Turkish Journal of Electrical Engineering and Computer Sciences

Volume 29 | Number 7 Article 11

1-1-2021

CSOP+RP: a novel constraints satisfaction model for requirements prioritizationin large-scale software systems

SOHEIL AFRAZ

HASSAN RASHIDI

NASER MIKAEILVAND

Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2021) 29: 3074 – 3091

doi:10.3906/elk-2102-139

CSOP+RP: a novel constraints satisfaction model for requirements prioritization in large-scale software systems

Soheil Afraz[1] **,Hassan Rashidi**[2,*]**, Nasser Mikaeilvand**[3]
[1]Faculty of Computer and Information Technology Engineering, Qazvin Branch, Islamic Azad University, Qazvin, Iran
[2]Department of Mathematics and Computer Science, Allameh Tabataba'i University, Tehran, Iran
[3]Department of Mathematics, Ardabil Branch, Islamic Azad University, Ardabil, Iran

Received: 28.02.2021 • **Accepted/Published Online:** 26.07.2021 • **Final Version:** 30.11.2021

Abstract: One of the main factors in the failure of software projects is the lack of attention to their requirements prioritization. In this paper, we propose a decision-oriented methodology with a novel model for requirements prioritization (RP) in large-scale software systems. The model is formulated based on the constraint satisfaction optimization problems (CSOP) approach, which we call CSOP+RP. The main objective of the model is to maximize the quality of the software in total, subject to the constraints on the budgets and importance level that pre-determined by the administrator. To evaluate CSOP+RP, we applied it to the police command-and-control system (PCCS), which is extensively used during the outbreak of the Coronavirus disease as an incident in terms of quality and speed of service. The results of various experiments show that the proposed model with its specific capabilities can find reasonable and a solution near optimal. Moreover, the sensitivity analysis indicates that the model is very sensitive to its parameters. Although, we applied the model to CCPS, the CSOP+RP is very general so that it can be applied to different types of software projects. Additionally, the model could be extended to other aspects and criteria of the RP problem and could play definite roles for optimal management of system resources.

Key words: Constraints satisfaction optimization problem (CSOP), decision-oriented methodology, general-model, requirements prioritization, ultra-large-scale system (ULSS)

1. Introduction

The best way to produce high-quality software is to have a thorough and correct understanding of the requirements, categorize stakeholders to accurately elicit the requirements, and then prioritize them. The requirement prioritization (RP) is accomplished after requirements elicitation. It is one of the principles of software system design and a complex multi-criteria decision-making process. The RP issues can be handled by multi-objective optimization (MOP) and constraint satisfaction problem (CSP) approaches. Many system stakeholders apply the RP methods for selected requirements.

Nowadays, the lack of attention to the requirements of software projects and their defective extraction is considered as one of the main factors in the failure of software projects. It is noteworthy that, the main reason for 56% of all defects in software projects refers to the requirements definition and analysis issues. Half of it is due to the weakness in writing the requirements and the existence of ambiguous or incorrect requirements, and the other half raises from the requirements specification problems [1]. Thus, one of the most vital phases in a successful and

*Correspondence: hrashi@gmail.com

high-quality software project is requirements engineering. Requirements engineering is a set of umbrella activities that cover the entire software development cycle. There are three important phases in requirements engineering: understanding, development, and management. The main activities in requirement engineering focus on the requirements development phase. They include elicitation, analysis, modeling, negotiation, specifications, and verification and validation. Therefore, focusing on this phase leads to extensible structures and user-friendly software, particularly by stakeholders. Accordingly, these activities lead to enhancement in software productivity and life-cycle [2].

Furthermore, changing the scales, increasing complexity, and possible heterogeneity of the components in ultra-large-scale software (ULSS) systems [3] have led to inefficiency of the existing software engineering approaches in terms of design, implementation, and testing.

The ULSS systems are complex software systems that in terms of scale and size beyond today's systems (such as the lines of code; people involved in the system; data stored, retrieved, manipulated and refined; the number of connections and interdependence of software components; hardware elements). These systems include thousands of platforms, sensors, and decision nodes that are connected to each other via wired and wireless heterogeneous networks. The characteristics of these systems have called into question the capability of current software engineering approaches such as the requirements engineering phase in building future systems that include billions of program code lines.

One of the characteristics of ULSS is the multiplicity of stakeholders, the inherent conflicting and unknown requirements. Meanwhile, many of these systems are constructed only by identifying and prioritizing the core and key requirements. Therefore, it is necessary to design and implement high-quality and integrated software systems on a large scale. Scalability is one of the major issues in software engineering and various solutions have been proposed to overcome it. Meanwhile, in this work, decision-making methods are considered as the primary approach to tackle the multi-criteria ranking issue in RP.

This paper provides a general model for requirements prioritization in ULSS systems. The general model optimally satisfies the expectations of a variety of stakeholders and software users. The proposed model aims to prioritize the core requirements in ULSS systems and increases the satisfaction level of stakeholder constraints. Meanwhile, it increases the life-cycle of different versions of a comprehensive system and their integration. Accordingly, the CSP approach is applied to dissolve the prioritization problem as well as the scalability issues in the ULSS.

The rest of the paper is organized as follows. In the next section, a review of the literature is provided. The decision-oriented methodology for prioritization of the requirements based on the collaboration of decision-making units (DMU) is described in Section 3. The proposed general model for prioritization is presented in Section 4 under methodology. Evaluation of the CSOP+RP based on the criteria and scenarios assumed by the police command-and-control system (PCCS) is discussed in Section 5. Finally, the summary and future works are presented in Section 6.

2. Literature review

Nowadays, a substantial number of empirical studies have been done into the RP and its artifacts [4]. Also, many requirements prioritization methods have been developed based on various parameters; among which value and cost are most prominent. Based on stakeholders' perspectives, the RP techniques are divided into two approaches: negotiation and method-oriented. The negotiation-oriented approaches eschew rework and additional costs. The method-oriented approaches are categorized into two subgroups of fundamental

and modern methods, which depend on how the requirements are processed. The fundamental methods in prioritization of requirements such as AHP[1], CV[2], NA[3], PG[4], WM[5], Triage, and some other techniques are addressed in [5]. More modern methods include methods that combine fundamental methods with the influence of different mathematical domains such as fuzzy logic, genetic algorithm, and probability theory. These approaches are proposed and implemented based on mathematical (especially, tensor decomposition [6]), data mining and machine learning, and search-based approaches.

The RP problem is known as a multi-criteria decision-making problem, and various solutions are proposed to solve this problem. The Ruby and Balkishan [7], made an incomplete and preliminary comparison on some methods and proposed fuzzy logic for RP but did not specify how fuzzy logic was applied. Dabbagh et al. carried out a separate survey of FRs (functional requirements) and NFRs (nonfunctional requirements) with an IPA (integrated prioritization approach) and a HAM (hybrid assessment method) and then compared them with AHP-based approaches [8]. FRs are the features of the system-to-be, whereas NFRs describe its quality attributes. NFRs affect the system as a whole and interact both with each other and with the functional requirements. In recent years, some researchers have presented SLRs (systematic literature review) on techniques and challenges of prioritizing requirements [9, 10]. The extensive NA method was proposed in [11], which was a combination of NA and PG with weights for the stakeholders. In addition, the D-Rank has provided a semi-automated RP method for ranking the requirements, which extracts the dependencies based on i* model [12].

