Value-Based Fuzzy Approach for Non-functional Requirements Prioritization

Khush Bakht Ijaz[1], Irum Inayat[1], Maya Daneva[2(✉)], and Faiza A. Bukhsh[2]

[1] NUCES, Islamabad, Pakistan
{i171097,irum.inayat}@nu.edu.pk
[2] University of Twente, Enschede, The Netherlands
{m.daneva,f.a.buksh}@utwente.nl

Abstract. Non-functional requirements (NFRs) are often addressed late in a project and, in turn, can get less attention in the requirements prioritization (RP) process. For various reasons, RP may happen based on functional requirements (FRs) only. While many approaches for prioritizing NFRs have been published, these are known also for some limitations, e.g. not being scalable, being domain-specific and not able to cope with changing requirements. In this paper, we proposes a value-based fuzzy approach for prioritizing NFRs together with FRs. Our proposed approach takes into account (1) the relationships of NFRs with FRs using experts' evaluations and fuzzy logic, and (2) the dependencies among both types of requirements and also the interdependencies that particularly exist among the NFRs themselves. We evaluated our proposal by conducting a real-world case study of an ATM system. We also compared the list of prioritized NFRs with the list of NFRs prioritized by different stakeholders on the basis of classification factors. The results of applying the proposed approach on NFRs of ATM system show that the approach produces a conflict-free and consistent list of prioritized NFRs.

Keywords: Non-functional requirements · Requirements prioritization · Fuzzy logic · Value-based requirements engineering · Design science · Empirical study

1 Introduction

Non-functional requirements (NFRs) are often addressed late [3] in requirements prioritization (RP) that happens early in the life cycle. One reason for this is that NFRs are rarely well-understood early in a project. Plus, variation of the perceived importance of the NFRs might well be possible due to various stakeholders' perspectives on them [31]. Systematic literature reviews on RP (e.g. [30, 31]) indicate that many approaches have been proposed for prioritizing NFRs as part of all requirements in a project, however these often lack scalability and pay only insufficient attention to requirements dependencies, be it dependencies between functional requirements (FRs) and NFRs, or NFRs interdependencies [33]. To counter these issues, recent efforts of the RE community focused on the application of fuzzy logic based techniques (e.g. [30]). While these proposals have been demonstrated to work in the specific contexts of the authors designing

B. Shishkov (Ed.): BMSD 2021, LNBIP 422, pp. 330–342, 2021.
https://doi.org/10.1007/978-3-030-79976-2_21

them, no generalizable and conclusive evidence has been produced so far regarding the strong and weak points of these approaches in real-world contexts. This paper contributes to the exploration of the application of fuzzy logic techniques for RP purposes. Drawing on previously published research [6, 7] on fuzzy logic in RP, we propose an approach named Value-based Fuzzy Requirement Prioritization that accounts for the relationships and interdependencies between NFRs and FRs [33] as perceived by experts in requirements engineering. Our proposal for a NFRs prioritization approach has been developed and evaluated by using Design Science [2] as our research method. Our work extends the application of fuzzy logic techniques [6, 7] to prioritize NFRs which so far has not been investigated in published literature. The proposed approach aims at helping requirement engineers in prioritizing large number of NFRs through a two-stage prioritization accounting for both stakeholders and experts. To evaluate this approach, we conducted a nearly real-world experimental study. The rest of the paper is organized as follows: Sect. 2 provides background on RP. Section 3 describes our research process. Section 4 proposes our RP approach. Section 5 is on its first experimental evaluation. Section 6 discusses this evaluation and the implications of this research. Sections 7 is on limitations and Section 8 concludes.

2 Background and Related Work

NFRs such as performance and security, can be considered as the constraints on a software system [8] that describe aspects such as how the system is performing and how secure it is to use, respectively. These aspects help software architects understand the architecture designs that best match the NFRs and the order in which they would be scheduled for implementation.

In the literature in the field of Requirements Engineering (RE), some approaches for NFRs prioritization exist. Examples are the CEP (Capture Elicit Prioritize) [11, 13] automated approach, the αβγ framework [12] for prioritizing NFRs, and the NERV methodology [14]. Next, an approach [19] leveraging the Analytic Hierarchy Process (AHP) [5] that focuses on interrelationships present between candidate NFRs has also been put forward. Moreover, other proposals include the hybrid approach that prioritizes FRs based on NFRs [15], the HAM (Hybrid Assessment Method) [16] that defines criteria for prioritization and also performs pair-wise comparisons of NFRs as used in [15] for defining trade-offs, simultaneous and separate prioritization approaches for NFRs and FRs [17], the NFR planning method for agile processes (NORPLAN) [18] which is a part of the NORMAP methodology discussed in [13]. Most of these approaches were demonstrated to work for only relatively small number of requirements. Moreover, studies show (e.g. [31]) that the proposed RP approaches are not guaranteed to be flexible and able to deal with ever-changing requirements (be it FRs or NFRs). This motivated our work on defining an approach that is both flexible and scalable in prioritizing NFRs.

