An Approach for Prioritizing NFRs According to Their Relationship with FRs

Mohammad Dabbagh and Sai Peck Lee

Abstract—Due to the budgetary deadlines and time to market constraints, it is essential to prioritize software requirements. The outcome of requirements prioritization is an ordering of requirements which need to be considered first during the software development process. To achieve a high quality software system, non-functional requirements (NFRs) must be taken into consideration throughout the prioritization process. Although several requirements prioritization methods have been proposed so far, the research on prioritizing non-functional requirements is limited. In this paper, we propose a quantitative approach based on Fuzzy Logic and Alpha cut approach which aims to perform the process of prioritizing non-functional requirements. Application of the proposed approach provides the practitioners with a prioritized list of non-functional requirements which are prioritized based on their relationship with functional requirements (FRs).

Index Terms—Requirements engineering, software requirements prioritization, non-functional software requirements.

I. INTRODUCTION

The ultimate goal of developing any software system is to satisfy various stakeholders’ needs [1]. Hence, managing the software requirements process plays a critical role towards the success of a software development project [2]. Requirements engineering, as a first step in the software development process, and its underlying activities can help practitioners to understand stakeholders’ needs and develop high quality software in an economic manner.

However, due to the budgetary deadlines and time to market constraints, it could be a challenge for requirements engineers to decide which requirements lead to high stakeholder satisfaction and need to be considered first. To address this concern and in order to reduce the cost and duration of a software project as well, it is essential to address the high-priority requirements before considering the low-priority ones [2]-[4]. Requirements prioritization can help to identify the most important requirements for a software system [5], and then proceed to develop the software according to these requirements. Hence, requirements prioritization has been recognized as one of the most important decision making processes during the software development process [1], [6], [7].

The requirements engineering community has classified the requirements of a software system into two main categories: functional requirements and non-functional requirements [8]. Functional requirements describe the functional behavior of the system whereas non-functional requirements express how good a system should work. It has been widely acknowledged that a quality attribute such as reliability, modifiability, performance, or usability is a non-functional requirement of a software system [8]-[10]. Considering merely functional requirements during requirements prioritization process may lead to failure in the final product or at least may result in a poor quality system. The achievement of quality attributes along with functional requirements is critical to the success of a software system [11]-[15].

Numerous methods on requirements prioritization have been introduced in recent years, the most widely known of which being Analytic Hierarchy Process (AHP) [16], Cost-Value approach [1], Wieger’s method [17], Value-Oriented requirements Prioritization (VOP) [18], and more recently there is a method which applied interactive genetic algorithm to prioritize requirements [19]. Although these methods have contributed a lot to software development process, they are not suitable for prioritizing non-functional requirements from the point that they have not addressed the prioritization of non-functional requirements according to their relationship with functional requirements. This motivation encouraged us for proposing an approach which aims to assist practitioners during the early phase of the software development process to decide which non-functional requirements are more significant and need to be considered first.

Our approach for prioritizing functional and non-functional requirements assists practitioners to obtain a prioritized list of non-functional requirements (NFRs) according to their relationship with functional requirements (FRs). According to the Berander and Andrews [20], non-functional requirements affect several functional requirements, from one to all FRs of a software system. By concentrating on this point, during this work, we try to find out the extent in which each NFR may affect a given FR. By determining such a value (i.e., the importance degree of an NFR for a given FR), our approach intends to prioritize NFRs. The framework produces a prioritized list of NFRs by calculating the total importance degree of each NFR with respect to all FRs using Triangular Fuzzy Number and Alpha cut approach. In other words, it means that an NFR which achieves the greatest total importance degree among all FRs, may be assigned as a high-priority NFR with respect to all FRs. In this paper, we have used the term non-functional requirements and quality attributes interchangeably to mean the same.

The remainder of this paper is organized as follows. Requirements prioritization methods from the literature are introduced in Section II. The proposed approach for
prioritizing non-functional requirements is illustrated in Section III. In section IV an illustrative example is described through which we explain how the four steps of our approach could be applied for a prioritization problem. The supporting tool is presented in Section V and finally, Section VI concludes and outlines future work.