The studies in [10, 13] evaluated RP techniques proposed between 2007 and 2019. New RP methods tend to use fuzzy logic and machine learning algorithms, especially in scalability issues. Ahuja et al. [14] presented a new technique using the least-squares-based random genetic algorithm for enhancement performance of RP. Their research was able to reduce time and decision-making efforts.

3. Decision-oriented methodology

The prioritization of ULSS requirements is an important challenge in requirement engineering. This section outlines the proposed decision-oriented methodology for RP to find the optimal requirements priority list in the ULSS using the CSOP+RP model.

A finite set of requirement pool including FR and NFR such that FR = {FR_1, FR_2. . .. . .., FRn} and NFR = {NFR_1, NFR_2. . .. . .., NFR_m} is considered for prioritization. Moreover, there is a set of DMUs, such that DMU = {DMU_1, DMU_2. . .. . .DMU_k}, plus a system administrator (Admin). The DMUs negotiate about current system requirements with the Admin and provide model inputs to the prioritization process with a collaborative approach. Certainly, a more appropriate decision can be made using the developed model based on decision makers' evaluation and comparison. The whole scheme of the designed methodology and the requirements prioritization process is summarized in Figure 1. The stages of this methodology and the prioritization process are briefly stated below.

- **Stage-1**: Firstly, the admin designates/candidates the current system requirements from the requirements pool based on attribute-driven design (ADD). The ADD method is a systematic step-by-step method for

[1]analytic hierarchy process (AHP)
[2]cumulative voting (CV)
[3]numerical assignment (NA)
[4]priority groups (PG)
[5]Weiger's method (WM)

designing the software architecture of a software-intensive system. It is an approach to defining software architectures by basing the design process on the architecture's quality attribute requirements [15].

- **Stage-2**: The candidate requirements are registered for the prioritization process by the admin.
- **Stage-3**: The admin and DMUs negotiate about candidate requirements and then finalize them.
- **Stage-4**: The DMUs and admin specify the relative importance and implementation cost of the requirements as the general model inputs.
- **Stage-5**: The admin has great expertise with the domain of the current software system. He/she weighs the DMUs views based on their expertise and then averages the weights.
- **Stage-6**: Construction of a customized model for the current system by the admin which includes the parameter setting and defining the constraints and function.
- **Stage-7**: The model is run by the admin based on the acquired inputs as well as the predefined constraints and objective function.
- **Stage-8**: Obtaining the final priority list for the current system requirements.
- **Stage-9**: Finally, the admin analyzes the resulting priority list and adopts the best managerial decisions.

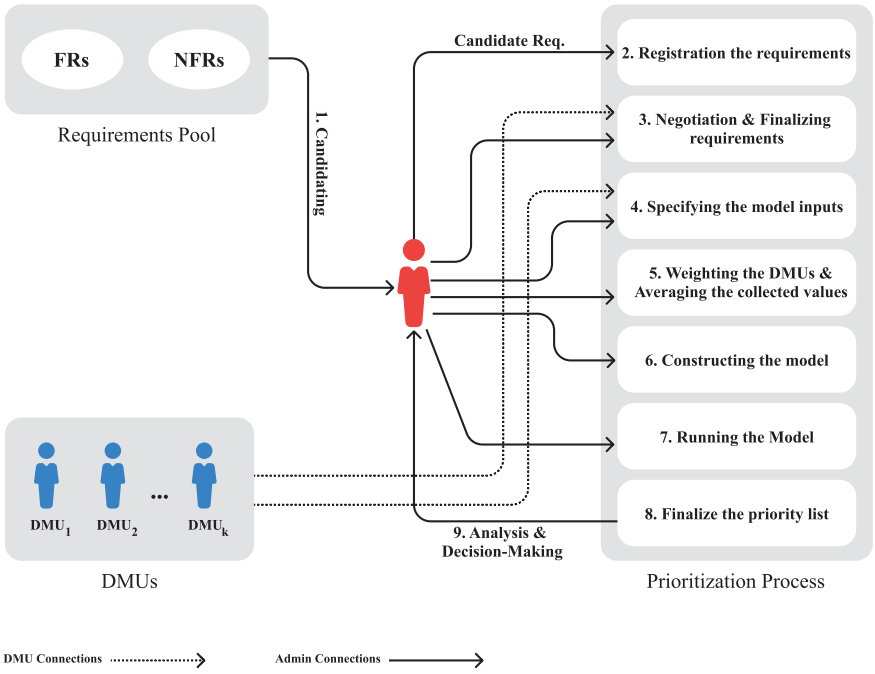

Figure 1. Overview of the proposed methodology for RP.

4. The proposed model

We propose a general mathematical model to support a variety of software systems. In fact, it is a linear integer model, and the data are static at the highest level because the software implementation and prioritization are mostly based on static data. With regard to the decision-oriented methodology defined in the previous section, the aims of designing the model are increasing the satisfaction levels of the stakeholders and the accuracy of the RP process in the ULSS. Accordingly, this model can reduce the system failure rates, and then rank requirements in terms of different views of stakeholders to meet realistic expectations of the system requirements.

In this model, the requirements are divided into two types, which are functional requirements (FR) and nonfunctional requirements (NFR) (see requirements pool in Figure 1). Also, sometimes a requirement can belong to more than one category at the same time (for example, business requirements). The requirements have specific characteristics that we describe as criteria. Some of these criteria such as cost, risk, and benefit determine the boundary for certain functionality, which must be minimized or maximized [16, 17]. Notably, the requirements themselves influence each other. This means that given the nature of some requirements, specifically the NFRs, their existence or absence can lead to conflicts or resolving the conflicts of the system requirements [18].

4.1. Assumptions

Some early research on the prioritization of requirements was based on simplified assumptions. These assumptions are known as limitations on the RP [19]. The following assumptions are considered in the proposed model.

Assumption-1 The system consists of several subsystems. For each subsystem, the FRs and NFRs are independent.

Assumption-2 The DMUs implicitly consider the influence of all NFRs (as the quality attributes) on all FRs.

Assumption-3 All of the input data, including the requirements weight (importance), costs, and budgets are decimal and greater than or equal to zero.

Assumption-4 Based on the domain of the software system, the number of FRs and NFRs in each subsystem is determined by the system administrator and DMUs.

Assumption-5 The DMUs are elected by the admin. Moreover, the number of DMUs in the proposed model varies depending on the type of software system, and the average value of DMU viewpoints is taken into account.

Assumption-6 For each subsystem, the unit-costs for leveling the NFR on any FR are definable.