Our work draws on published research by other authors on intelligent value-based approaches to NFRs prioritization. To the best of our knowledge, four studies [6, 7, 26, 27] have been published on such approaches. Ramzan et al. [7] proposed an intelligent value-based technique for RP based on fuzzy logic and expert systems. Kukreja et al. [26] presented a method concerned with providing verification to show that value based RP

frameworks are effective and can help software development organizations to implement the most important requirements in earlier phases of software development life cycle. Next, the approach presented in [6] attempts to overcome the limitations present in existing RP approaches and suggests an intelligent-value-based approach able to produce the list of requirements prioritized based on the value assigned to the requirements. Finally, Padmanabhuni [27] deals with identifying a suitable framework for value-based RP. The selection of this framework is based on the nature of requirements. The work of these researchers [6, 7, 26, 27] inspired us in including the value-based perspective on software engineering, in the development of our approach.

3 Research Methodology

Our research process was inspired by the Design Science methodology [2] which aims at creating artefacts (methods and techniques) to solve real-life problems in information systems development and in software engineering. A design-science-based research process starts with goal-setting, and then proceeds with the creation of a method proposal and its evaluation in a realistic context. In the next sections, we first present the proposal of a RP method for NFRs and then we use an example of its application in a real-world case of an ATM system. The overall goal for using design science is to create a RP approach that accounts for the interdependencies among NFRs in a project as well as the interdependencies between NFRs and FRs.

4 Our Proposed Approach

This section provides a brief description of the concept of value-based fuzzy RP and then it elaborates on our proposed approach (called Value-based Fuzzy Requirement Prioritization).

As already said, our approach is grounded on the fuzzy logic theory and the value-based perspective. The fuzzy logic theory served as the foundation to create method of reasoning that resemble human reasoning. In the case of RP, generally fuzzy logic emulates the way of human decision making that involves the range of possibilities between digital values YES and NO. Our method also draws on the Value Based Intelligent RP technique of Ramzan et al. [7], which promotes iterative and multilevel prioritization and classification of NFRs from the perspectives of (i) stakeholders and (ii) experts in software projects. The iterative nature of the prioritization process makes sure that requirements are evaluated and re-evaluated by different actors and a more realistic priority ranking is achieved. In the technique of Ramzan et al. [7], assuming a set of elicited requirements has been documented and made available for RP, a *two-fold prioritization* takes place: first, the requirements are prioritized by the participating stakeholders and then by RE experts. Plus, the stakeholders themselves are prioritized by the experts. A priority assigned to a stakeholder is called a stakeholder profile. The experts rank stakeholders profiles on a scale of 1–10, where 1 means the least important, and 10 means the most important. The experts use this scale also for ranking the requirements. Using the stakeholders' profiles, the experts identify the importance of requirements provided by the particular stakeholder. Experts assign prioritization values to the requirements on

the basis of requirement classification factors (RCFs) – these are prioritization criteria that are chosen by the RE experts specifically for the project as a whole, or for particular groups of requirements within the project. Examples of prioritization criteria are: importance, risk, requirement dependencies, development time, cost and technical debt. For a specific project and requirement in this project, these RCFs are assigned a score in the range of 0–5. The lowest value i.e. 0 indicates that particular factor is not present in a particular requirement and 5 indicates the high involvement of a classification factors in requirements. The requirement value (RV) is then estimated by using the formula [7] below:

$$RV = 0.35 + 0.02\left\{\sum_{i=1}^{n} pRCFi + \sum_{i=1}^{n} rRCFi\right\}$$

where *pRCF* stands for 'project-specific Requirements Classification Factor' and *rRCF* stands for 'requirement-specific Requirement Classification Factors. We note that the RV value is in the range of 0.35 (when all RCFs are scored to be 0) to 1.35 (when all RCFs are scored to be 5). We note that the constant 0.35 has been determined by Ramzan et al. [7] and we borrowed it in our method for the reason of assuring consistency of terminology.

Second, a fuzzy logic based algorithm (namely, the *fuzzy c mean algorithm* [21]) is then applied. This means that requirements are grouped by applying fuzzy membership value. Furthermore, we use the concept of intelligent value-based RP to the area of NFRs. And finally, our proposed approach aims at achieving two goals: (1) it should account for the dependencies between FRs and NFRs, and (2) it should account for the inter-dependencies of the NFRs themselves.

Below we describe the steps of the approach. Assuming a list of FRs and NFRs exists for a project, our approach includes:

Step 1. Determine the importance value of each NFR with respect to given FRs. Based on this importance value, the preliminary list of ranked FRs and NFRs is obtained.
Step 2. A decision matrix is constructed by placing the NFRs in a column and putting the corresponding FRs in rows.
Step 3. In the matrix, an ordinal scale is used to assign importance value to NFRs with respect to FRs. These values are put in the cells of the matrix. The values are assigned on a scale of 0–1, where 1 means 'most important' and 0 means "not important".
Step 4. NFRs final ranking is calculated by taking the weighted average of all the values belonging to NFR against all FRs. The NFR which gets the highest weight is given the highest priority.
Step 5. In order to account for the interdependencies among NFRs and to assure that the NFRs are conflict-free, we employ the NFRs conflict resolution method that was first presented by Dabbagh and Lee [28]. It identifies conflicting NFRs and then offers strategies to experts to resolve the NFRs conflicts before prioritization takes place. Because of space limitation, we do not present this approach in detail. Instead, we refer interested readers to the articles of Dannagh and Lee [28].
Step 6. The list of conflict-free NFRs is given to the experts. They assign prioritization values to the NFRs on the basis of the RCFs (i.e. importance, risk, requirement dependency, development time, cost and penalty). As indicated earlier, each of these factors is assigned a score in the range of 1–5.

