_1$ (Absolutely Low) and $\tilde{N}_7$ (Absolutely High) have a smaller interval length because they commonly reflect two special extreme cases.

Based on the stakeholders fuzzy assessment, Chen and Sanguansat's fuzzy ranking method [27] is used to rank these fuzzy assessment results. The ranking result shows the degree of importance for each NFR and will be used in the tradeoff for conflicting NFRs. Chen and Sanguansat's fuzzy ranking method overcomes the drawbacks of some existing methods for ranking fuzzy numbers, i.e., capable of dealing with crisp values, is able to distinguish fuzzy numbers with different signs, is not pessimistic, etc.

Assume that there are k stakeholders and g NFRs to be ranked, where $\tilde{N}_{ij} = (n_{i1}, n_{i2}, n_{i3}, n_{i4})$ is the fuzzy assessment of each NFR$_i$ $(1 \leq i \leq g)$ that is assigned by stakeholder $_j(1 \leq j \leq k)$. First compute the average value $\tilde{N}_i$ from $\tilde{N}_{ij}$ $(1 \leq i \leq g)$ for each NFR$_i$ $(1 \leq i \leq g)$ in Eqs. (1):

$$\tilde{N}_i = (n_{i1}, n_{i2}, n_{i3}, n_{i4}) = \left(\frac{1}{k}\right) \otimes (\tilde{N}_{i1} \oplus \tilde{N}_{i2} \oplus \ldots \oplus \tilde{N}_{ik}) = \frac{1}{k}\sum_{j=1}^{k} \tilde{N}_{ij} \quad (1)$$

Then based on Chen and Sanguansat's fuzzy ranking method [27], calculating the areas $Area_{iL}^-$, $Area_{iR}^-$, $Area_{iL}^+$, and $Area_{iL}^+$ on the negative side and the positive side in Eqs. (4), (5), (6), and (7), respectively. They denote the trapezoidal areas from the membership function curves of $f_{N_i}^L$ and $f_{N_i}^R$, which are defined in Eqs. (2) and (3).

$$f_{N_i}^L = \frac{(x - n_{i1})}{(n_{i2} - n_{i1})}, \quad \mathrm{n}_{i1} \leq x < n_{i2} \quad (2)$$

$$f_{N_i}^R = \frac{(x - n_{i4})}{(n_{i3} - n_{i4})}, \quad n_{i3} \leq x < n_{i4} \quad (3)$$

$$Area_{iL}^- = \frac{(n_{i1} + 1) + (n_{i2} + 1)}{2} \quad (4)$$

$$Area_{iR}^- = \frac{(n_{i3} + 1) + (n_{i4} + 1)}{2} \quad (5)$$

$$Area_{iL}^+ = \frac{(1 - n_{i1}) + (1 - n_{i2})}{2} \quad (6)$$

$$Area_{iR}^+ = \frac{(1 - n_{i3}) + (1 - n_{i4})}{2} \quad (7)$$

To calculate the ranking score of each NFR$_i$ $(1 \leq i \leq g)$, the positive and negative influence values on the ranking score are calculated in Eqs. (8) and (9). The ranking score of $\tilde{N}_i$ is defined in Eqs. (10)

$$XI_{\tilde{N}_i} = Area_{iL}^- + Area_{iR}^- \quad (8)$$

$$XD_{\tilde{N}_i} = Area_{iL}^+ + Area_{iR}^+ \quad (9)$$

$$RN_i = Score(\tilde{N}_i) = \frac{XI_{\tilde{N}_i} - XD_{\tilde{N}_i}}{XI_{\tilde{N}_i} + XD_{\tilde{N}_i}} \quad (10)$$

RN_i is the final ranking value for the NFR$_i$ $(1 \leq i \leq g)$. The larger the ranking score of RN_i, the more important the corresponding NFR is. These importance ranking values will be used to help make tradeoff decision below. In the following, the conflicting NFRs are analyzed and a method to selecting best tradeoff in conflicting NFRs is presented.

B. TRADEOFF FOR CONFLICTING NFRS

Our tradeoff method is based on the production theory in microeconomics and the linear programming in management science. Basic concepts of production theory and linear programming are presented in Appendix B.

In this paper, a conflicting strategy tradeoff is represented as an isoquant and the tradeoff factors are the NFRs that are affected by the conflicting strategy. The importance ranks and baselines of the conflicting NFRs are constraints. When using the conflicting strategy to increase one NFR but decrease another NFR, plotting the quantity relations of these increasing and decreasing NFRs gives a corresponding tradeoff curve. A NFRs importance ranking constraint is represented by the ray from the origin with slope equaling to the ratio of conflicting NFRs importance ranking values *RNs* (See Eqs. (10)). A baseline constraint is the lowest or highest satisfaction level of each NFR. This baseline is determined by the stakeholders or the results from prototyping. The two constraints are graphed as constraint lines. By using NFRs baselines to constrain the tradeoff curve, we get an area of acceptable options and in this area, the intersection point of the NFR importance ranking constraint line and the tradeoff curve is the best option of the conflicting strategy.

1) A TRADEOFF CURVE FOR A CONFLICTING STRATEGY

In order to describe the tradeoff for a conflicting strategy, isoquant in production theory is used to represent the tradeoff curve and the affected NFRs are the tradeoff factors.

As shown in Figure 3, NFR$_1$ and NFR$_2$ are two NFRs and their degrees of satisfaction are affected by a conflicting strategy S. Four alternative options O_1, O_2, O_3, and O_4 can be chosen in S. By joining these points of options, we get a tradeoff curve. This curve is piecewise linear with three pieces. Every option O_m $(1 \leq m \leq t)$ on this tradeoff curve describes an elasticity of substitution between NFR$_1$ and NFR$_2$. Suppose that NFR_{im} is a set of measurement values that option O_m is measured in NFR$_i$, and nfr_{im} is one measurement value in NFR_{im}. In Figure 3, for each O_m $(1 \leq m \leq 4)$, NFR_{1i} and NFR_{2i} are two sets of the measurement values

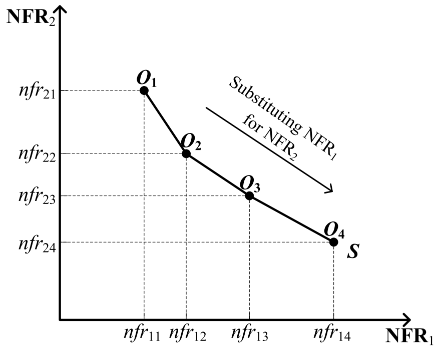


FIGURE 3. Tradeoff curve for strategy S.

of O_m in NFR$_1$ and NFR$_2$, respectively. If we choose O_4, we get the highest measurement value of NFR$_1$ which is nfr_{14}, but the lowest value of NFR$_2$ which is nfr_{24}. If we choose O_3, the corresponding measurement value of NFR$_2$ increases from nfr_{24} to nfr_{23}, while NFR$_1$ will decrease from nfr_{14} to nfr_{13}. This shows the conflicting relationships between NFR$_1$ and NFR$_2$. Therefore, making tradeoff decision on the conflicting NFR$_1$ and NFR$_2$ is essentially making tradeoff decision on choosing from options O_1, O_2, O_3, or O_4.