Assumption-7 For each subsystem, the weight of the significant level of one NFR on the other FR is determined by DMUs.

Assumption-8 The nominal scale between the five levels of importance defined for the requirements (Very low = 1, Low = 2, Medium = 3, High = 4, Very high = 5) is equal.

Assumption-9 All the constraints defined by the system administrator must be satisfied.

Assumption-10 The intermediate and output values obtained from solving the model are nominal.

Assumption-11 Each of the decision variables and weight of significant level (see Assumption-7) has an upper-bound (UB) and lower-bound (LB), which are determined by the system administrator.

Assumption-12 Along with the identified subsystems, the main goal of the model is to maximize the level of the importance or quality of the software in total.

It is also worth noting that, after elicitation and selecting the system requirements in the requirements engineering process, prioritizing the activities is carried out to rank them according to the order of importance and subsequent implementable versions. Consequently, this is a big step in making important decisions to enhance the economic value and quality of a software system.

4.2. Problem formulation

Here, we formulated the problem based on the CSOP approach [20]. The CSOP is rarely used to solve software engineering problems to optimality. There are two reasons for choosing to represent and formulate the problem as CSOPs rather than mathematical programming. Firstly, representation as CSOP is often much closer to the original problem, which means that the variables of the CSOP directly correspond to problem entities, and the constraints can be expressed without having to be translated into linear inequalities. This makes the formulation simpler, the solution easier to understand, and the choice of good heuristics to guide the solution strategy more straightforward in a very large search space. Secondly, although CSOP algorithms are essentially very simple, they can sometimes find a solution more quickly than integer programming methods. Moreover, given the generality of the CSOP approach and its potential to model various decision-making problems in particular problems that have solutions in discrete space, we were persuaded to use the CSOP approach to find the optimal solution or a solution near optimal to the RP problem.

In the CSOP approach, other than a solution that satisfies all of the constraints, we need to find the optimal solution or a solution near optimal to the RP problem. There are four elements in the CSOP: decision variables, the domain of variables, constraints, and the objective function (s) [21]. In the following sections, we will further describe these four elements of the model. Furthermore, the selection of several quality attributes (QA) is significant in ULSS as a set of the model's input. The most important QA for ULSS systems is described in [18].

4.2.1. Parameters required for the decision process

According to the assumptions above, before solving the proposed model and subsequent final decision making, the following parameters must be specified at the beginning of the RP process:

N The number of functional requirements (FRs)

M The number of nonfunctional requirements (NFRs)

K The number of decision-making units (DMUs)

BFR_j The total budgets assigned to a FR_j per man-hours, j = 1, 2,…,N

$BNFR_i$ The total budgets assigned to a NFR_i per man-hours, i = 1, 2,…,M

$W_{i,j}$ The weight of significant level of the NFR_i on the FR_j , i = 1, 2,…,M; j = 1, 2,…,N ($LB \leq W_{i,j} \leq UB$)

$C_{i,j}$ The unit-cost of implementation of the NFR_i on the FR_j , i = 1, 2,…,M; j = 1, 2,…,N

The matrix shown in Table 1 and Table 2 is used to obtain the $C_{i,j}$ and $W_{i,j}$ parameters. The last row of Table 1 contains the total budget of FR_i (BFR_j), and the last column shows the total allocated budget for NFR_i ($BNFR_i$). For example, the value of BFR1 is the total budget of FR1 on which we want all NFRs to run. Moreover, other cells of Table 1 include the unit-cost for the implementation of an NFR_i on an FR_j. Similarly, in Table 2, another matrix is considered for the $W_{i,j}$ parameter, which has the average weight of all K DMU views. Each cell in this matrix represents a weight of a significant level of the NFR_i on the FR_j.

To avoid the complexity in the proposed general model, we have considered the most important parameters: the importance and budget parameters from the DMU point of view. They are optimized according to the constraints to determine the required priorities and decision variables. This means that the average point of view of decision-makers within the proposed system is considered in the proposed approach.

Table 1. Unit-cost of implementation of the NFR_i in the FR_j.

	FR_1	FR_2	FR_3	...	FR_M	$BNFR_i$
NFR_1						$BNFR_1$
NFR_2						$BNFR_2$
NFR_3						$BNFR_3$
⋮						⋮
NFR_N						$BNFR_M$
BFR_j	BFR_1	BFR_2	BFR_3	...	BFR_N	

Table 2. Weight of significant level of the NFR_i in the FR_j.

	FR_1	FR_2	FR_3	...	FR_M
NFR_1					
NFR_2					
NFR_3					
⋮					
NFR_N					

4.2.2. Decision variables and domains

The decision variables in this model are to determine the priority of the NFR_i on the FR_j. Based on the assumption-11, each of these variables is denoted by X_{ij} whose possible discrete domain is positive integers between LB_{ij} and UB_{ij} defined by DMU. Since there are N and M requirements for FR and NFR in software projects respectively, we have $M \times N$ decision variables that their values will be determined by solving the model. The priority level of the NFRs on the FRs can be represented in a matrix form as shown in Table 3. They are analyzed by the system administrator for decision-making and the optimal management of software system resources. Also, each cell in Table 3 can be limited to domains defined by the system administrator. For example, scalability as a quality attribute is crucial in web systems. So, the scalability requirement of the software system can be set between 3 and 5 by the system administrator. On the other hand, each of the FRs domains can be limited to a given range on all NFRs in a particular software project.

4.2.3. Objective function

In this model, the objective function is to maximize the Z value (see Assumption-12), according to Equation (1). It is the sum of the product of the weight for each NFR on each FR and the priority level of each corresponding decision variable. So, the z value is calculated by Equation (1), that is the sum of the weights multiplied by the decision variables:

$$Max\ Z = \sum_{i=1}^{M} \sum_{j=1}^{N} W_{i.j} \times X_{i.j,} \tag{1}$$

Table 3. The decision variables matrix.

	FR_1	FR_2	FR_3	...	FR_M
NFR_1					
NFR_2					
NFR_3					
⋮					
NFR_N					

where X_{ij} is the priority of the NFR_i that influences the FR_j.

4.2.4. Constraints

There are four categories of constraints in the CSOP+RP model. The constraints in the categories (2) and (3) are applied to limit the defined budgets for functional requirements (BFR) and nonfunctional requirements (BNFR). In fact, both constraints are used to control the FRs and NFRs implementation cost. Also, there is a constraint for controlling the upper-bound (UB) and lower-bound (LB) of the decision variables, where usually UB = 5 and LB = 1 (See the constraints in the category (4) and Assumption-11). Moreover, the decision variables must have integer values, which are given in category (5) of the constraints.

$$\sum_{i=1}^{M} C_{i.j} \times X_{i.j} \leq BFR_j \quad \text{for} \quad j = 1, 2 \ldots N. \tag{2}$$

$$\sum_{j=1}^{N} C_{i.j} \times X_{i.j} \leq BNFR_i \quad \text{for} \quad i = 1, 2 \ldots M. \tag{3}$$

$$LB_{i.j} \leq X_{i.j} \leq UB_{i.j}, \quad \text{for } i = 1, 2 \ldots M;\ j = 1, 2 \ldots N. \tag{4}$$

$$X_{i.j} \text{ is Integer for } i = 1, 2 \ldots N;\ j = 1, 2, \ldots N. \tag{5}$$

Totally, there are $(M + N) + (M \times N)$ constraints in the model.