Step 7. Requirement value (RV) is then estimated by applying the formula below. As indicated earlier, RV is in the range 0.35 to 1.35.

$$RV = 0.35 + 0.02\left\{\sum_{i=1}^{6} RCFi\right\}$$

Step 8. The *Fuzzy c mean algorithm* [21] is then applied to the final list of requirements prioritized using RVs and the initial prioritized list of NFRs. This ultimately leads to final priority ranks.

5 Our First Evaluation

The proposed approach has been evaluated by conducting an experimental study using real world data in a case of an ATM system. A real set of requirements has been obtained from the software requirement specification document of this ATM system. Both FRs and NFRs were defined. The total number of requirements is 40, including 20 FRs (see Fig. 1) and 20 NFRs (see Fig. 2).

FUNCTIONAL REQUIREMENTS OF ATM SYSTEM:

FR 1.1	If no cash card is in the ATM, the system shall display message on the screen.
FR 1.2	If the ATM is running out of money, no card shall be accepted. System displays an error message.
FR 1.3	The ATM has to check if the entered card is a valid cash card.
FR 1.4	If the cash card is valid, ATM shall read the serial number and bank code.
FR 1.5	System shall ask the user to enter his password. ATM shall verify the bank code and password with bank computer.
FR 1.6	The bank computer gets a request from the ATM to verify an account.
FR 1.7	If it is not a valid bank code the bank computer shall send a message to the ATM.
FR 1.8	The bank computer checks if the password is valid for a valid cash card.
FR 1.9	Bank computer shall process a transaction from the ATM if the details are valid.
FR 1.10	System shall display error message to the user if password and serial number are incorrect.
FR 1.11	If the password and serial number are correct authorization process is finished.
FR 1.12	If a card is entered more than three times and each time wrong password is entered then the card is kept by the ATM and system displays error message to the customer.
FR 1.13	System shall offer different kinds of transactions i.e. withdraw and deposit.
FR 1.14	System shall re initiate transaction dialogue if the amount entered by the user is not within the pre defined transaction policy.
FR 1.15	System shall perform valid transaction and wait for response from bank computer.
FR 1.16	System shall dispense the money if transaction is successful.
FR 1.17	Bank computer shall update the account of customer after processing the transaction.
FR 1.18	After dispensing the money, system shall log the amount with serial number of card. Notification is sent to the bank about dispensed money.
FR 1.19	If the transaction is not successful an error message should be displayed and the card should be ejected.
FR 1.20	ATM shall allow the customer to re login the account.

Fig. 1. FRs of the ATM system

The experiment started by first prioritizing the NFRs based on the six prioritization criteria (these are the RCFs) as shown in Fig. 3 (see columns B to G) with the participation of 7 stakeholders. This is **Step 1** in our approach. In Fig. 3, the criterion in column B reflects how much each NFR is important to be implemented for the stakeholders. The requirement dependency criterion (Req-dep, see column D) shows the extent to which particular NFRs is dependent upon a FR. Development time is the time required to implement the requirement. Cost of implementing the requirement has been calculated by adding Requirement dependency and development time. Penalty value is assigned

NON FUNCTIONAL REQUIREMENTS OF ATM SYSTEM:

NFR 1.1	The card verification time must not exceed 0.8 seconds under normal server workload and 1 second under peak server workload.
NFR 1.2	The pin number verification time must not exceed 0.3 sec. Under normal server workload and 0.5 sec. under peak server workload.
NFR 1.3	Cash withdrawal transaction time must not exceed 4 sec. under normal server workload and 5 sec. under peak server workload.
NFR 1.4	Receipt printing time after must not exceed 3 sec. Under normal server and peak server workload.
NFR 1.5	The product shall have a backup power supply in case of power failures.
NFR 1.6	Any abnormal operations shall result in the shutting down of the system.
NFR 1.7	After abnormal shutdown of the ATM, the system shall have to be manually restarted by maintenance personnel.
NFR 1.8	The system shall be compatible with AIMS security standards.
NFR 1.9	If there is no response from the bank computer after a request within 2 minutes the card is rejected with an error message.
NFR 1.10	User should be provided with only three attempts for login failing which his card needs to be blocked.
NFR 1.11	The ATM network has to be available 24 hours a day.
NFR 1.12	Only maintainers are allowed to connect new ATM's to the network.
NFR 1.13	The system should have the mechanism of self monitoring periodically in order to detect any fault.
NFR 1.14	Passwords shall not contain name of customers as they are easy to be hacked.
NFR 1.15	The memory system of ATM shall be of non volatile type.
NFR 1.16	The data communication protocol shall be such that it ensures reliability and quality of data and voice transmission in a mobile environment.
NFR 1.17	The system should inform the main branch automatically as soon as it detects any error. The kind of fault and the problem being encountered should also be mentioned by the system automatically.
NFR 1.18	Touch screen and button response time must not exceed 5000ms.
NFR 1.19	All functionality of ATM system shall be thoroughly tested.
NFR 1.20	There shall be a secured cash vault with a combination locking system.