Since that the units of the measurements for the NFRs are different and the NFRs importance ranking values *RNs* are unitless. When a tradeoff curve is graphed, the values for the options must be normalized. According to the need to show the relationships between the increasing and decreasing values, Min-Max scaling is used and the scale is in a fixed range of 1 to t, where t is the number of options. The equation for this normalization is in (11).

$$nfr'_{im} = 1 + (nfr_{im} - nfr_{im-\min})(t-1)/(nfr_{im-\max} - nfr_{im-\min}) \quad (11)$$

In equation (11), nfr_{im} is an original measurement value that an option O_m $(1 \leq m \leq t)$ is measured in NFR$_i$, while nfr_{im}' is the normalized value, where $nfr_{im-\min}$ and $nfr_{im-\max}$ are the corresponding minimum and maximum values that the option O_m is measured in NFR$_i$.

The next step is to evaluate how much the decreasing of NFR$_1$ we choose to sacrifice for the increasing of NFR$_2$, or how much of the decreasing of NFR$_2$ we choose to sacrifice for the increasing of NFR$_1$.

2) CHOOSING BEST OPTION FROM A CONFLICTING STRATEGY

To answer the question of how to choose a best option from a conflicting strategy, referring to linear programming, Figure 4 illustrates a visualized tradeoff for alternative options O_1, O_2, O_3, O_4 in strategy S.

By plotting all the option points O_m $(1 \leq m \leq t)$ we get a tradeoff curve as shown in Figure 4. If the baseline constraint for each NFR can be collected from the stakeholders, we can graph a baseline constraint line, such as line $NFR_{2-\min}$ for NFR$_2$ or $NFR_{1-\max}$ for NFR$_1$ in Figure 4. For instance, "Refresh screen in less than 2 seconds" is one baseline

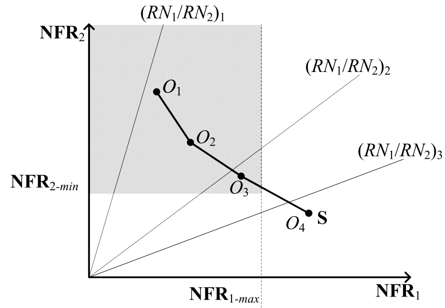


FIGURE 4. Choosing options from conflicting strategy *S*.

constraint of refresh speed. For a performance requirement, 2 seconds is a maximum baseline constraint. Any option that causes the refresh speed more than 2 seconds is not acceptable. Therefore, the shaded rectangle is an acceptable option region. Only the options in this region can be accepted. The acceptable option region constrains the acceptable options, but which option is the best one is still a problem. In Figure 4, O_1, O_2 and O_3 are three acceptable options. To determine the best one from them, we plot the NFRs importance ranking constraint line and the option that is nearest to this constraint line is the best. That is, the best option is the one that is acceptable first and then determined by the NFRs importance ranks.

In Figure 4, the slope of the NFRs importance ranking constraint line is RN_1/RN_2, where RN_i is calculated in the preceding Eqs. (10). If the importance ranking constraint line is $(RN_1/RN_2)_1$, the best option is O_1 because NFR$_2$ is more important than NFR$_1$ and the measurement value of option O_1 for NFR$_2$ is higher than NFR$_1$. Similarly, if the constraint line is $(RN_1/RN_2)_2$, the best option is O_3. If the constraint line is $(RN_1/RN_2)_3$, although O_4 is the nearest option, it cannot be selected because it is out of the acceptable options region. In this circumstance, the next nearest option O_3 is the best option.

In order to find the best option, linear programming is used. Suppose that NFR'_{uj} and NFR'_{vj} are two sets of normalized measurement values for option O_m $(1 \leq m \leq t)$ in conflicting strategy S and S is the conflicting strategy for NFR$_u$ and NFR$_v$. RN_u and RN_v are the importance ranking values of NFR$_u$ and NFR$_v$. The objective function and the constraints of linear programming are in Eqs. (12), (13), (14), and (15). The best option is the result of the objective function with the minimized value.

Objective function to be minimized is

$$OP = \sum_{m=1}^{t} |RN_u \cdot NFR'_{um} - RN_v \cdot NFR'_{vm}| \quad (12)$$

Baseline constraints for NFR$_u$:

$$nfr'_{um} \leq RN_{u-\max} \quad \text{or } nfr'_{um} \geq RN_{u-\min} \quad (13)$$

Baseline constraints for NFR$_v$:

$$nfr'_{vm} \leq RN_{v-\max} \quad \text{or } nfr'_{vm} \geq RN_{v-\min} \quad (14)$$

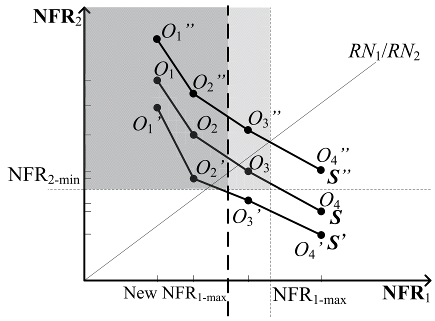


FIGURE 5. Strategy in changing environments.

Nonnegative constraints:

$$nfr_{um} \geq 0 \quad \text{and } nfr_{vm} \geq 0 \tag{15}$$

Since the best option is the one that is closest to the NFRs importance ranking lines and restricted to the acceptable option region, objective function (12) is a distance equation. Here we ignore the denominator $\sqrt{RN_u^2 + RN_v^2}$ in the equation because it is no use in our comparison.

In the preceding analyses, if the prototyping is used we have implicitly assumed a given technique, platform, cost, etc. However, over time, technological progress, platform upgrades, cost changes, and new management take place frequently. All these changes can be shown by a shift of the tradeoff curves or constraint lines. As shown in Figure 5, S' and S'' are two tradeoff curves that capture two different changes to the tradeoff curve of S. Moving from S to S' means the changes keep the same satisfaction level of NFR_1 but decrease the level of NFR_2. Moving from S to S'' means the changes keep the same satisfaction of NFR_1 but increase NFR_2. If the changes affect the baselines of NFRs, such as decreasing the cost of the project, the line of the Original $NFR_{1-\max}$ shifts to the left to a new $NFR_{1-\max}$. As illustrated in Figure 5, the movement of the NFR_1 baseline shrinks the acceptable options region. For example, if O_3 is the best option of strategy S before the baseline changes, after the change it is out of the acceptable option region and cannot be chosen to be the best option.

Algorithm 1 shows pseudo-code for the tradeoff for conflicting NFRs.