5. Evaluation (Control experiment)

In this section, we evaluate the proposed general model on a case study, known as police command-and-control system (PCCS). Although we could focus on different case studies, we worked on this system because it is one of the most important needs and basic applications during the Coronavirus pandemic in many countries. Different versions of PCCS have been developed and are being used. Due to the importance of re-evaluating and improving the system performance, specially NFRs, some military and medical system managers as stakeholders have decided to optimize the prioritization of FRs and NFRs. They aim to allocate an appropriate budget so that the total satisfiability and quality of the whole system are maximized. In the following, we provide more details of PCCS and its adaptation as a ULSS, which proves that the optimal prioritization of requirements can play an important role in human life. In particular, during the outbreak of the Coronavirus disease (COVID-19) the PCCS could increase the quality and speed of service to different people in the community as system users [22]. Nowadays, the operational approaches of the COVID-19 pandemic are very vital in human life and software

system development so, must be accurately analyzed and prioritize the requirements to increase performance and user satisfaction.

5.1. Case study (PCCS)

Basically, a mini-requirement for PCCS is briefly described in [23] and then the system is expanded in [24–26]. This police service system must respond as quickly as possible to many reported incidents (for example, people suspicious of contracting Coronavirus, who has contacted PCCS). Its main goals are to ensure that incidents are recorded and routed to the most appropriate police vehicle. Due to PCCS fertility for reusability in both application and system software, we selected PCCS in our study. The FRs and NFRs of PCCS are depicted in Table 4 and Table 5. Also, the full specification of the system and its implementation are given in [23]. Although the PCCS system has already been developed, to enhancement the importance of some requirements and system services, we need to revise and redesign the requirement engineering process. Apart from 13 FRs identified in Table 4, the selection of the 10 NFRs (see Table 5) for PCCS is based on the presented catalog in the type of information system [18] and ULSS characteristics (Stage-1 to Stage-3 of methodology).

To adapt the PCCS system to a large scale, it clear that the selection and adaptation of PCCS systems to ULSS systems is important in two main aspects of adaptability and scalability. The first aspect is the compliance of the PCCS system requirements with the key characteristics of requirements on ULSS. For instance, the conflict on some requirements is due to the existence of multiple stakeholders (diversity) and ever-changing system requirements, which from the perspective of the end-users, the system must have a quick response and be available to the efficient. Although for system managers, security and reliability are considered critical. The second aspect is the inherent scalability of the PCCS system, which can be developed both horizontally (scale-in) and vertically (scale-out) to be defined as the ULSS system. In detail, vertical scaling is like changing the FR and NFR requirements of system components in order to increase efficiency, and horizontal scaling is like increasing the number of users and external entities of the system, which shows the necessity of extensibility and increasing interoperability between PCCS system components [27].

Table 4. The subsystem of PCCS with its FRs.

Subsystems	Notation	FR requirement (Abbreviation.)
Registration	FR_1	Call taking (CT)
	FR_2	Incident registration (IR)
Decision making	FR_3	Create response (CR)
	FR_4	Find closest unit (FU)
	FR_5	Alert emergency service (ES)
	FR_6	Get position of units (GP)
Dispatching	FR_7	Dispatch units (DU)
	FR_8	Send data (SD)
	FR_9	Response to incident (RI)
	FR_{10}	Incident/unit management (IM)
	FR_{11}	Request more units (RU)
Reporting	FR_{12}	Sending report (SR)
	FR_{13}	Closing incident (CI)

Table 5. NFRs in the PCCS system.

Notation	NFR requirement (Abbreviation.)
NFR_1	Security (SY)
NFR_2	Performance (PF)
NFR_3	Availability (AV)
NFR_4	Accessibility (AC)
NFR_5	Usability (US)
NFR_6	Reliability (RL)
NFR_7	Maintainability (MT)
NFR_8	Interoperability (IO)
NFR_9	Scalability (SC)
NFR_{10}	Portability (PO)

5.2. Solving the model

By solving the model, the priorities of the requirements will be determined (Stage-6 of the methodology). As shown before, different solution methods can be developed and implemented on different kinds of systems and programming languages. To solve the model, we focus on the spreadsheet and its extensions. The main reason for using spreadsheets includes extensive statistics, forecasting, modeling tools, and database capabilities.

Among the Add-In packages, many are built for decision support system (DSS) development. These DSS add-ins include Solver and What'sBest for linear and nonlinear optimizations. As a solution, we employed the Solver tool in Excel software in which the gradient-based methods is used to improve the current solution based on calculating the slope of the target function numerically.The Solver is a powerful tool for optimization of linear and nonlinear problems [28].In this tool, the generalized reduced gradient (GRG) algorithm is used [29], which can be an effective algorithm to linear and nonlinear smooth problems and can find a solution close to the optimal value with a slight difference.

5.3. Results and discussion

As mentioned in Section 5.1, we need to design suitable experiments for PCCS to evaluate the proposed model. We prepared a questionnaire that contains a general description of the PCCS system and its requirement specifications along with $W_{i,j}$ and $C_{i,j}$ input tables to the system stakeholders and DMU's. Then system stakeholders, including experts, requirement engineers, system designers, and software architects completed the questionnaire and submitted their comments to the system administrator considering the limitations defined on the values. The system administrator then controls the values of the questionnaires then averages and applies them as the technical coefficients of the problem in the proposed model (Stage-5 of the methodology). Finally, based on the resulting outputs, the system administrator performs its analyses and makes its managerial decisions. Of course, it is important to note that the administrator of the system can remove some of the requirements out of the prioritization process based on the proposed model depending on the inherent priority of some FRs or NFRs in the system implementation.

Similar to Table 1 and Table 2 in the PCCS case study, we have two sets of quantitative input data, as shown in Table 6 and Table 7. They show the average weight of significant level and the unit-cost of implementing of K DMU (K = 5 in PCCS) views in each NFR_i on FR_j, respectively (Stage-4 and Stage-5 of the methodology).

Many PCCS experiments have been carried out under different BNFR, BFR, and boundary parameters as well as the $W_{i,j}$ and $C_{i,j}$ values of the base configuration to obtain the desired results (Stage-6 and Stage-7 of the methodology). Table 8 shows the value of the objective function obtained from solving the model for 27 different experiments based on the basic configuration. All experiments are solved by GRG solver under the value $\varepsilon = 0.0001$ in a PC with an Intel Core i3 processor and 4 GB of RAM.