Fig. 2. NFRs of the ATM system

when a required feature is not implemented. The values corresponding to each NFR and each criterion (also called RCF) are assigned by the 7 different stakeholders on a scale from 1–5. The two rightmost columns indicate the prioritized NFRs based on these 6 RCFs. Fig. 3 shows that NFR 1.13 and 1.18 are assigned highest priority values. In contrast, the least values are assigned to NFR 1.1 and 1.4. This means that NFR 1.13 and 1.18 are required to be implemented first and NFR 1.1 and 1.4 should be implemented the last.

A	B	C	D	E	F	G	H	I
Req id	Importance	Risk	Req-dep	Devp-Time	cost	Penalty	Priotity %	Priority
NFR 1.1	4	1	2	2	4	5	0.09	9
NFR 1.2	5	4	2	3	5	5	0.14	14
NFR 1.3	4	4	1	3	4	5	0.12	12
NFR 1.4	3	2	2	2	4	4	0.09	9
NFR 1.5	5	4	5	5	10	5	0.19	19
NFR 1.6	1	4	4	2	6	2	0.11	11
NFR 1.7	2	4	4	4	8	2	0.14	14
NFR 1.8	5	4	4	2	6	4	0.15	15
NFR 1.9	2	4	3	1	4	1	0.1	10
NFR 1.10	3	4	3	3	6	1	0.13	13
NFR 1.11	5	5	4	5	9	4	0.19	19
NFR 1.12	4	4	4	4	8	5	0.16	16
NFR 1.13	5	5	5	5	10	5	0.2	20
NFR 1.14	5	5	4	3	7	5	0.17	17
NFR 1.15	2	3	3	3	6	3	0.11	11
NFR 1.16	4	5	4	4	8	2	0.17	17
NFR 1.17	4	4	4	4	8	2	0.16	16
NFR 1.18	5	5	5	5	10	2	0.2	20
NFR 1.19	4	2	2	2	4	5	0.1	10
NFR 1.20	3	3	4	5	9	1	0.15	15

Fig. 3. Prioritized list of NFRs based on RCFs selected for the project

In order to assign an importance value to each NFRs with respect to FRs, a decision matrix was created (**Step 2**) by placing all 20 NFRs in columns and all FRs in rows (see Fig. 4).