IV. CASE STUDY

The CNTF and the proposed method were applied to a case study which came from our previous work [21]. In our previous work, goal-oriented modeling and reasoning for NFRs are proposed and used in a systematic study of security infrastructure system (SIS) software. By borrowing concepts from goal-oriented requirements modeling, a NFRs model was designed. By using backward reasoning in the model, conflicting NFRs were found, but without providing a tradeoff solution for them. In the next sections, our aim is to analyze these conflicting NFRs and find the best options for them.

Algorithm 1 Tradeoff for Conflicting NFRs

Input:
$\tilde{N}_{ij} = (n_{i1}, n_{i2}, n_{i3}, n_{i4})$: fuzzy assessment of NFR_i that is assigned by $stakeholder_j$;
O: the set of strategy options;
nfr_{im}: measurement value that an option O_m ($O_m \in O$) is measured in NFR_i;
Output:
A best option;
Begin
For each NFR **do**
$\tilde{N}_i = (n_{i1}, n_{i2}, n_{i3}, n_{i4}) = \frac{1}{k}\sum_{j=1}^{k} \tilde{N}_{ij}$,
/* $\tilde{N}_i$ is the average value of $\tilde{N}_{ij}$ */
$RN_i = Score(\tilde{N}_i)$ /* RN_i is the ranking score of $\tilde{N}_i$ */
For each conflicting NFR_u and NFR_v **do**
For all O_m in O **do**
Normalize nfr_{um} to nfr_{um}',
Normalize nfr_{vm} to nfr_{vm}',
$op_m = |RN_u \cdot nfr'_{um} - RN_v \cdot nfr'_{vm}|$
Sort O into a list OP based on op_m values from small to large,
If $RN_{u-\max} = 0$ **and** $RN_{u-\min} = 0$ **and** $RN_{v-\max} = 0$ **and** $RN_{v-\min} = 0$ **then**
Return first option in OP
Else
For first option **to** last option **do**
If $(nfr'_{um} \leq RN_{u-\max}$ or $nfr'_{um} \geq RN_{u-\min})$ and $(nfr'_{vm} \leq RN_{v-\max}$ or $nfr'_{vm} \geq RN_{v-\min})$ **then**
Return current option in OP
Else move to the next option in OP
End

A. ASSESSING THE IMPORTANCE OF EACH NFR

As mentioned above, the stakeholder importance assessment on each NFR is one constraint for choosing the best option. Thus, with the aim of the SIS software evolution, six stakeholders were organized into a team. They were a software engineering technical manager T_1, a superintendent of the SIS T_2, a software developer T_3, an expert of Public Key Infrastructure T_4, a deputy of the certificate owner T_5 and a maintenance team T_6. Their importance assessments and the corresponding fuzzy ranking values (*RNs*) are listed in Table 3. Note that the NFRs in Table 3 are taken from the case study in our previous work [21].

B. TRADEOFF FOR THE CONFLICTING STRATEGIES

Based on the importance ranking values (RNs) of the NFRs in Table 3, the conflicting strategies which are listed in Table 2 (in section II) were analyzed one by one in the following.

1) INTERFACE DESIGN

One of the new evolution requirements was running SISUA on multiple operating systems. "Interface design ″ strategy

TABLE 3. NFRs importance assessment results and ranks.

NFRs	T_1	T_2	T_3	T_4	T_5	T_6	RNs
Maintainability	H	H	H	H	M	AH	0.7787
Security	H	AH	H	H	FH	AH	0.8268
Functional suitability	AH	H	AH	H	AH	AH	0.8973
Reliability	FH	H	M	H	M	H	0.6793
Usability	AL	L	AL	AL	L	L	0.1251
Compatibility	FH	FH	M	M	FH	FH	0.6023
Performance	M	L	L	L	M	FL	0.3207

was designed to fulfill this compatibility requirement. However, this strategy has a negative impact on the maintainability and security requirements. New interfaces designed for the new operating systems will make the maintenance work complicated. Moreover, the new interfaces will enlarge the attack surface of the software. Attack surface was proposed by Manadhata and Wing [30] and is used to measure the security of the software. It shows a set of ways in which an adversary can enter the software and potentially cause secure damage. The smaller the attack surface, the more secure the software. Therefore, the new method, channel, and data interfaces for compatibility must be designed carefully for the tradeoff among these conflicting NFRs. For the tradeoff, an interface test prototype was developed to test the necessary interfaces that could be added for the new system. In the prototype, different interface design options were tested. In order to control the potential intrusion through the new interfaces, attack surface measurements for all the options were calculated for predicting the security of these options. Borrowing from Manadhata and Wing [30], an attack surface is measured in the following equation:

$$AS = \sum_{e \in E} der_e(e) + \sum_{c \in C} der_c(c) + \sum_{d \in D} der_d(d) \qquad (16)$$

In Eqs. (16), $der_i(i) = count(i) \times q_i/e_i$ $(i = e, c, d)$, where e, c, d are method, channel and data interfaces which may be utilized for intrusion; $count(i)$ is the numbers of the interfaces; and q_i/e_i is the interfaces' damage potential-effort ratio. The damage potential-effort ratio is the benefit to an attacker in using an interface in an attack and the effort is the cost for the attacker in using the interface.

Based on the compatibility requirement, new interfaces were added for accessing files and devices, and monitoring files and inputs/outputs. According to the categories of attack surface interfaces, the new interfaces were network methods, file and device management methods, network channels, files, strings, and URL data. The optional interfaces and the number of these interfaces are described in Table 4.

The type of interfaces and the damage potential-effort ratios in Table 4 were borrowed from Manadhata and Wing's work [30]. The new interfaces and the number of these interfaces were obtained from the prototype. Based on these new interfaces, six different interface design options

TABLE 4. New interfaces for SIS software.

Type of interfaces	New interfaces	Count	q_i/e_i
Methods	Root method privilege with authenticated access right.	One to four entry point, one to two exit point	5/3
	Authenticated method privilege with authenticated access right.	Two to eight entry point, one to four exit point	3/3
Channels	SSL protocol with remote authenticated access right	One	1/1
	Socket protocol with local authenticated access right	One	1/4
Data	File data item with root access right	Three to six	1/5
	File data item with authenticated access right	Three to nine	1/3

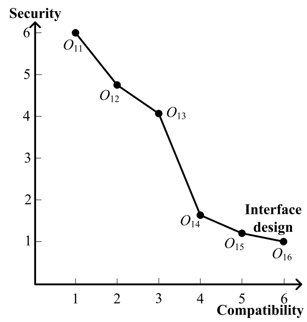


FIGURE 6. Tradeoff curve for "Interface design".

were designed. Table 5 lists these options. For each option, the number of interfaces is different for different interface design.

As mentioned above, different interface design has different attack surface measurement. Using the above equations (16) and (11), the attack surfaces of the six interface design options are calculated and normalized, and the results are listed in Table 5. Then, the interface design options are depicted in a tradeoff curve, which is shown in Figure 6.