According to Table 8, under certain values for the parameters in the 5 experiments, the algorithm could not find any feasible solution (NFS =not feasible solution). Moreover, based on the objective function defined in the subsection 4.2.3, the purpose of solving the problem is to prioritize the requirements and maximize the productivity of the software system. Hence, the results obtained from the model solution and the required computation to determine the importance of the four subsystem requirements are defined in Table 8. After solving the model and finding the optimal or a solution near optimal, the final priority list containing values of the decision variables is obtained and shown in Table 9 (decision matrix), which indicates the importance level of requirements (Stage-8 of the methodology). In conclusion, we have tested all possible scenarios to find the

solution on a small scale. Then, with Excel software, we obtained the optimal results, which showed that the results were the same. Finally, we have generalized the work on a large scale.

Table 6. The average weight of significant level of the NFR_i on the FR_j based on the views of five DMUs (K = 5).

	FR_1	FR_2	FR_3	FR_4	FR_5	FR_6	FR_7	FR_8	FR_9	FR_{10}	FR_{11}	FR_{12}	FR_{13}
NFR_1	4	4	2	2	3	3	4	5	4	4	3	5	4
NFR_2	4	5	4	5	3	5	5	4	4	5	4	5	4
NFR_3	3	3	4	3	2	3	3	4	3	2	2	3	2
NFR_4	5	2	2	2	2	3	3	3	4	3	4	4	2
NFR_5	3	3	2	5	2	2	4	3	2	2	3	4	2
NFR_6	3	4	3	3	2	2	3	3	3	1	3	3	2
NFR_7	2	3	2	1	1	1	2	3	3	2	2	1	2
NFR_8	1	2	1	4	4	1	1	1	2	1	4	2	1
NFR_9	2	4	4	3	4	2	3	3	2	2	3	1	1
NFR_{10}	2	1	2	2	3	1	1	2	1	1	3	2	1

Table 7. The unit-cost of implementation of the NFR_i on the FR_j based on K DMU views.

	FR_1	FR_2	FR_3	FR_4	FR_5	FR_6	FR_7	FR_8	FR_9	FR_{10}	FR_{11}	FR_{12}	FR_{13}
NFR_1	5	4	2	3	2	2	3	4	3	2	2	3	4
NFR_2	4	4	5	5	4	5	4	5	4	4	3	4	4
NFR_3	3	2	2	2	2	3	2	2	2	2	2	3	2
NFR_4	2	2	3	3	2	3	3	2	3	2	2	3	3
NFR_5	2	2	3	2	2	3	3	2	3	2	2	2	3
NFR_6	2	3	2	2	2	3	2	4	3	2	2	4	3
NFR_7	2	3	2	1	1	1	2	3	3	2	2	1	2
NFR_8	2	1	2	2	1	2	2	3	2	3	2	3	2
NFR_9	4	3	2	3	3	2	3	2	3	2	3	3	2
NFR_{10}	3	2	1	1	2	3	2	2	3	2	2	3	1

Figure 2 shows the results of the model solution for the requirements of registration (Reg.), decision-making (DM.), dispatching (Disp.), and reporting (Rep.) subsystems. In general, the importance and priority of all system requirements are shown in Figure 3.

We can obtain the following observations from Figure 2.

- **Observation-1:** In the Reg. subsystem (Figure 2 (a)), the highest importance for FR_1 (CT) is based on the implementation of NFR_4 (AC) and NFR_2 (PF) with values of 5 and 3, respectively. Also, the highest priority for FR_2 (IR) requirements is the implementation of NFR_7 (MT) and NFR_5 (US) requirements, which are 5 and 4, respectively.
- **Observation-2:** In the DM subsystem (Figure 2 (b)), implementation of NFR_5 (US), NFR_8 (IO), and NFR_{10} (PO) requirements on FR_4 (FU) as the FR has the highest priority. Except for NFR_1 (SY), the implementation of other NFRs on FR_6 (GP) has minimum importance.

Table 8. The unit results of solving the model for different experiments.

Experiment	Budget allocated to FR_j (BFR_j for j = 1, 2 …N)	Total budget unused for all FR (man-hours)	Budget allocated to NFR_i ($BNFR_i$ for i = 1, 2 …M)	Total budget unused for all NFR (man-hours)	Objective function value
1	40	2	65	132	671
2	40	4	65	134	632
3	44 (+10%)	15	65	93	730
4	48 (+20%)	8	65	34	802
5	36 (-10%)	2	65	184	591
6	32 (-20%)	2	65	236	501
7	40	3	71 (+10%)	193	671
8	40	3	77 (+20%)	253	671
9	40	-	48 (-10%)	-	NFS
10	40	-	41(-20%)	-	NFS
11	44 (+10%)	1	71 (+10%)	139	744
12	48 (+20%)	9	71 (+20%)	155	810
13	30	4	[65,65,65,55,55,45,40, 40,55,55]	154	444
14	30	20	All=4 except NFR2=55	45	407
15	30	8	[39,55,33,39,35,36,33, 37,37,40]	2	440
16	32	32	[39,55,33,39,35,36,33, 37,37,40]	0	452
17	33	45	[39,55,33,39,35,36,33, 37,37,40]	0	453
18	31	-	[39,55,33,39,35,36,33, 37,37,40]	-	NFS
19	[30,30,29,32,29,30,29, 31,32,22,32,32,25]	8	[39,55,33,39,35,36,33, 37,37,39]	8	423
20	[30,30,26,30,25,30,27, 27,30,22,31,30,25]	0	[39,55,29,37,33,34,31, 34,35,36]	0	414
21	30	-	40	-	NFS
22	32	-	40	-	NFS
23	40	4	65	199	677
24	40	5	65	95	716
25	40	3	65	172	612
26	40	8	65	138	681
27	60	185	60	5	779

Notes: In Table 8, all of the single numbers in the BFR_j and $BNFR_i$ columns are given based on the same values for all requirements. Also, the values of these parameters for experiments 13 to 20 are specified separately in brackets. On the other hand, except for Experiment 2, where the values of the decision variables are between values 1 and 3 ($1 <= X_{ij} <= 3$), the values of the decision variables in the other experiments are between 1 and 5 ($1 <= X_{ij} <= 5$). Furthermore, the number of FRs and NFRs is the same in all experiments except rows 23 (N = 13, M = 11), 24 (N = 14, M = 10), 25 (N = 12, M = 10), and 26 (N = 13, M = 9), which are intended for sensitivity analysis with different values. The symbol "+" and "- "show the increase and decrease of budget allocated to FRs and NFRs relative to basic configuration (experiment 1) respectively.

Table 9. The decision matrix for basic configuration (final results of the proposed model).