	NFR 1.1	NFR 1.2	NFR 1.3	NFR 1.4	NFR 1.5	NFR 1.6	NFR 1.7	NFR 1.8	NFR 1.9	NFR 1.10	NFR 1.11	NFR 1.12	NFR 1.13	NFR 1.14	NFR 1.15	NFR 1.16	NFR 1.17	NFR 1.18	NFR 1.19	NFR 1.20
FR1.1	0.25	0.25	0.25	0	0.5	0.75	0.25	0.75	0	0.25	1	0.5	0.75	0	0.75	1	0.75	1	0.5	0.5
FR1.2	0.5	0	0	0.25	0.75	1	0.25	0.25	0	0.5	0.75	0.5	1	0.25	0.5	0.5	1	0.5	0.75	0.75
FR1.3	1	0.75	0.25	0.5	0.75	0.75	0.25	0.5	0.25	0.25	0.75	0.5	1	0.5	0.75	0.5	0.75	1	0.5	0.75
FR1.4	0.25	1	0.25	0.25	0.25	0.25	0.25	0.25	0.75	0.25	0.25	0.25	0.75	0.5	0.75	0.25	0.75	0	0.25	0.25
FR1.5	0.75	0.25	0.75	0.5	0.75	0.75	0.5	0.5	0.75	0.5	0.5	0.5	0.75	1	0.5	0.5	0.75	0.5	0.5	0.5
FR1.6	0	0.5	0	0.75	0	0	1	1	0.25	0.75	0.75	0.25	0	1	0.75	1	0	0.5	0.25	0.75
FR1.7	0.25	0.75	0.25	0.75	0.25	0.25	0.75	0.25	1	0.75	0.75	0.25	0	0.75	0.5	0.25	0	0.25	0.75	0.25
FR1.8	0.5	0	0.5	0.75	0.5	0.5	0.75	0.25	0.75	0.25	0.25	0.25	1	0.5	0.25	0.25	0.75	0.5	0.25	0.25
FR1.9	0.5	0.25	0.75	0	0.25	0.75	0.5	0.5	0.75	0.5	0.5	0.5	0.75	0.5	0.5	0.5	0.75	0.5	0.5	0.5
FR1.10	0	0.5	0.25	0	0.75	0.75	0.25	1	0	0.25	0.75	0.75	0.5	0.25	0.25	0.75	1	0.75	0.25	0.75
FR1.11	0.75	0.75	0.75	0.25	0	0.25	0.25	0.25	0	0.75	0.75	0	0	0.75	0.25	0.25	0	0.5	0.75	0.5
FR1.12	0.75	0.75	0	0.5	0.25	0.75	1	0.25	0.75	1	0.25	0	0.75	0	0.25	0.25	0.75	0.25	0.25	0.25
FR1.13	0.75	0.75	0.25	0.25	0.5	0.75	0.25	0.5	0.75	0.5	1	0.5	0.75	0.5	0.75	0.5	0.75	0.25	0.5	0.5
FR1.14	0	0.5	0.5	0.5	0.25	0.5	0.25	0.75	0.5	0.25	0.75	0.5	0.25	0.5	0.75	0.75	0	0.75	0.25	0.75
FR1.15	0.25	0.25	0.75	0.75	0.75	0.25	1	0.25	0.5	0.75	0.75	0.25	0.5	0.75	0.75	0.25	0	0.75	0.75	0.25
FR1.16	0.75	0.25	1	0.75	0	0.25	0.5	0.25	0.75	0.25	0.25	0.25	0.75	0	0.25	0.25	0.75	1	1	1
FR1.17	0	0.25	0.25	0	0.25	0.25	0.5	0.5	0.75	0.5	0.5	0.5	0.75	0.5	0.5	0.5	0.75	0.5	0.5	0.5
FR1.18	0.25	0	0.75	1	0.5	0.75	0.25	0.75	0	0.25	0.75	0.5	0.5	0.25	0.5	0.75	0.25	0.75	0.25	0.75
FR1.19	0.5	0.25	0.5	0.25	1	0.5	0.5	0.25	0	1	0.75	0.25	1	0.75	1	1	0	0.75	0.75	0.75
FR1.20	0.75	0	0.75	0.5	0	0.25	0.25	0.5	0.75	0.25	0.5	0.25	0.75	0	0.25	1	0.75	0.25	1	0.75
WAvg%	**43.75**	**40.0**	**39.75**	**42.5**	**41.25**	**38.8**	**47.5**	**45**	**46.25**	**48.75**	**58.75**	**36.25**	**62.5**	**50**	**51.25**	**56.25**	**52.5**	**60**	**53.75**	**57.5**

Fig. 4. Importance values assigned to NFRs with respect to FRs (**Steps 2, 3 and 4**).

In Fig. 4, using the scale of 0 to 1 defined in [29], the following values are assigned: 1 (being very highly important), 0.75 (highly important), 0.5 (low important), 0.25 (very low important) and 0 (not important). This is **Step 3** in our approach. The NFRs final ranking (**Step 4**) is calculated by taking the *weighted average* of all the values belonging to NFR against all FRs, see the last row in the table of Fig. 4. The NFR which gets highest weight is given higher priority.

To acknowledge for NFRs interdependencies, conflicts between NFRs are identified and resolved (**Step 5**) by using the technique of Dabbagh and Lee [28]. We note that in the case of the ATM, its application led to the elimination of two NFRs as throughout the conflict resolution they were found unimportant for the project. The list of remaining 18 conflict-free NFRs is then given to the group of ten experts (**Step 6**) tasked with assigning prioritization values to the NFRs based on the selected RCFs for the project as shown in Fig. 3: importance, risk, requirement dependency, development time, cost and penalty. Fig. 5 shows the scores assigned to all NFRs by the experts, in regard to these six RCFs.

For each NFR, the requirement value is then estimated by applying the formula for RV:

$$RV = 0.35 + 0.02\left\{\sum_{i=1}^{6} RCFi\right\}$$

This is **Step 7** of our approach. The RV value against each NFR is shown in Fig. 6, left. Those NFRs having higher RV value are given higher priority. In line with this, the highest priority is assigned to NFR 1.13 (see the top row in Fig. 6, left).

The fuzzy c mean algorithm [21] is then applied (Step 8) to the final list of requirements that are prioritized using the RV values as shown in Fig. 6, left. In this study, the

	RCFs					
	Importance	Risk	Req-dependency	Dev-Time	Cost	Penalty
NFR 1.13	5	4	5	4	9	5
NFR 1.18	4	4	5	4	9	4
NFR 1.11	4	3	4	4	8	4
NFR 1.20	4	5	2	2	4	5
NFR 1.16	3	2	1	1	2	4
NFR 1.19	3	1	4	2	6	4
NFR 1.17	3	2	3	3	6	3
NFR 1.15	2	2	1	2	3	2
NFR 1.14	1	4	4	4	8	1
NFR 1.10	1	4	4	4	8	4
NFR 1.7	2	2	3	4	7	2
NFR 1.9	3	3	2	2	4	1
NFR 1.8	3	3	2	3	5	1
NFR 1.1	2	3	1	4	5	3
NFR 1.4	2	1	2	3	5	1
NFR 1.2	2	2	2	3	5	2
NFR 1.3	1	1	4	1	5	1
NFR 1.6	1	3	5	4	9	2