II. RELATED WORK

As the complexity of software systems increases, practitioners are forced to make trade-offs between conflicting requirements in order to complete projects on predefined schedule. Priority assessment of requirements is one of the techniques which can be useful to assist practitioners to resolve trade-offs. Thus, requirements prioritization has become an increasingly important part of ensuring the success of a project, and consequently, various works pointed out the importance of the problem of requirements prioritization in the software engineering domain.

Even though prioritization techniques have mostly been adopted with respect to functional requirements [14], [21], several studies have shown the significance of non-functional requirements in software projects [22]-[25], and not correctly taking non-functional requirements into consideration is identified as one of the ten biggest risks in requirements engineering [26]. Therefore, non-functional requirements need to be considered throughout the first phase of the software development process (i.e. requirements engineering phase) and it has been acknowledged that the achievement of non-functional requirements along with functional requirements is critical to the success of a software system [11]-[15]. So, it is a critical task to recognize the important NFRs from the less important ones to guarantee the satisfaction of desired objectives.

There are several techniques introduced in the literature to identify the high-priority requirements from the low-priority ones. One of the powerful and flexible techniques which has been widely used to prioritize requirements is the Analytic Hierarchy Process (AHP), first introduced by Satty [16]. In AHP, all pairs of requirements are compared together to determine the priority level of one requirement over another requirement. By applying AHP, all requirements are first put in the rows and columns of a matrix. Then the user specifies his/her priority to each pair of requirements by assigning a preference value which is between 1 to 9, where 1 expresses equal value while 9 indicates extreme value. After that AHP converts these scales to numerical values, and consequently, a numerical priority is derived for each requirement. There are also some variations of AHP that have been proposed by researchers, such as Cased-Based Ranking (CBRank) [27] which adopts a preference elicitation process that combines sets of preferences elicited from human decision makers with sets of constraints which are automatically computed through machine learning techniques. It also exploits knowledge about partial rankings of the requirements that may be encoded in the description of the requirements themselves as requirement attributes (e.g., priorities or preferences). Cost-Value approach [1] is another AHP-based approach which prioritizes requirements based on two aspects: importance of requirement to customers, and cost of implementing each requirement.

In [17], Wiegner presents a framework in which requirements are prioritized based on four criteria defined as benefit, penalty, cost, and risk. The input values of these criteria are assessed on a scale from 1 (minimum) to 9 (maximum). The customer representative determines the benefit and penalty values whereas the development representative provides the cost and risk values associated with each requirement. Then, by using a formula, the relative importance value of each requirement is calculated. Azar et al. [18] describe a value-oriented prioritization (VOP) framework where requirements are linked to business values and prioritized based on the stakeholder ratings. VOP uses the relationships that exist between core business values to assess and prioritize relative relationships among those values, requirements and ensure their traceability. Company executives identify the core business values and use a simple ordinal scale to weight them according to their importance to the organization. Based on the data, a prioritization matrix is then constructed. In [4], a correlation-based priority assessment is proposed which prioritizes software process requirements gathered from multiple stakeholders by incorporating inter-perspective relationships of requirements. In [3], a technique is presented for partially automating the prioritization of requirements and raw feature requests in large-scale elicitation processes. Another approach in [19] applied genetic algorithm for prioritizing requirements in order to make prioritization scalable by reducing the number of pairwise comparisons. However, no particular method or approach has been proposed for performing the prioritization of non-functional requirements according to their relationship with functional requirements.

III. THE PROPOSED APPROACH

In order to perform the prioritization of non-functional requirements according to their relationship with functional requirements, we proposed an approach consisting of four steps, as shown in Table I. The first step is to elicit functional requirements as well as non-functional requirements. The second step is to set up the \( n \) functional requirements and the \( m \) non-functional requirements in the rows and columns of an \( n \times m \) decision matrix, respectively. The third step performs the elicitation of the importance degree of each NFR with respect to each FR. This step is the only step that requires decision maker input. The fourth and final step is an aggregation process to determine the prioritization of NFRs ranking with respect to all FRs using Triangular Fuzzy Number and Alpha cut approach. This step provides a decision maker with a prioritized list of NFRs with respect to all FRs.

As can be observed in Table I, using the proposed approach to prioritize NFRs involves four steps. The detailed explanation of these steps is illustrated over the following steps.