In the tradeoff curve, the compatibility value for each option O_{1m} $(1 \leq m \leq 6)$ is the compatible level. There are six different compatible levels. The higher value of the level means higher compatibility. Correspondingly, the security values are the normalized attack surface values. Since the bigger value of the attack surface measurement means the lower security, we used the reversed minimum and maximum values for the normalization of the attack surface measurements. Thus, the higher the value of the normalized attack surface means higher security.

To find the best option for the tradeoff between security and compatibility, using the equations (12) to (15), baseline constraint and importance ranking constraint should be added. But, after the discussion within the project team, there were no baseline constraints for security and compatibility. All options were acceptable. According to the importance ranks in Table 3, the importance ranking values for security and compatibility are 0.8268 and 0.6023. The corresponding NFRs importance ranking constraint line is drawn in Figure 7.

TABLE 5. Six options for "Interface design".

Options	Details of the options	Compatibility level	Attack surface	Normalized attack surface	Values of objective function
O_{11}	One entry point and one exit point for root method with authenticated access right. Two entry point and one exit point for authenticated method with authenticated access right. Adding protocol SSL and socket. Three data items with root access right and three with authenticated access right.	1	9.18	6	11.5941
O_{12}	One entry point and one exit point for root method with authenticated access right. Four entry point and two exit point for authenticated method with authenticated access right. Adding protocol SSL and socket. Three data items with root access right and six with authenticated access right.	2	13.18	4.77	5.9736
O_{13}	Two entry point and one exit point for root method with authenticated access right. Four entry point and two exit point for authenticated method with authenticated access right. Adding protocol SSL and socket. Six data items with root access right and six with authenticated access right.	3	15.45	4.07	1.615
O_{14}	Three entry point and two exit point for root Method with authenticated access right. Six entry point and two exit point for authenticated method with authenticated access right. Adding protocol SSL and socket. Three data items with root access right and three with authenticated access right.	4	20.18	1.62	6.91
O_{15}	Three entry point and two exit point for root Method with authenticated access right. Six entry point and two exit point for authenticated Method with authenticated access right. Adding protocol SSL and socket. Six data items with root access right and six with authenticated access right.	5	21.78	1.13	10.769
O_{16}	Four entry point and two exit point for root Method with authenticated access right. Eight entry point and two exit point for authenticated method with authenticated access right. Adding protocol SSL and socket. Six data items with root access right and nine with authenticated access right.	6	25.45	1	13.7704

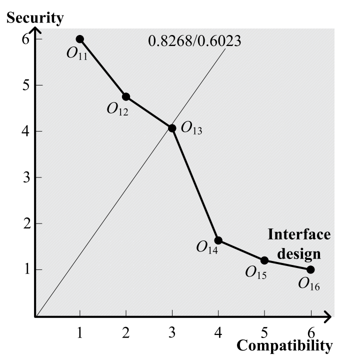


FIGURE 7. Constraints for the tradeoff curve.

Based on the tradeoff curve and the importance ranking constraint line, the best option is O_{13}. To validate this conclusion, using equations (12) to (15), the value of the objective function for each option is calculated and listed in the last column of Table 5. We see that the value of O_{13} is the minimized value. Thus, the option O_{13} is the best option in "Interface design" strategy. We also note that the decrease of security from O_{13} to O_{14} in Figure 7 is sharp. Through analysis, this sharp decrease is because the adding interfaces in C_{14} are method interfaces. The damage potential-effort ratio of method interface was higher than the other two types of interfaces. Thus, the project team was asked to pay more attention when adding new method interfaces.

Then, within the project team, we discussed the conflict of "Interface design" to the maintainability. Adding more interfaces could make maintenance work harder, but there is no appropriate way for us to quantify its degree. Therefore, we decided to adopt the option O_{13}. But the maintenance data during the running of the SIS software will be collected in the future for the continuous analysis.

2) REDUNDANCY DESIGN

"Redundancy design" strategy adds duplication of hardware and data to increase reliability. The redundant resources are prepared for backup and critical component failure. The running SIS software already had redundancy for data backup. But, when the new version of the SIS software is running for the test, the old version software should be running together until it has been replaced. This multi-version of the SIS software needs redundancy support. For the redundancy design, the new interfaces for the future multi-version running have been analyzed in the "Interface design" strategy. Since the importance ranks of compatibility and reliability are 0.6023 and 0.6793. They are very close. Therefore, based on the above analysis of "Interface design", we did not analyze the conflict of "Redundancy design" between maintainability and security.

For the conflict of "Redundancy design" to the performance, we tested its impact in the prototype. Although the performance of speed was reduced and the performance of space was enlarged, it is acceptable because the SIS software does not need real-time response and the enlarged space is acceptable.

3) DEFINE MINIMUM CRYPTOGRAPHIC DESIGN

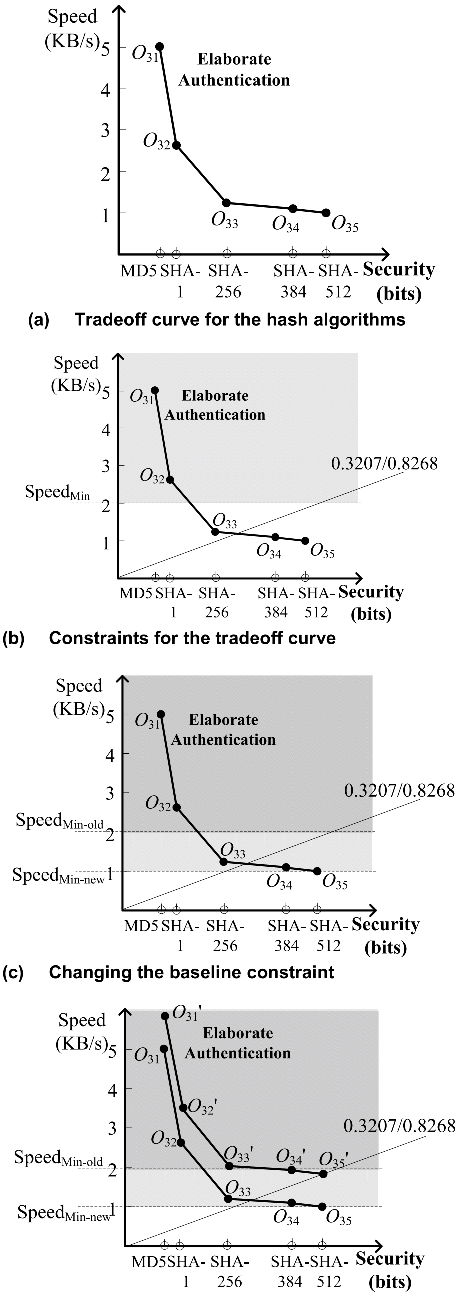
For the evolution of the SIS software, "Define minimum cryptographic design" strategy was used to strengthen the security of the software such as authentication, authorization,

TABLE 6. Time performances and security measurements of the hash algorithms.