Fig. 5. Scores assigned to the NFRs by 10 field experts (**Step 6**)

NFR 1.13	RV=0.99
NFR 1.18	RV=0.95
NFR 1.11	RV=0.89
NFR 1.20	RV=0.79
NFR 1.16	RV=0.61
NFR 1.19	RV=0.75
NFR 1.17	RV=0.75
NFR 1.15	RV=0.59
NFR 1.14	RV=0.79
NFR 1.10	RV=0.85
NFR 1.7	RV=0.75
NFR 1.9	RV=0.65
NFR 1.8	RV=0.69
NFR 1.1	RV=0.71
NFR 1.4	RV=0.63
NFR 1.2	RV=0.67
NFR1.4	RV=0.63
NFR 1.3	RV=0.61
NFR 1.6	RV=0.83

NFRs	Priority value	C1	C2	C3
NFR 1.1	(14,11)	4.95	8.03	11.9
NFR 1.2	(16,13)	7.49	10.7	14.7
NFR 1.3	(17,17)	10.6	13.8	18.1
NFR 1.4	(15,15)	7.73	11.0	15.2
NFR 1.6	(18,5)	10.1	11.8	13.9
NFR 1.7	(11,10)	1.83	4.89	8.91
NFR 1.8	(13,12)	8.56	7.62	11.7
NFR 1.9	(12,14)	4.96	8.17	12.5
NFR 1.10	(10,4)	5.98	5.15	5.90
NFR 1.11	(3,3)	9.27	6.06	1.75
NFR 1.13	(1,1)	12.1	8.86	4.57
NFR 1.14	(9,6)	3.92	3.04	5.16
NFR 1.15	(8,18)	8.16	10.2	14.2
NFR 1.16	(5,16)	7.37	8.29	11.7
NFR 1.17	(7,9)	2.36	1.19	5.47
NFR 1.18	(2,2)	10.7	7.45	3.16
NFR 1.20	(4,7)	5.94	2.74	2.65
NFR 1.19	(6,8)	3.71	0.61	4.11

Fig. 6. The RV values for the NFRs (left) and the final cluster values after 4th pass (right)

number of clusters has been selected as c=3. Thus, each cluster contains 6 NFRs (as the total is 18). Randomly three centroid values have also been given to these clusters i.e. for cluster 1, value 9 has been given as initial prioritization value and 10 is given as RV-based prioritized value. Similarly, random values have been assigned to other clusters as well. The distance from a point to cluster center has been defined as the Euclidean distance. The fuzzifier value "m" – which is the parameter controlling how fuzzy the cluster will be – is assumed to be 2 in this study. The number of centroids has been set as 3. Fig. 6 (see the right side) shows the final cluster values after the 4th pass.

The final prioritized list of NFRs is shown in Fig. 7. Therein, the highest priority is assigned to NFR 1.13 and least priority is assigned to NFR 1.15. To sum up, by applying value based fuzzy prioritization approach to the NFRs of the ATM system, we obtained

Final prioritized list of conflict free NFRs	
NFR	**Priority**
NFR 1.13	1
NFR 1.18	2
NFR 1.11	3
NFR 1.10	4
NFR 1.6	5
NFR 1.14	6
NFR 1.20	7
NFR 1.19	8
NFR 1.17	9
NFR 1.7	10
NFR 1.1	11
NFR 1.8	12
NFR 1.2	13
NFR 1.9	14
NFR 1.4	15
NFR 1.16	16
NFR 1.3	17
NFR 1.15	18

Fig. 7. Final prioritized list of NFRs (the result of Step 8)

a conflict-free list of prioritized NFRs. This list accounted for both the dependencies between NFRs and FRs and the interdependencies among the NFRs themselves.

6 Discussion

We compared the list of prioritized NFRs obtained by applying our value-based fuzzy prioritization approach (Fig. 7) with the list of NFRs prioritized by stakeholders on the basis of RCFs i.e. the prioritization criteria (Fig. 3). The prioritization criteria have been kept same for both the RP processes (i.e. for the value-based NFRs prioritization (our proposed approach) and the prioritization based on values assigned by stakeholders as shown in Fig. 3). The results of our experimental study show that the proposed approach produces better outcome in three regards: (i) our approach produced a *ranking* (as opposed to the percentages indicated in the column "Priority %" in Fig. 3), (ii) we obtained a prioritized list of *conflict-free* NFRs, and (iii) we implemented the *two-stage* RP by involving 10 experts in addition to the 7 stakeholders (Fig. 3). Our research has some implications for practitioners and for researchers. First, the proposed method keeps the requirement engineers focused on the prioritization of NFRs during early software development phase. As earlier research suggested [32], often the NFRs are treated late and then approached by software architects, developers, and even testers. Second, having a method for RP of NFRs helps project managers including NFRs in the estimation of effort of their projects. As Kassab et al. [34] indicate, project estimation is better, if it accounts for the NFRs' interdependencies and if it is aware of the NFRs conflicts. To RE researchers, the proposed fuzzy logic based approach opens up some interesting research directions First, the method assumes that stakeholders will assign priorities to the existing requirements. If stakeholder's involvement is not possible at that stage, what