**Step 1:** Determine candidate FRs and NFRs

The first step is to identify the FRs and NFRs which require to be prioritized for inclusion in a software system. Let \( n \) be the number of candidate FRs and \( m \) the number of candidate NFRs. For demonstration of the process, we assume that there are four candidate functional requirements:
FR1, FR2, FR3, and FR4, and three non-functional requirements: NFR1, NFR2, and NFR3, which need to be ranked using the proposed approach.

<table>
<thead>
<tr>
<th>Step#</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Determine candidate FRs and NFRs</td>
</tr>
<tr>
<td>2</td>
<td>Construct the decision matrix</td>
</tr>
<tr>
<td>3</td>
<td>Elicit the importance degree of each NFR with respect to each FR</td>
</tr>
<tr>
<td>4</td>
<td>Calculate NFRs final ranking with respect to all FRs using Triangular Fuzzy Number and Alpha cut approach</td>
</tr>
</tbody>
</table>

**Step 2:** Construct the decision matrix

The second step is to generate an \( n \times m \) decision matrix, namely, \( D \), and insert the \( n \) functional requirements along with \( m \) non-functional requirements in the rows and columns of the decision matrix, respectively. Following the example stated in the previous step, a \( 4 \times 3 \) matrix is constructed, as shown in Fig. 1. The instructions on how to fill up the elements of the matrix \( D \) will be described in Step 3.

\[
D = \begin{bmatrix}
FR1 & NFR1 & NFR2 & NFR3 \\
FR1 & - & - & - \\
FR2 & - & - & - \\
FR3 & - & - & - \\
FR4 & - & - & - \\
\end{bmatrix}
\]

**Step 3:** Elicit the importance degree of each NFR with respect to each FR

The third step elicits the decision maker judgements for determining the importance degree of each NFR for a given FR. To elicit such an extent, our approach uses two scales: nominal scale and actual scale. Nominal scale is an interface scale which is utilized in order to enhance the user-friendliness of the approach for interacting with decision makers so that the decision maker does not have to know details about the actual scale. On the other side, the actual scale is a numerical scale which is used for internal calculations within the approach. In fact, the proposed approach exploits a five-point scale as actual scale. Table II demonstrates these scales.

**TABLE II: NOMINAL SCALE AND ACTUAL SCALE USED IN THE PROPOSED APPROACH**

<table>
<thead>
<tr>
<th>Nominal Scale</th>
<th>Actual Scale</th>
</tr>
</thead>
<tbody>
<tr>
<td>Very high importance (VHI)</td>
<td>1</td>
</tr>
<tr>
<td>High importance (HI)</td>
<td>0.75</td>
</tr>
<tr>
<td>Low importance (LI)</td>
<td>0.5</td>
</tr>
<tr>
<td>Very low importance (VLI)</td>
<td>0.25</td>
</tr>
<tr>
<td>No importance (NI)</td>
<td>0.001</td>
</tr>
</tbody>
</table>

Given a pair of FR and NFR (selected from rows and columns of matrix \( D \), respectively), the decision maker performs a set of activities aiming at determining the importance degree of each NFR for achieving each associated FR (see activities “select a pair” and “collect the importance degree of an NFR for a given FR” in Fig. 1). Each pair (i.e., FR and NFR) is assigned a value belonging to the nominal scale (see left column of Table II) which represents a qualitative measure of importance relation between the corresponding requirements. So, for all \( i \) and \( j \) with \( 1 \leq i \leq n \) and \( 1 \leq j \leq m \), the importance degree of the \( j \)th non-functional requirement for achieving the \( i \)th functional requirement would be assessed by a decision maker, leading to the value \( D_{ij} \) using Table II (e.g. if the value of \( D_{23} = \text{“very high importance”} \), it means that the decision maker believed that the non-functional requirement NFR3 has “very high importance” degree for achieving functional requirement FR2).

**Step 4:** Calculate NFRs final ranking with respect to all FRs using Triangular Fuzzy Number and Alpha cut approach

When all pairs have been evaluated, our proposed approach carries out the priority assessment of NFRs with respect to all FRs using Triangular Fuzzy Number and Alpha cut approach. In fact, during this step, IPA creates a prioritized list of NFRs through calculating the total importance degree of each NFR with respect to all FRs. The rationale behind this idea is that an NFR which achieves the highest total importance degree among all FRs, could be assigned as being a high-priority NFR. The following subsections illustrate a stepwise process of computing the priority vector of NFRs:

1) Convert the elements of matrix \( D \) into numerical values: First, the proposed approach converts all values of the decision matrix \( D \), which were specified according to the interface scale, into corresponding actual scales, resulting in the matrix \( \tilde{D} \).