Options	Hash Algorithm	Processing Time (KB/s)	Normalization of processing time	Security (bits)	Normalization of security	Values of objective function
O_{31}	MD5	412	5	<64	1	3.8133
O_{32}	SHA-1	293	2.63	<80	1.4	1.7255
O_{33}	SHA-256	224	1.26	128	2.6	null
O_{34}	SHA-384	219	1.16	192	4.2	null
O_{35}	SHA-512	211	1	224	5	null

and integration et al. For instance, the original hash algorithm for authentication was message-digest 5 (MD5). Now, more secure hash algorithms had been proposed. But, different hash algorithms have different security measurements and different processing times. The security measurements of hash algorithms are based on collision attack and could be measured in bits, such as 64 bits, 128 bits or 256 bits, where more bits means higher security. The processing times for these algorithms could be tested in the prototype. Thus, the conflict of this strategy between security and performance was analyzed first. Table 6 shows the security measurements and the processing times of the hash algorithms.

For the better security of authentication, a higher secure algorithm should be chosen but the corresponding processing time is a conflict. A higher secure hash algorithm usually needs more processing time. Based on the values of the security measurements and the processing times in Table 6, a tradeoff curve is drawn in Figure 8 (a). Similarly, to choose the best option from O_{3m} ($1 \leq m \leq 5$), the baseline constraint and importance ranking constraint are added, as shown in Figure 8 (b). The importance ranking constraint was taken from Table 3. The baseline constraint was the discussion result of the project team. From Figure 8 (b), the best option may be O_{33}, but it is out of the acceptable option region. To choose the best option, the values of the objective function for all the options were calculated and listed in the rightmost column in Table 6. O_{32} has the minimum value in the objective function. O_{33}, O_{34}, and O_{35} do not have values of objective function because they all do not satisfy the baseline constraint. However, since this baseline constraint was discussed before we used a prototype to make tradeoff decision, it was proposed to discuss again within the project team. Based on Figure 8 (b), if we reduce the baseline of speed from 2KB/s to 1KB/s, all options will be available. Also, when running all the secure hash algorithm (SHA) algorithms in the prototype, their different speeds did have a negative impact since the input data for the hash algorithms were relatively small. Thus, we added a new baseline constraint $\text{Speed}_{min-new}$ in Figure 8 (c) and calculated their values of objective function again. The values for O_{33}, O_{34}, and O_{35} are 0.208, 0.3879, and 0.7767 respectively. In the new acceptable option region, O_{33} is the best option. Furthermore,



(a) Tradeoff curve for the hash algorithms

(b) Constraints for the tradeoff curve

(c) Changing the baseline constraint

(d) Changing the running platform for the prototype

FIGURE 8. Tradeoff for "Define minimum cryptographic design" strategy.

when we analyzed their values of the objective function, although the option O_{33} has the minimum value it is just a little bit less than options O_{34} and O_{35}. Thus, we installed our prototype to a higher performance computer and tested again. All options had faster processing times and a new tradeoff curve is drawn in Figure 8 (d). Based on the same constraints, O_{35}' (SHA-512) become the new best option. For the higher security of the SIS software, the project team decided to choose SHA-512.

Similarly, all the other cryptographic algorithms were analyzed and discussed. Corresponding decisions had been made.

For the conflict of ‘‘Define minimum cryptographic design’’ between security and usability, the project team also analyzed it in the prototype. Since all the cryptographic algorithms were designed and tested in the prototype, the feedbacks from the users showed that the negative impact of this strategy could be ignored. Therefore, after the discussion, the project team decided to continually use the prototype to understand users' perceptions of how well the software works and based on their feedback to improve the usability of the software in the future.

4) DEFINE MINIMUM SECURITY CRITERIA

‘‘Define minimum security criteria’’ strategy sets up multi-layer defense devices and software, such as routers, firewalls, and Intrusion Detection Software (IDSs) et al., to protect the SIS software. It strengthens the security of the software but has the same negative impacts on performance and usability. Using the same tradeoff for ‘‘Define minimum cryptographic design’’ strategy, the tradeoff decisions were provided for finding appropriate defense design.

C. DISCUSSIONS

Throughout the tradeoff analysis process, we discussed our methods and the results within the project team. Overall, as compensation for the previous work [21], making tradeoff decision for the conflicting NFRs is valuable in assisting human judgment in conflicting NFRs implementation, especially for the software that is needed to satisfy multiple NFRs. From the visualizing the tradeoff results, many meaningful lessons had been learned. Firstly, the project team could better understand their selections of the options. Also, the tradeoff process could be visually adjusted, which helped the project team to choose the best options. Secondly, several potential risks which were caused by the conflicting NFRs could be identified so that they could be monitored, reduced or mitigated.

Although we have achieved a positive result from this case, we still find a limitation to be solved. Our case study is only one specific case in many cases. In this case, each solution decision was made in isolation to the others. However, the conflicting NFRs were inter-related. For example, all four strategies in this case were related to security. If we chose more secure cryptographic algorithms in the ‘‘Define minimum cryptographic design’’ strategy, should we add more interfaces or redundancies for less security but better compatibility and reliability in ‘‘Interface design’’ and ‘‘Redundancy design’’ strategies? The answer in this case is no because more secure cryptographic algorithms only making it difficult for attacks to crack the encrypted or hashed data. But, if more interfaces or redundancies are designed, the attackers could choose to use these interfaces and redundancies to attack the software because they are not protected by the cryptographic algorithms. Comparing to the compatibility and reliability requirements in SIS software, the security requirement is more important, as shown in Table 3. Therefore, the chosen options in these strategies were relatively isolated. But, high security cryptographic design and defense design will affect the response time of the SIS software. The response time we tested in the prototype may be too ideal and it may need to be adjusted in the real software. Therefore, facing more complicated cases, we will study how to combine multi-objective optimization methods with our framework and conduct further validation in more realistic cases.

D. THREATS TO VALIDITY

In this section, we describe the different threats that could affect the validity of the case study. The discussed threats affect the construct, internal, external and conclusion validity of the results.

1) CONSTRUCT VALIDITY

Construct validity is affected by the choice of a single method, fuzzy assessment, to obtain the NFRs importance ranks. Other techniques, such as AHP, can also be used to prioritize the NFRs. These other techniques will be tested in future work.

2) INTERNAL VALIDITY

Internal validity is threatened by the selection of our case. Some of the NFRs of our case are clearly more important, such as security requirement. For some software projects, there may be no significant difference in the importance of their NFRs. Our method will have limitations for such cases because our tradeoff is determined by the NFRs importance ranks. However, testing more NFRs ranking techniques to mitigate the threat of construct validity may also help to mitigate this threat.