heuristics the RE specialist responsible for the project, could adopt in order to be able to create the inputs for the matrix that is created in the first steps of our method? Second, it is good to know if the applicability of the method depends on the characteristics of the application domain and the size of the project. Finally, we note that our evaluation study included 40 requirements in a realistic project (ATM). Although, this is more than what is included in the demonstrations of other published RP proposals, we can no claim for certain the scalability of our approach to contexts in which thousands or hundreds of thousands of requirements are to be prioritized. We think that researchers together with practitioners may conduct an empirical research on large-scale requirements repositories to explore the scalability, the accuracy and the efficiency of the proposed approach. Also, a comparative study may also be conducted to explore how our method compares with other techniques.

7 Limitations

This research has some limitations. As Hevner et al. [2] suggests, an important validity concern is the generalizability of our proposal. We did a very first evaluation of the RP method on a realistic project in experimental settings. This included 40 requirements. Clearly, most projects would have more. We however think that if the steps of our approach are fully automated, dealing with long lists of requirements would not be a problem. Would the approach be suitable to all project contexts? We think that the approach is more suitable for projects in which diverse NFRs are to be considered and their dependencies must be analysed in detail. This would be for example in contexts of large scale online systems, such as gaming software, social network software or e-commerce sites that are concerned with scalability, performance, privacy, security, usability, learnability and ease of use. In contrast to this, we think that certain types of applications such as administrative software systems (e.g. in human resources, in accounting) where the number of users is limited, would not benefit vastly from our approach. This is not to say that the projects developing such applications can not employ our method; they certainly could, however they might find no big difference between using our method and any other RP technique available in the marketplace. In this kind of contexts, there are usually a very small number of NFRs [32] and they are well understood and the trade-offs among them might well be known to the software architects.

8 Conclusions

This paper proposed an approach that employs intelligent value based fuzzy prioritization to NFRs. We made a step towards a RP solution based on fuzzy logic by extending the earlier work done on value based fuzzy requirement prioritization [6, 7]. Using a realistic case, an ATM system and its FRs and NFRs specifications, we have demonstrated that if a requirements engineer has a specification at his disposal, he/she would be able to prioritise the NFRs early in the development process by taking into account the NFRs interdependencies and also the dependencies between NFRs and FRs.

Manually performing all steps and computations is time consuming and requires a lot of efforts. Moreover the computations may also be prone to errors as they have been

performed manually. Therefore, our immediate future work is to build a tool support for this approach so that all these limitations can be overcome. Plus, we plan to use this tool in other realistic contexts in order to evaluate the strong and weak points of our proposed method. Specifically, we expect to start more evaluation studies in the Netherlands in those companies that are RE research partners in the research projects of the last co-author). The proposed approach will be applied on larger set of non-functional requirements i.e. more than 20, and the results will then be compared with other existing approaches for NFR prioritization. More subjects will be involved to get larger number of functional and non-functional requirements of any real software system in order to give stronger foundation to the results.

We before, our future work also includes empirical studies in companies in order to evaluate the acknowledge that our experimental study was applied to relatively small set of NFRs (20). Ther applicability, the usefulness and the utility of our approach in real-world projects. Only then, we could make firmer conclusions about the qualities of our proposed method.

References

1. Yin, B., Jin, Z.: Extending the problem frames approach for capturing non-functional requirements. In: IEEE 11th International Conference on Computer and Information Science, pp. 432–437 (2012)
2. Hevner, A.R., March, S.T., Park, J., Ram, S.: Design science in information systems research. MIS Q. **28**(1), 75–105 (2004)
3. Nguyen, Q.L.: Non-functional requirements analysis modeling for software product lines. In: ICSE Workshop on Modeling in Software Engineering, pp. 56–61 (2009)
4. Gupta, V., Chauhan, D.S., Dutta, K.: Exploring reprioritization through systematic literature surveys and case studies. Springerplus **4**(1), 1–15 (2015). https://doi.org/10.1186/s40064-015-1320-0
5. Kassab, M., Kilicay-Ergin, N.: Applying analytical hierarchy process to system quality requirements prioritization. Innovations Syst. Softw. Eng. **11**(4), 303–312 (2015). https://doi.org/10.1007/s11334-015-0260-8
6. Ramzan, M., Jaffar, M.A., Iqbal, M.A., Anwar, S., Shahid, A.A.: Value based fuzzy requirement prioritization and its evaluation framework. In: 4th International Conference on Innovative Computing, Information and Control (ICICIC), pp. 1464–1468 (2009)
7. Ramzan, M., Jaffar, A., Ali Shahid, A.: Value based intelligent requirement prioritization (Virp): Expert driven fuzzy logic based prioritization technique. Int. J. Innov. Comput. Inf. Control (2011)
8. Baskaran, S.: A survey on prioritization methodologies to prioritize non functional requirements. Int. J. Comput. Sci. Bus. Inform. **12**(1), 32–44 (2014)
9. Dabbagh, M., Lee, S.P., Parizi, R.M.: Functional and non-functional requirements prioritization: empirical evaluation of IPA, AHP-based, and HAM-based approaches. Soft. Comput. **20**(11), 4497–4520 (2015). https://doi.org/10.1007/s00500-015-1760-z
10. Phillips, L.B., Aurum, A., Svensson, R.B.: Managing Software Quality Requirements. In: 38th Euromicr, pp. 349–356 (2012)
11. Maiti, R.R., Mitropoulos, F.J.: Capturing, eliciting, predicting and prioritizing (CEPP) non-functional requirements metadata during the early stages of agile software development. In: SoutheastCon, pp. 1–8 (2015)