	FR_1	FR_2	FR_3	FR_4	FR_5	FR_6	FR_7	FR_8	FR_9	FR_{10}	FR_{11}	FR_{12}	FR_{13}
NFR_1	2	2	2	1	4	5	1	2	1	5	1	2	1
NFR_2	3	2	1	3	2	1	2	2	2	1	2	2	3
NFR_3	1	3	4	1	1	1	3	5	3	1	1	1	4
NFR_4	5	1	1	2	2	1	2	1	1	5	4	1	1
NFR_5	1	4	1	5	1	2	1	2	1	1	2	5	1
NFR_6	2	1	1	2	1	1	5	1	2	1	1	1	2
NFR_7	1	5	2	1	2	1	1	1	4	2	1	1	2
NFR_8	2	1	1	5	1	1	1	1	1	1	2	2	2
NFR_9	2	1	5	1	2	2	1	2	1	2	1	1	1
NFR_{10}	1	1	4	5	2	1	1	1	1	1	5	1	3

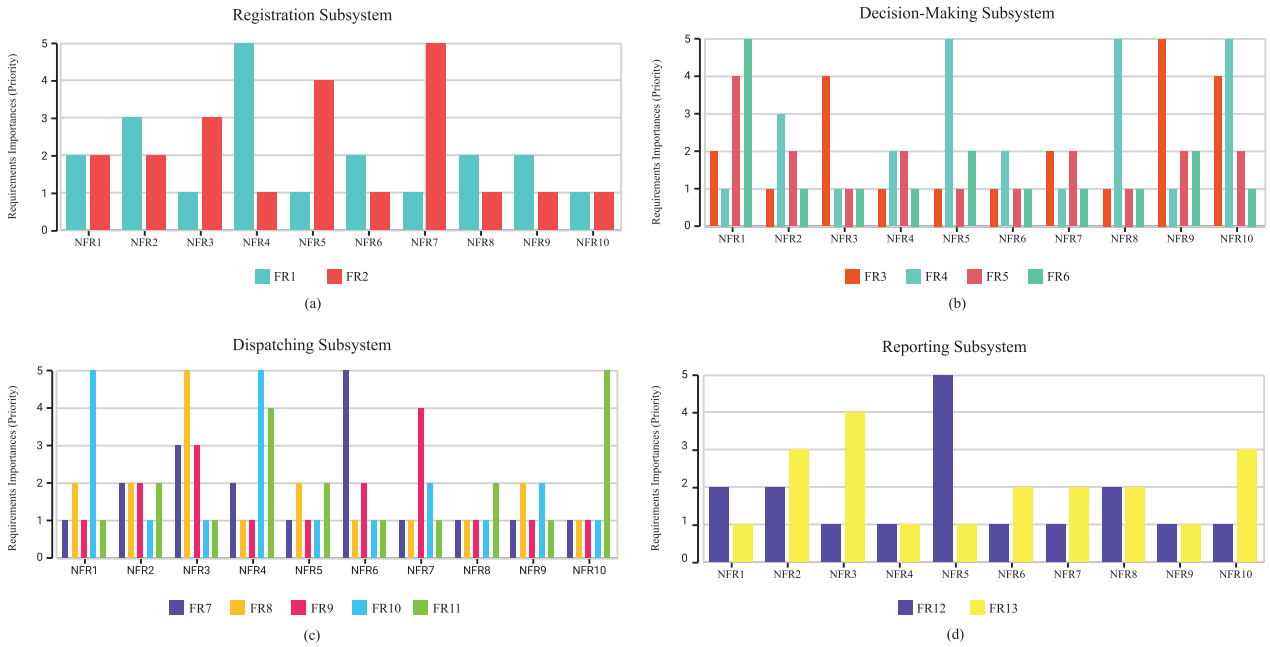

Figure 2. Requirements priorities in PCCS subsystems: (a) Registration subsystem (b) Decision-making subsystem (c) Dispatching subsystem (d) Reporting subsystem.

- **Observation-3:** In the Disp. subsystem (Figure 2 (c)), due to the allocated budget, implementing NFR_1 and NFR_4 requirements on FR_{10}, has the highest priority. Also, the most important NFR for FR_{11} is only NFR10 with a value of 5, and the implementation of NFR_2 has a medium priority for all FRs of this subsystem.
- **Observation-4:** In the Rep. subsystem (Figure 2 (d)), relatively, implementation of the majority of NFRs on FR_{12} has a low priority. However, this does not include NFR_5 with the highest possible value. Also, about the FR_{13}, the most important quality attributes can be NFR_3, NFR_2, and NFR_{10} with values of 4, 3, and 3, respectively.

- **Observation-5:**: In general, according to the calculation of the total relative weight of FRs and NFRs, the FR with the highest priority is FR_4 (FU) from the subsystem DM with a weight range of 26 in the PCCS system (Figure 3). Also, based on rational expectations of the performance of the prioritization process, the most important NFRs in this basic configuration are NFR_1 (PF) with a total weight of 29.

  It is important to note that among the various experiments, the best result was obtained for the value of the objective function in the basic configuration (values of Table 6 and Table 7) under the BFR and BNFR parameters. Thus, the value of the objective function under the constraints considered for the decision variables listed in Table 8 is 414. Given that the minimum unused amount budget of requirements is obtained and the amount of unused of both budgets allocated in it is zero (Experiment 20 of Table 8), the value of the objective function is optimal. Consequently, these observations can help the requirements engineering team and system designers to schedule the project and increase the quality of performance and efficiency of the system.

  In this study, we have done the optimization work. This optimizes the prioritization of FRs and NFRs, and naturally, the result of optimization creates higher value for the objective function that can be satisfactory at the macro level. Moreover, sensitivity analysis will determine the sensitivity of the solution to the parameters from different aspects.

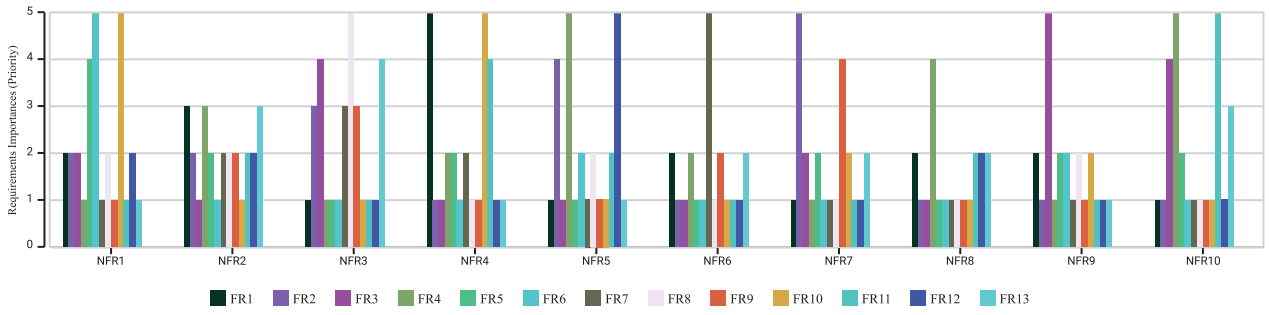

Figure 3. Requirements priorities in the total system of PCCS.