3) EXTERNAL VALIDITY

According to stakeholder different roles, responsibilities, and experiences, they may have different opinions on the importance of each NFR. In our previous work [21], we used the interactive and iterative Delphi method [31] to reconcile the trustworthiness requirements of multiple stakeholders and achieve consensus. In this work, for the stakeholder fuzzy assessment of the importance of each NFR, we also used the Delphi method to eliminate leaning effects of their different opinions. However, different stakeholder preferences still might be a threat to the validity of the NFRs assessment.

We ignored the impact of the prototype implementation techniques and the prototype developer's level of expertise. Obviously, the different techniques and levels of expertise could lead to different tradeoff results. Although we need different tradeoff analysis in a different project, inappropriate techniques or the lack of expertise still might be a threat.

4) CONCLUSION VALIDITY

Conclusion validity is threatened by the use of a single case. Despite the methods and tradeoff results being discussed in the project team, the findings are not conclusive. This is why we only claim that we assist human judgment in conflicting

NFRs implementation. We also showed that further validation in more realistic cases would be conducted.

V. RELATED WORK

We next offer a quick overview of and comparison with some related tradeoff methods. Based on the win conditions captured by the WinWin software [7], [8], Boehm and In developed quality attribute risk and conflict consultant (QARCC) to identify quality requirement conflicts [9], [10] and software cost option strategy tool (S-COST) to resolve cost-oriented NFRs conflicts [9], [10]. But, S-COST was designed to optimize only for a single objective (cost).

Marew *et al.* [32] proposed a quantified version of SIG, quantified SIG (Q-SIG). Instead of assigning qualitative descriptions, they quantified subjective measurements to analyze the tradeoff of the different tactics (the specific mechanisms used to fulfill NFRs). Zhu *et al.* [20] further proposed fuzzy qualitative and quantitative SIG (FQQSIG) model for NFRs correlations analysis. Some other research work also used quantitative measures. To illustrate the complexity of the design of an evaluation mechanism for a goal-oriented language, Amyot *et al.* [33] proposed quantitative, qualitative, and hybrid evaluation algorithm. Liaskos *et al.* Elicited quantitative priorities among preference goals in [34] and quantitative contribution measures in goal models in [35]. The quantitative values they used were all assigned by the experts' or users' subjective opinions or preferences. Our work also relies on stakeholders for assessing the importance of the various NFRs. But, compared to their work, we try to use prototyping results as objective measurement values to deal with the conflicting NFRs and help to make tradeoff decisions.

Nguyen *et al.* [36] proposed the constrained goal model (CGM) for multi-objective reasoning. The contribution of this work was being exploited in several directions. Angelopoulos *et al.* used qualitative information in Zanshin [37] and quantitative constraints along with one or several objective functions in Prometheus [38] to an Optimization Modulo Theories/Satisfiability Modulo Theories (OMT/SMT) solver. For the next release problem, Aydemir *et al.* [39] also used OMT reasoner in multi-objective quantitative optimization. The results showed that quantitative optimization performed better [38]. Similarly, we also gained an effective tradeoff decision when using quantitative information. The main bottleneck of their approach is deriving the quantitative impact on the NFRs. Complicated stakeholder preferences will make this quantitative impact difficult to derive. This quantitative impact may also have an inaccurate problem. But, using their multi-objective optimization methods to improve our method is part of our future work.

Adding quantitative analysis to deal with the tradeoff problem is a useful supplement in goal-oriented approaches. To add a quantitative method to evaluate and make tradeoff on NFRs, in this paper, we use the results from prototyping and borrow the techniques from production theory and linear programming to provide quantitative tradeoff decision. Using the objective quantitative values from prototyping can better assist human judgment in conflicting NFRs implementation.

VI. CONCLUSION

Dealing with NFRs is a major challenge in software development [1]. Analyzing conflicts in NFRs is a major task in large software systems development projects [6]. Throughout the requirement engineering process, when improving the quality of software, selecting effective strategies for every NFR is an essential activity in requirements engineering. It involves evaluating how well each strategy satisfies the stakeholders NFRs and how to make tradeoff decision in conflicting NFRs. Economics is the study of how people make decisions in resource-limited situations [40], [41]. Resource-limited situations mean that a tradeoff decision must be made. A tradeoff is where one thing increases and another must decrease. In this paper, production theory is used to make tradeoff decisions for conflicting NFRs. In the tradeoff, the quantitative analysis provides a more accurate comparison of alternative options. The discussions in Section IV proved progress in tradeoff analysis of conflicting NFRs; positive results are obtained in the case study, but the limitations and the threats to validity are also identified and our research is still evolving. Therefore, more work will be conducted in the future:

(1) In a real project, many strategies are limited by the cost. Our first further work is to use cost estimation model to analyze the quantitative cost relationships between strategy options, and provide cost-effective options.

(2) Further validation should be conducted, including more appropriate metrics and more accurate measurements. Also, our framework should be used in more variety of realistic cases. However, challenges in designing effective studies (realistic vs. easily measurable) is also a potential work item in the future.

APPENDIX A
BASIC CONCEPTS OF FUZZY SET THEORY

Fuzzy set theory was introduced by Zadeh [25] as an extension of the classical notion of set. Unlike the classical set in which a membership function can only take on precise values, the membership function in the fuzzy set represents the grade of membership and associates each point with a real number in the interval [0, 1]. In fuzzy set theory, various fuzzy numbers were defined. Trapezoidal fuzzy numbers (TrFNs) are often used in practice. A TrFN $\tilde{N}$ can be defined as (n_1, n_2, n_3, n_4) shown in Figure 9 which has membership function $\mu_{\tilde{N}}(x)$ in Eqs. (17).

$$\mu_{\tilde{N}}(x) = \begin{cases} \dfrac{x - n_1}{n_2 - n_1}, & n_1 < x \leqslant n_2 \\ 1, & n_2 < x \leqslant n_3 \\ \dfrac{x - n_4}{n_3 - n_4}, & n_3 < x \leqslant n_4 \\ 0, & \text{otherwise} \end{cases} \tag{17}$$

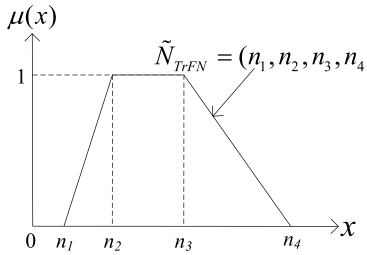


FIGURE 9. A TrFN $\tilde{N}$.

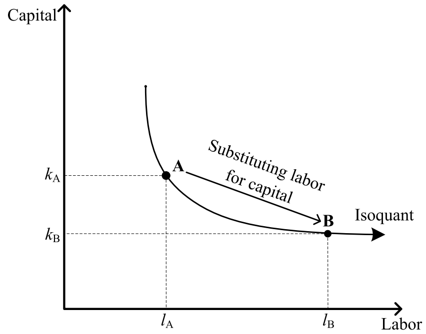


FIGURE 10. Isoquant curve.

where n_1, n_2, n_3, n_4 are real numbers and $n_1 \leq n_2 \leq n_3 \leq n_4$. The membership function of a TrFN is piecewise linear and trapezoidal which can capture the vagueness of those linguistic variables, such as "quite young" or "very important". Linguistic variables are variables whose values are words or sentences in a natural or artificial language [26]. Each linguistic variable can correspond to a fuzzy number.