2) Set up Triangular Fuzzy Number: In order to aggregate the different importance degrees of each NFR for different FRs, the Triangular Fuzzy Number (TFN) is calculated. TFN is capable of aggregating the subjective opinions of a decision maker through fuzzy set theory. In this study, we applied TFN since it is the most popular fuzzy number among the various shapes of fuzzy numbers. The Triangular Fuzzy Number \( T_{xi} \) is represented using the following equations (1-2):

\[
T_{xi} = (L_{xi}, M_{xi}, H_{xi}), \ i = 1..m, \text{ and } L_{xi}, M_{xi}, H_{xi} \in [0.001,1] \quad (1)
\]

\[
M_{xi} = \frac{\sum_{a=1}^{n} \min(D_{xia} \cdot D_{xib} \cdot D_{xic} \ldots D_{xin})}{n} \quad (2)
\]

where \( T_{xi} \) indicates the Triangular Fuzzy Number of non-functional requirement “\( x_i \)”, \( L_{xi} \), and \( H_{xi} \) represent the lowest and highest value of non-functional requirement “\( x_i \)”, respectively; \( M_{xi} \) is generated by calculating the geometric mean of all values belonging to the non-functional requirement “\( x_i \)”; \( m \) is the total number of NFRs; \( n \) is the total number of FRs; and \( D_{xia} \) specifies an opinion of a decision maker toward the importance degree of the non-functional requirement “\( x_i \)” for achieving the functional requirement “\( a \)”.

3) Constructing the fuzzy priority vector: After calculating the TFN value for each NFR, the fuzzy priority vector, namely, \( \tilde{F}_{x} \), is generated, as illustrated in Fig. 2. Notice, the values of \( \tilde{F}_{x} \) are derived from equation (1).

4) Defuzzification process: This study exploits the Alpha cut
approach, proposed by Liou and Wang [28], as shown in equation (3), to perform the defuzzification process. The defuzzification is accomplished in order to convert the calculated TFN values into quantifiable values, leading to the priority vector $W$.

$$\mu_{\alpha, \beta}(F_{x_i}) = [\beta \times f_a(L_{x_i}) + (1 - \beta) \times f_a(H_{x_i})], 0 \leq \alpha, \beta \leq 1$$

(3)

where $f_a(L_{x_i}) = (M_{x_i} - L_{x_i}) \times \alpha + L_{x_i}$, which represents the left-end boundary value of alpha cut for $F_{x_i}$; and $f_a(H_{x_i}) = H_{x_i} - (H_{x_i} - M_{x_i}) \times \alpha$, which indicates right-end boundary value of alpha cut for $F_{x_i}$.

$$T_{x_1} \quad T_{x_2} \quad \ldots \quad T_{x_m}$$

Fig. 2. Fuzzy priority vector, $F_x$.

In this context, $\alpha$ and $\beta$ carry the meaning of preferences and risk tolerance of decision maker, respectively. These two values range between 0 and 1, in such a way that a lesser value indicates greater uncertainty in decision making. Since preferences and risk tolerance are not the focus of this paper, a value of 0.5 is used for both $\alpha$ and $\beta$, for an extremely optimistic nor pessimistic about his/her judgments.

Finally, by normalizing the calculated priority vector, $W$, the vector $NW$ of normalized weights is obtained using equation (4):

$$NW_j = \frac{W_j}{\sum_{j=1}^{n} W_j}$$

(4)

The decreasing ordered non-functional requirements indicate the final ranking, where the most important non-functional requirement is the one with the highest $NW_j$ value. By applying the steps stated above, a decision maker is provided with a prioritized list of NFRs along with their corresponding importance values with respect to all existing FRs.

IV. ILLUSTRATIVE EXAMPLE

An intuitive comprehension of the proposed approach can be achieved by applying the proposed approach to an example, step by step, to demonstrate how the four steps of the approach could be utilized for a prioritization problem.