5.4. Sensitivity analysis

After solving the model and obtaining the above observations, sensitivity analysis is performed (Stage-9 of methodology). Sensitivity analysis conceptually determines the sensitivity of decision variables and the objective function value to the changes in parameters. Due to the lack of automatic sensitivity analysis in the GRG algorithm, this is done by trial and error. To this end, the average values of the input parameters of the model (from K DMU) are determined and the model is executed by the system administrator (Stage-5 and Stage-6 of methodology). These parameters include $W_{i,,j}$, $C_{i,,j}$, and the allocated budget (BFR and BNFR) within the boundaries defined for the decision variables for each of the FRs and NFRs. Here, we have considered all the influential factors, including the number of requirements, the budget allocated to them, as well as the importance of the requirements based on the domain of the proposed system to make the best decisions for the system. This section presents sensitivity analysis and examining the effects of changing the most critical parameters.

Based on the budget situation from Table 10 and Table 11, as the first aspect of sensitivity analysis in this model, we focused on changing the values of budget parameters (BFR and BNFR). In detail, only one unit of the allocated budget to the FR_7 and FR_8 is unused. Hence, the amount of budget consumption can be acceptable to the system administrator. However, for the budget allocated to NFRs, in the worst-case scenario, the NFR_8, NFR_7, and NFR_6 requirements budgets of 28, 26, and 21 units remain unused, respectively. For example, in the worst case, approximately 43% of the budget defined for the NFR_8 quality attribute was not used in this case study, which is unacceptable to the system administrator. Because the system administrator can use this remaining budget for decision-making and optimizing the requirement engineering process and system design. As a result, it will increase the overall efficiency of the system and the level of stakeholder satisfaction with the RP process. The results of this analysis and comparisons of different experiments are illustrated in Figure 4 (a), which shows the direct effect of these changes on the value of the objective function.

The second aspect of sensitivity analysis focuses on tracing changes in the basic configuration elements and examining their effects on the objective function. As shown as in Figure 4 (b), in a specific analysis, according to Experiment-2 in Table 8, the value of the objective functions reaches 632 (OF = 632) by changing the boundaries of the decision variables to 1–3 ($1 <= X_{ij} <= 3$) in the basic configuration. Therefore, a change of about 5% in the value of the objective function will be observed which is ideal for system stakeholders. Overall, the purpose of changes on the number of FR and NFR requirements by N and M parameters respectively is to perform sensitivity analysis on these values. This means that we want to determine the sensitivity in the near-optimal solution of the defined objective function if these values change.

Table 10. The status of the allocated FR budgets in the basic configuration.

BFR_i	BFR_1	BFR_2	BFR_3	BFR_4	BFR_5	BFR_6	BFR_7	BFR_8	BFR_9	BFR_{10}	BFR_{11}	BFR_{12}	BFR_{13}
Allocated BFR_j	40	40	40	40	40	40	40	40	40	40	40	40	40
Used BFR_j	40	40	40	40	40	40	39	39	40	40	40	40	40
Unused BFR_j	0	0	0	0	0	0	1	1	0	0	0	0	0

Table 11. The status of the allocated NFR budgets in the basic configuration.

$BNFR_i$	$BNFR_1$	$BNFR_2$	$BNFR_3$	$BNFR_4$	$BNFR_5$	$BNFR_6$	$BNFR_7$	$BNFR_8$	$BNFR_9$	$BNFR_{10}$
Allocated $BNFR_j$	65	65	65	65	65	65	65	65	65	65
Used $BNFR_j$	61	62	61	55	55	44	39	37	52	52
Unused $BNFR_j$	4	3	4	10	10	21	26	28	13	13

6. Conclusion and future works

The RP is one of the main research concerns in requirements engineering. That is why more effort needs to be made by software engineering researchers in RP issues. It is necessary to provide the optimal or a near optimal solutions and algorithms to prioritize software requirements. In this research, we proposed a decision-oriented methodology with a novel and general mathematical model, based on the CSOP approach. The model is known as "CSOP+RP", which has three main components: the decision variables with their domains, objective function, and constraints. The proposed methodology and model are applied to a ULSS case study, called PCCS, so that the near optimal solutions were found in search space. It is worth mentioning that the proposed general model can be applied to all types of software systems in different domains. Moreover, according to the

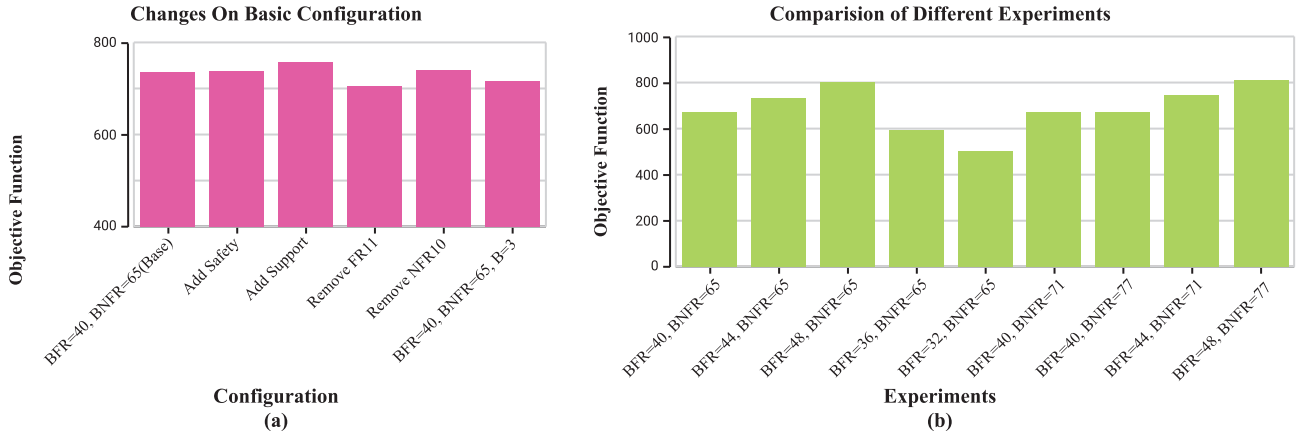

Figure 4. (a) Comparison of different experiments on the basic configuration. (b) Trace of changes in the basic configuration.

results of various experiments for sensitivity analysis, it was found that with separate or simultaneous changes in the values of the problem parameters, reasonable and ideal changes occur in the value of the objective function. Therefore, based on the system administrator's expectations, relative harmony in the values of the decision variables used to prioritize the requirements is significant and observable.

To summarize, this model can help the system administrator to schedule the implementation of requirements based on a set of priorities. Moreover, it can determine the relative importance of subsystems and modules in the next-release products (NRP) and software product-line (SPL). The generality of the proposed model is so great that it can be used on a large scale. At this scale, a large number of decision variables can be identified along with a large number of functional and nonfunctional requirements and the model can be solved for such problems.

Concerning the flexibility of the proposed general model, with the changes in the basic configuration, variations in execution speed and the value of the objective function were observed. The changes include removing and adding requirements that are dependent on the nature of the system and directly handled by the system administrator. In particular, the extensibility features of the general model based on using other essential parameters such as risk and penalty can be considered for future works. Furthermore, the fuzzification of the proposed general model and multi- objective optimization algorithms in this model to solve the RP problem can be addressed as future work.