APPENDIX B
BASIC CONCEPTS OF PRODUCITON THEORY AND LINEAR PROGRAMMING

In production theory [28], [29], when labor and capital are both variable inputs, an isoquant shows the various combinations of the two inputs that can be used to produce a specific level of output. In general, the elasticity of substitution measures the degree to which an input can be substituted for another input in production. In Figure 10, two combinations, A and B, show the elasticity of substitution.

Furthermore, it is very important for an organization to be able to choose the best inputs for their production. Therefore, minimizing the cost of producing a given level of output, the organization must produce at the point where an isocost line is tangent to the isoquant. The point of tangency is the least-cost input combination. In Figure 11, A is the cost-minimization point and also represents an optimal input substitution. However, the isocost line may not be the only constraint faced by an organization in trying to reach the highest isoquant. In the real world, organizations often face more than one constraint. One of the other constraints may be a production process with a technologically fixed input combination. This production process is represented in the input space by a straight line. As shown in Figure 11, $K_{\text{constraint}}$ and $L_{\text{constraint}}$ are two constraints for capital input and labor input. Given the constraints of these production processes, an area of acceptable inputs is provided to an organization. In Figure 11, point B is not acceptable because its labor input is beyond the labor constraint. Point A is acceptable and is the cost-minimization point and also represents an optimal input substitution.

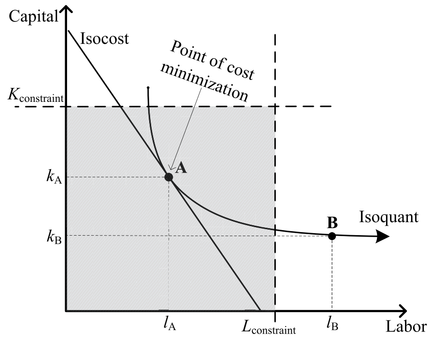


FIGURE 11. Isoquant curve and constraints.

Linear programming maximizes or minimizes a linear function subject to linear constraints. The preceding isocost constraint and production process constraints can be defined in linear programming. Taking the maximum problem as an example, in linear programming, it can be expressed in canonical form as

$$\begin{aligned} &\text{Maximize } c^T x \\ &\text{Subject to } Ax \leq b \\ &\text{And } x \geq 0 \end{aligned}$$

where x represents the vector of variables to be determined, c and b are vectors of coefficients, A is a matrix of coefficients, and $(\cdot)^T$ is the matrix transpose. The expression to be maximized or minimized is called the objective function ($c^T x$ in this case). The inequalities $Ax \leq b$ and $x \geq 0$ are the constraints which specify a convex polytope over which the object function is to be optimized.

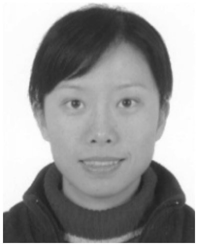
ACKNOWLEDGMENT

Special thanks to Dr. Josephine E. Olson at University of Pittsburgh for her great help, advice and endless patience in improving their writing.

REFERENCES

[1] D. Ameller, X. Burgués, D. Costal, C. Farré, and X. Franch, "Non-functional requirements in model-driven development of service-oriented architectures," *Sci. Comput. Program.*, vol. 168, pp. 18–37, Dec. 2018.
[2] T. Erl, *Service-Oriented Architecture: Concepts, Technology, and Design*. Upper Saddle River, NJ, USA: Prentice-Hall, 2005.
[3] D. Ameller, C. Ayala, J. Cabot, and X. Franch, "Non-functional requirements in architectural decision making," *IEEE Softw.*, vol. 30, no. 2, pp. 61–67, Mar./Apr. 2013.
[4] L. Chung, B. A. Nixon, E. Yu, and J. Mylopoulos, *Non-Functional Requirements in Software Engineering*. Norwell, MA, USA: Kluwer, 2000.
[5] B. Lawrence, K. Wiegers, and C. Ebert, "The top risk of requirements engineering," *IEEE Softw.*, vol. 18, no. 6, pp. 62–63, Nov./Dec. 2001.
[6] C.-L. Liu, "CDNFRE: Conflict detector in non-functional requirement evolution based on ontologies," *Comput. Standards Inter.*, vol. 47, pp. 62–76, Aug. 2016.