12. Aasem, M., Ramzan, M., Jaffar, A.: Analysis and optimization of software requirements prioritization techniques. In: International Conference on Information and Emerging Technologies, pp. 1–6 (2010)
13. Maiti, R.R., Mitropoulos, F.J.: Capturing, eliciting, and prioritizing (CEP) NFRs in agile software engineering. In: SoutheastCon pp. 1–7 (2017)
14. Domah, D., Mitropoulos, F.J.: The NERV methodology: a lightweight process for addressing non-functional requirements in agile software development. In: SoutheastCon pp. 1–7 (2015)
15. Garg, U., Singhal, A.: Software requirement prioritization based on non-functional requirements. In: 7th International Conference on Cloud Computing, Data Science and Engineering, pp. 793–797 (2017)
16. Dabbagh, M., Lee, S.P., Parizi, R.M.: Application of hybrid assessment method for priority assessment of functional and non-functional requirements. In: ICISA 2014, pp.1–4 (2014)
17. Chopra, R.K., Gupta, V., Chauhan, D.S.: Experimentation on accuracy of non functional requirement prioritization approaches for different complexity projects. Perspect. Sci., **8**, Supplement C, pp. 79–82 (2016)
18. Farid, W.M., Mitropoulos, F.J.: NORPLAN: non-functional requirements planning for agile processes. IEEE Southeastcon, pp. 1–8 (2013)
19. Fellir, F., Nafil, K., Touahni, R.: System requirements prioritization based on AHP. In: 3rd IEEE CIST, pp. 163–167 (2014)
20. Paucar, L.H.G., Bencomo, N.: ARRoW: tool support for automatic runtime reappraisal of weights. IEEE 25th RE, pp. 458–461 (2017)
21. Bezdek, J.C.: Pattern Recognition with Fuzzy Objective Function Algorithms (1981)
22. Thakurta, R.: A framework for prioritization of quality requirements for inclusion in a software project. Softw. Qual. J. **21**(4), 573–597 (2013)
23. Kassab, M.: An integrated approach of AHP and NFRs framework. In: RCIS 2013, pp.1–8 (2013)
24. Singh, P., Singh, D., Sharma, A.: Rule-based system for automated classification of non-functional requirements from requirement specifications. In: ICACCI 2016, pp. 620–626 (2016)
25. Dhingra, S., Savithri, G., Madan, M., Manjula, R.: Selection of prioritization technique for software requirement using fuzzy logic and decision tree. In: IC-GET, pp. 1–11 (2016)
26. Kukreja, N., Payyavula, S.S., Boehm, B., Padmanabhuni, S.: Value-based requirements prioritization: usage experiences. Procedia Comput. Sci. **16**, 806–813 (2013)
27. Padmanabhuni, S.: Selecting an appropriate framework for value-based requirements prioritization. In: Proceedings of the 2012 IEEE 20th RE, pp. 303–308 (2012)
28. Dabbagh, M., Lee, S.P.: A consistent approach for prioritizing system quality attributes. In: 14th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing, pp. 317–322
29. Dabbagh, M., Lee, S.P.: An approach for integrating the prioritization of functional and nonfunctional requirements. Scientific World Journal (2014)
30. Bukhsh, F.A., Bukhsh, Z.A., Daneva, M.: A systematic literature review on requirement prioritization techniques and their empirical evaluation. Comput. Stand. Interfaces **69**, 103389 (2020)
31. Alsaqaf, W., Daneva, M., Wieringa, R.: Quality requirements in large-scale distributed agile projects – a systematic literature review. In: Grünbacher, P., Perini, A. (eds.) REFSQ 2017. LNCS, vol. 10153, pp. 219–234. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-54045-0_17

32. Daneva, M., Herrmann, A., Buglione, L.: Coping with Quality Requirements in Large. Contract-Based Projects. IEEE Softw. **32**(6), 84–91 (2015)
33. Martakis, A., Daneva, M.: Handling requirements dependencies in agile projects: a focus group with agile software development practitioners. In: RCIS 2013, pp. 1–11 (2013)
34. Kassab, M., Ormandjieva, O., Daneva, M.: Scope management of non-functional requirements. Euromicro 2007, pp. 409–417 (2007)