Step 1: Let us consider a prioritization problem defined over a set of four functional requirements $FRs = \{FR1, FR2, FR3, FR4\}$ and three non-functional requirements $NFRs = \{NFR1, NFR2, NFR3\}$, as well.

Step 2: In this step, a decision matrix of $4 \times 3$ is constructed where the rows correspond to functional requirements and the columns correspond to non-functional requirement.

Step 3: To fill up the elements of the decision matrix, the judgments of a decision maker are elicited and inserted into the matrix, as indicated in Fig. 3.

Step 4: To calculate the priority vector of NFRs with respect to all FRs, the elements of matrix $D$ are converted to actual scales (Step 4.1), the TFN is calculated for each NFR (Step 4.2), the fuzzy priority vector is constructed (Step 4.3), and the defuzzification is done in order to achieve the priority vector $W$ and $NW$ (Step 4.4). Table III shows this process where the elements of matrix $D$ with actual scales are indicated on the left side while the values of $F_{x_i}, W$ (Eq. 3), and $NW$ (Eq. 4) related to each NFR are represented on the right side of Table III. By calculating the priority vector $NW$, we are provided with a prioritized list of NFRs (numbers within parenthesis represent the priority of each NFR for considering during the development process).

![Table III: Example of Computing NFRs Priority Vector with Respect to All FRs](Fig. 3. The decision matrix is filled up with nominal scale values.)

V. TOOL SUPPORT

The proposed approach is supported by a software tool which allows to automate the steps of the proposed approach. The tool guides the user to apply his/her judgments between all possible pairs of functional and non-functional requirements in a similar way as the approach. Fig. 4 shows a snapshot of the tool graphical user interface.

The tool supports the user in the whole elicitation process. In particular, after the authentication, the tool presents the user an agenda of elicitations. The user can analyze the degree of every non-functional requirement for achieving each functional requirement, by selecting one of the radio buttons indicated in Fig. 4. When the user clicks ‘Submit’, the next pair of requirements is displayed. Finally, once all the evaluations have been performed, the system is able to compute and show the prioritized list of non-functional requirements, and their respective priority values.

The tool was developed in Microsoft Visual Studio 2008 and .NET Framework 3.5 using C# programming language to implement the proposed approach. Furthermore, for backing-up the data generated by TIPA, Microsoft SQL Server 2008 R2 was utilized.
VI. CONCLUSION

In this paper, we presented a detailed description of the proposed approach for non-functional requirements prioritization. This approach exploited the Fuzzy Logic and Alpha cut approach which allows the practitioners to perform the priority assessment of non-functional requirements according to their relationship with functional requirements since these two types of requirements have an impact on each other. The main contribution of the proposed approach over existing work is to establish a linkage between FRs and NFRs to perform the process of prioritizing NFRs.

We believe that the approach introduced in this paper is simple, easy to use and generates an accurate result. The result of applying the proposed approach for a given software project, which is an ordered list of non-functional requirements, may assist practitioners to concentrate on the most important non-functional requirements early in the life cycle rather than later in the life cycle when modifications are often difficult and impractical to accomplish. It can also help software architects to consider the most significant non-functional requirements as the main driver to design the system’s software architecture and also simplify the selection of suitable guidelines for achieving the desired quality attributes of a software system.

As part of the future work, it would be of interest to conduct a study which aims to evaluate the proposed approach in a real industrial setting in order to investigate the effectiveness as well as weakness of the approach in practical solutions.

REFERENCES


Mohammad Dabbagh is a PhD candidate in the Department of Software Engineering at University of Malaya, currently working as a research assistant. His research interests include requirements engineering, requirements prioritization, software quality, and software traceability. Mohammad received his master degree in software engineering.

Sai Peck Lee is a professor in the Department of Software Engineering at University of Malaya. She obtained her Ph.D. degree in computer science from Université Paris 1 Panthéon-Sorbonne. Her current research interests include requirements engineering, software traceability and clustering, software quality, software reuse, object-oriented techniques and CASE tools development. She has published an academic book, book chapters as well as over 100 papers in various refereed journals and conference proceedings. She has also actively participated in conference program committees and is currently in several experts review panels, both locally and internationally. She is a member of IEEE.