[7] B. Boehm, P. Bose, E. Horowitz, and M.-J. Lee, "Software requirements as negotiated win conditions," in *Proc. IEEE Int. Conf. Requirements Eng. (ICRE)*, vol. 4, Apr. 1994, pp. 74–83.
[8] B. Boehm, P. Bose, E. Horowitz, and M. J. Lee, "Software requirements negotiation and renegotiation aids: A theory-W based spiral approach," in *Proc. 17th Int. Conf. Softw. Eng. (ICSE)*, vol. 4, Apr. 1995, pp. 243–253.
[9] B. Boehm and H. In, "Identifying quality-requirement conflicts," *IEEE Softw.*, vol. 13, no. 2, pp. 25–35, Mar. 1996.
[10] H. In, "Conflict identification and resolution for software attribute requirements," Ph.D. dissertation, Dept. School Eng., Univ. Southern California, Los Angeles, CA, USA, 1998.
[11] J. Mylopoulos, L. Chung, and B. Nixon, "Representing and using nonfunctional requirements: A process-oriented approach," *IEEE Trans. Softw. Eng.*, vol. 18, no. 6, pp. 483–497, Jun. 1992.
[12] L. Chung and B. A. Nixon, "Dealing with non-functional requirements: Three experimental studies of a process-oriented approach," in *Proc. 17th Int. Conf. Softw. Eng. (ICSE)*. New York, NY, USA: ACM, 1995, p. 25.
[13] L. Chung, B. A. Nixon, E. Yu, and J. Mylopoulos, "Non-Functional requirements in software engineering," in *International Series in Software Engineering*, vol. 476. New York, NY, USA: Springer-Verlag, 1999.
[14] L. Chung and J. C. S. do Prado Leite, "On non-functional requirements in software engineering," in *Conceptual Modeling: Foundations and Applications*. Berlin, Germany: Springer-Verlag, 2009, pp. 363–379.
[15] I. J. Jureta, A. Borgida, N. A. Ernst, and J. Mylopoulos, "Techne: Towards a new generation of requirements modeling languages with goals, preferences, and inconsistency handling," in *Proc. 18th IEEE Int. Requirements Eng. Conf.*, vol. 11, Sydney, NSW, Australia, Sep./Oct. 2010, pp. 115–124.
[16] J. Cleland-Huang, W. Marrero, and B. Berenbach, "Goal-centric traceability: Using virtual plumblines to maintain critical systemic qualities," *IEEE Trans. Softw. Eng.*, vol. 34, no. 5, pp. 685–699, Sep. 2008.
[17] L. Cysneiros, E. Yu, and J. Leite, "Cataloguing non-functional requirements as softgoals networks," in *Proc. 11th IEEE Int. Requirements Eng. Conf.*, Monterey Bay, CA, USA, Sep. 2003, pp. 13–20.
[18] B. A. Nixon, "Management of performance requirements for information systems," *IEEE Trans. Softw. Eng.*, vol. 26, no. 12, pp. 1122–1146, Dec. 2000.
[19] E. A. Oladimeji, S. Supakkul, and L. Chung, "Security threat modeling and analysis: A goal-oriented approach," in *Proc. 10th IASTED Int. Conf. Softw. Eng. Appl.*, vol. 11, 2006, pp. 13–15.
[20] M.-X. Zhu, X.-X. Luo, X.-H. Chen, and D. D. Wu, "A non-functional requirements tradeoff model in trustworthy software," *Inf. Sci.*, vol. 191, pp. 61–75, May 2012.
[21] X. Zhang, X. Wang, and Y. Kang, "Trustworthiness requirement-oriented software process modeling," *J. Softw., Evol. Process*, vol. 30, no. 12, 2018, Art. no. e1991.
[22] B. Boehm and N. Kukreja, "An initial ontology for software qualities," in *Proc. 25th Int. Symp. Annu. (INCOSE)*. Seattle, DC, USA: Wiley, Oct. 2017, vol. 20, no. 3, pp. 18–28.
[23] Mckinsey & Company. (2012). *Delivering Large-Scale IT Projects on Time, on Budget, and on Value*. [Online]. Available: http://www.mckinsey.com/business-functions/business-technology/our-insights/delivering-large-scale-it-projects-on-time-on-budget-and-on-value,10
[24] D. Wu, "Wikiwinwin—A WIKI-based collaboration framework for rapid requirements negotiations," Ph.D. dissertation, Dept. School Eng., Univ. Southern California, Los Angeles, CA, USA, 2014.
[25] L. A. Zadeh, "Fuzzy sets," *Inf. Control*, vol. 8, no. 3, pp. 338–353, Jun. 1965.
[26] L. A. Zadeh, "The concept of a linguistic variable and its application to approximate reasoning—I," *Inf. Sci.*, vol. 8, no. 3, pp. 199–249, 1975.
[27] S.-M. Chen and K. Sanguansat, "Analyzing fuzzy risk based on a new fuzzy ranking method between generalized fuzzy numbers," *Expert Softw. Appl.*, vol. 38, pp. 2163–2171, Mar. 2011.
[28] D. Salvatore, *Microeconomics: Theory and Applications*. New York, NY, USA: Macmillan, 1986.
[29] W. Sher and R. Pinola, *Microeconomic Theory: A Synthesis of Classical Theory and the Modern Approach*. New York, NY, USA: Elsevier, 1981.
[30] P. K. Manadhata and J. M. Wing, "An attack surface metric," *IEEE Trans. Softw. Eng.*, vol. 37, no. 3, pp. 371–386, May 2011.
[31] N. Dalkey, O. Helmer, and O. Helmer, "An experimental application of the delphi method to the use of experts," *Manage. Sci.*, vol. 9, no. 3, pp. 458–467, Apr. 1963.
[32] T. Marew, J.-S. Lee, and D.-H. Bae, "Tactics based approach for integrating non-functional requirements in object-oriented analysis and design," *J. Softw. Softw.*, vol. 82, pp. 1642–1656, Oct. 2009.
[33] D. Amyot, S. Ghanavati, J. Horkoff, G. Mussbacher, L. Peyton, and E. Yu, "Evaluating goal models within the goal-oriented requirement language," *Int. J. Intell. Softw.*, vol. 25, no. 8, pp. 841–877, 2010.
[34] S. Liaskos, S. A. McIlraith, S. Sohrabi, and J. Mylopoulos, "Representing and reasoning about preferences in requirements engineering," in *Best Papers of Re'10: Requirements Engineering in a Multi-Faceted World*, vol. 16. London, U.K.: Springer-Verlag, 2011, pp. 227–249.
[35] S. Liaskos, J. Rina, and A. Jorge, "On eliciting contribution measures in goal models," in *Proc. 20th IEEE Int. Conf. Requirements Eng. (RE)*, Sep. 2012, pp. 221–230.
[36] C. M. Nguyen, R. Sebastiani, P. Giorgini, and J. Mylopoulos, "Multi-objective reasoning with constrained goal models," *Requirements Eng.*, vol. 23, pp. 189–225, Jun. 2018.
[37] K. Angelopoulos, V. E. S. Souza, and J. Mylopoulos, "Dealing with multiple failures in zanshin: A control-theoretic approach," in *Proc. 9th Int. Symp. Softw. Eng. Adapt. Self-Manag. Syst., (SEAMS)*, Hyderabad, India, Jun. 2014, pp. 165–174.
[38] K. Angelopoulos, F. B. Aydemir, P. Giorgini, and J. Mylopoulos, "Solving the next adaptation problem with prometheus," in *Proc. IEEE 10th Int. Conf. Res. Challenges Inf. Sci. (RCIS)*, Jun. 2016, pp. 1–10.
[39] F. B. Aydemir, F. Dalpiaz, P. Giorgini, and J. Mylopoulos, "The next release problem revisited: A new avenue for goal models," in *Proc. IEEE 26th Int. Requirements Eng. Conf. (RE)*, Aug. 2018, pp. 5–16.
[40] B. Boehm, *Software Engineering Economics*. Upper Saddle River, NJ, USA: Prentice-Hall, 1981.
[41] B. Boehm, "Software engineering economics," *IEEE Trans. Softw. Eng.*, vol. SE-10, no. 1, pp. 4–21, Jan. 1984.



XUAN ZHANG was born in Kunming, Yunnan, China, in 1978. She received the B.S. and M.S. degrees in computer science and the Ph.D. degree in system analysis and integration from Yunnan University, Yunnan, in 2000, 2003, and 2014, respectively.

From 2003 to 2012, she was an Assistant Professor with the School of Software, Yunnan University, where she has been an Associate Professor, since 2012. She is the author of two books and more than 60 articles. She has been principal investigator for more than ten national, provincial, and private grants and contracts. Her research interests include software process, trustworthy software, and requirement engineering.

XU WANG was born in Kunming, Yunnan, China, in 1976. He received the B.S., M.S., and Ph.D. degrees in economics from Yunnan University, Yunnan, in 1998, 2006, and 2010, respectively.

From 2010 to 2016, he was an Assistant Professor with the School of Economics, Yunnan University. Since 2016, he has been an Associate Professor with the School of Economics, Yunnan University. He is the author of three books and more than 50 articles. His research interests include business process management, monetary economics and banking, and financial security.