References

[1] Vaz E. Delivering better projects on time by ensuring requirements quality upfront. In International Council on Systems Engineering (INCOSE) International 17 Symposium Wiley 2018; 28 (1): 575-586. doi:10.1002/j.2334-5837.2018. 00501.x

[2] Han J. TRAM: A tool for requirements and architecture management. In: Proceedings 24th Australian Computer Science Communications, IEEE Computer Society, ACSC 2001. pp. 60-68. doi:10.1109/ACSC.2001.906624

[3] Northrop L, Feiler P, Gabriel RP, Goodenough J, Linger R et al. Ultra-large-scale systems: The software challenge of the future. Software Engineering Institute, Carnegie-Mellon University, Pittsburgh ,2006.

[4] Thakurta R. Understanding requirement prioritization artifacts: a systematic mapping study. Requirements Engineering, springer 2017; 22 (4): 491-526. doi:10.1007/s00766-016-0253-7

[5] Lehtola L, Kauppinen M. Suitability of requirements prioritization methods for market-driven software product development. Software Process: Improvement and Practice. Wiley 2006; 11 (1):7-19. doi:10.1002/spip.249

[6] Misaghian N, Motameni H. An approach for requirements prioritization based on tensor decomposition. Requirements Engineering. Springer 2018; 23 (2): 169-88. doi:10.1007/s00766-016-0262-6

[7] Ruby B. Role of fuzzy logic in requirement prioritization. International Journal of Innovative Research in Science, Engineering and Technology. 2015; 4 (6): 4290-4297.

[8] Dabbagh M, Lee SP, Parizi RM. Functional and non-functional requirements prioritization: empirical evaluation of IPA, AHP-based, and HAM-based approaches. Soft computing. Springer 2016 ;20 (11): 4497-4520.doi: 10.1007/s00500-015-1760-z

[9] Hujainah F, Bakar RB, Al-Haimi B, Abdulgabber MA. Stakeholder quantification and prioritisation research: a systematic literature review. Information and Software Technology. 2018; 102: 85-99.doi: 10.1016/j.infsof.2018.05.008

[10] Bukhsh FA, Bukhsh ZA, Daneva M. A systematic literature review on requirement prioritization techniques and their empirical evaluation. Computer Standards & Interfaces. 2020; 69: 103389.doi: 10.1016/j.csi.2019.103389

[11] Yaseen M, Ibrahim N, Mustapha A. Requirements prioritization and using iteration model for successful implementation of requirements. International Journal of Advanced Computer Science and Applications (IJACSA). 2019; 10 (1): 121-127.doi: 10.14569/IJACSA.2019.0100115

[12] Shao F, Peng R, Lai H, Wang B. DRank: A semi-automated requirements prioritization method based on preferences and dependencies. Journal of Systems and Software. 2017; 126: 141-156.doi: 10.1016/j.jss.2016.09.043

[13] Achimugu P, Selamat A, Ibrahim R, Mahrin MN. A systematic literature review of software requirements prioritization research. Information and software technology. 2014; 56 (6): 568-585.doi: 10.1016/j.infsof.2014.02.001

[14] Ahuja H, Batra U. Performance enhancement in requirement prioritization by using least-squares-based random genetic algorithm. Innovations in Computational Intelligence 2018, Springer, Singapore, pp. 251-263.doi: 10.1007/978-981-10-4555-4_17

[15] Cervantes H, Kazman R. Designing software architectures: a practical approach. Addison-Wesley Professional, 2016.

[16] Riegel N, Doerr J. A systematic literature review of requirements prioritization criteria. In: International Working Conference on Requirements Engineering: Foundation for Software Quality, Springer, 2015. pp. 300-317.doi: 10.1007/978-3-319-16101-3_22

[17] Araújo AA, Paixao M, Yeltsin I, Dantas A, Souza J. An architecture based on interactive optimization and machine learning applied to the next release problem. Automated Software Engineering 24 (3): 623-671. doi :10.1007/s10515-016-0200-3

[18] Mairiza D, Zowghi D. Constructing a catalogue of conflicts among non-functional requirements. In: International conference on evaluation of novel approaches to software engineering, Springer, Berlin, Heidelberg; 2010. pp. 31-44.doi: 10.1007/978-3-642-23391-3_3

[19] Yaseen M, Mustapha A, Ibrahim N. An approach for managing large-sized software requirements during prioritization. In: 2018 IEEE Conference on Open Systems (ICOS), IEEE; 2018. pp. 98-103.doi: 10.1109/ICOS.2018.8632806

[20] Ghédira K, Dubuisson B. Constraint satisfaction and optimization problems. Constraint Satisfaction Problems, Wiely 2013, pp.165-180. doi: 10.1002/9781118574522.ch7

[21] Löffler S, Liu K, Hofstedt P. A Meta Constraint Satisfaction Optimization Problem for the Optimization of Regular Constraint Satisfaction Problems. In Proceedings of the 11th International Conference on Agents and Artificial Intelligence, Prague, Czech Republic, ICAART (2); 2019, pp. 435-442. doi: 10.5220/0007260204350442

[22] Schreck J, Baretton G, Schirmacher P. Situation of the German university pathologies under the constraints of the corona pandemic-evaluation of a first representative survey. Der Pathologe, 2020; 41 (4). doi:10.1007/s00292-020-00791-y

[23] Sommerville Y. Software Engineering, 9th Edition. Pearson Education,2010.

[24] Rashidi H. Software Engineering-A programming approach, 2nd Edition. Allameh Tabataba'i University Press, Iran, 2014 (in Persian).

[25] Tam KY, Feng YK, Lai MC. Effective Use of Policing Systems: A Two-Stage Study of the Shakedown Period of System Implementation. IEEE Transactions on Engineering Management, 2019: 1-16. doi:10.1109/TEM.2019.2938983

[26] Oshana R. A User Interface: Police Command and Control System. Software Engineering for Embedded Systems, 1st edition, Newnes, 2013, pp. 1043-1087.

[27] Safwat A, Senousy MB. Addressing challenges of ultra large scale system on requirements engineering. Procedia Computer Science. 2015; 65: 442-449. doi:10.1016/j.procs.2015.09.116

[28] Hashemi SH, Mousavi Dehghani SA, Samimi SE, Dinmohammad M, Hashemi SA. Performance comparison of GRG algorithm with evolutionary algorithms in an aqueous electrolyte system. Modeling Earth Systems and Environment 2020; (6): 2103-2110. doi: 10.1007/s40808-020-00818-6

[29] Rudd K, Foderaro G, Zhu P, Ferrari S. A generalized reduced gradient method for the optimal control of very-large-scale robotic systems. IEEE Transactions on Robotics 2017; 33 (5): 1226-1232. doi:10.1109/TRO.2017.2686439