Journal of Informatics and Web Engineering
Vol. 1 No. 1 (March 2022) eISSN: 2821-370X

Incorporating Semi-Automated Approach for Effective Software Requirements Prioritization: A Framework Design

Fang-Fang Chua[1*], Tek-Yong Lim[2], Bushra Tajuddin[3], Amarilis Putri Yanuarifiani[4]

[1,2,3]Multimedia University, Malaysia
[4]Telkom University, Indonesia
**corresponding author: (ffchua@mmu.edu.my, ORCiD: 0000-0003-4114-1320)*

Abstract - Software Requirements Prioritization (SRP) is one of the crucial processes in software requirements engineering. It presents a challenging task to decide among the pool of requirements and the variance of the stakeholder's needs in prioritizing requirements. Semi-automated requirements prioritization is implemented in both manual and automatic processes. When prioritizing requirements, these aspects such as importance, time, cost and risk, should be taken into account. The emergence of machine learning is advancing to improve and automate the SRP process whereby decision making can be performed with minimal human intervention. Incorporating machine learning approaches in prioritization techniques can be implemented in the ranking process and classifying the priority group of the software requirements. A Semi-Automated Requirements Prioritization framework (SARiP), which implements semi-automatic process in requirements prioritization is proposed. SARiP concentrates on the areas related to prediction of requirements priority group and ranks requirements using classification tree and ranking algorithm. SARiP has been successfully evaluated in the government sector domain by the i-Tegur team from the Department of Information Technology, Ministry of Housing and Local Government of Malaysia (KPKT). 80% of the participants agreed that SARiP is extremely likely to help the participants in prioritizing the requirements for their projects. All participants agreed that SARiP is reliable and useful. Recording the requirements and results for the prioritization will be considered for future work and traceability function will be included to trace the requirements changes.

Keywords— Requirements Engineering, Software Requirements Prioritization, Kano model, Semi-automated, Ranking

Received: 23 December 2021; Accepted: 11 February 2022; Published: 16 March 2022

I. INTRODUCTION

Software Requirements Prioritization (SRP) is one of the crucial processes in software requirements engineering. One of the challenges during software development could be the selection of the 'right' requirements among a pool of requirements to fulfill different stakeholder's needs. This may lead to difficulties in organizing an effective prioritization process, forcing the organization to either rely on developer's experience or perform an additional requirements analysis and validation. Making a wrong selection will not only decrease the quality of the developed software but it will also incur additional cost for rectification processes in later phases. Thus, requirements prioritization would help to discover the most desirable requirements in different software product releases.

Journal of Informatics and Web Engineering
https://doi.org/10.33093/jiwe.2022.1.1.1

Published by MMU Press. URL: https://journals.mmupress.com/jiwe

Most of the time, prioritization needs to be done constantly or urgently especially when facing dynamic changes of project constraints. For example, prioritization for agile projects is a critical strategy in product releases through a series of time boxes running on a fixed schedule [1]. As requirements are changing frequently through the development cycle, [2] proposed a framework to prioritize requirements in agile software development which start early from user story. Besides, SRP also has to cope with the limited resources of projects whereby the process involved in managing different requirements are based on relative importance and urgency [3]. If budgets are limited or time is running out, [3] stressed that adequate prioritization should be performed in order to ensure the highly important requirements are attended to urgently. The classical requirement prioritization methods for incremental software development consider mostly the functional requirements and often ignore the non-functional requirements which might lead to disastrous failures in the final system [4]. In view of the mentioned problems, [4] proposed a novel prioritization approach for functional requirements which also considers the conflicts among non-functional requirements.

Current SRP processes are being performed offline, manually and independently which is time consuming especially if the amount of requirements is massive. In short, when the requirements are wrongly prioritized and not validated, this may lead to the delay of product delivery and the worst case; it will impact the overall cost, time and risk of the project. In addressing the mentioned problem statements, there is a need to identify the prioritization criteria to be used in software requirements selection. It is also crucial to identify the methods to be used to perform SRP processes automatically for quality decision making. Besides, the importance of evaluating the proposed framework in a real-world context will be highlighted in this proposed work. The aims of this research work are (1) to identify the prioritization criteria for requirements selection, (2) to propose a semi-automated framework that prioritizes software requirements for practical decision making and (3) to evaluate the proposed framework in the government sector domain. A Semi-Automated Requirements Prioritization framework (SARiP), which implements both manual and automatic process in requirements prioritization is proposed. The research work will focus on the software requirements prioritization for both functional and nonfunctional requirements in the government sector domain.

The proposed framework aims to automate activities which are involved in the SRP process which helps in the requirements selection as well. One of the software applications used by the public and developed by the Malaysian government, namely i-Tegur from the Ministry of Housing and Local Government of Malaysia (KPKT) will be used as the case study in this research study. The "project team" involved in this research study consists of a project manager, two system analysts, one business analyst, one data analyst, three developers and two testers while the "stakeholders" consist of the management staff of KPKT and representatives from local councils across Peninsular Malaysia. The business analyst and data analyst which are also the owner of the project were representing the stakeholders for this project. The proposed work contributes in the process of managing the relative importance, criticalness and urgency of different requirements types automatically and thus helps in reducing the risk during software development.

The remainder of this paper is divided as follows: Related Work section describes and compares the existing works in software requirements prioritization; Methodology section explains the six phases involved in this research; Proposed Solution section presents the proposed overall architecture and workflow; User Evaluation section presents the user evaluation results and the work is concluded in Conclusion section.

II. RELATED WORK

Relevant subject topics such as process flow of requirements prioritization, requirements prioritization techniques, machine learning techniques used in prioritization and existing semi-automated framework for prioritization are reviewed and analyzed. SRP is one of the crucial processes in software requirements engineering. It presents a challenging task to make a decision based on a set of requirements and the variance of the stakeholder's needs in prioritizing requirements. The prioritizing aspects such as time, importance, risk, cost, penalty, volatility, other aspects and combining different aspects of prioritization should be taken into account while prioritizing requirements [5],[6]. A SRP has three consecutive stages; (1) preparation stage which includes collecting requirements, choosing prioritization techniques and finalizing prioritization criteria; (2) execution stage which performs requirements prioritization activities and (3) presentation stage to present the results of the prioritization [7].

The process flow of requirement prioritization can be categorized as manual and semi-automated. The manual process flow starts manually from the beginning until the end of the requirement prioritization process whereas the semi-automated process flow starts with the manual process and followed by automated process to finalize the requirement prioritization. The most challenging task of manual process flow is to handle the conflict between different stakeholder's needs while prioritizing the requirements. Hence, various semi-automated requirements prioritization is

being proposed as a solution to minimize the conflict between stakeholders. [8] proposes a new semi-automated scalable prioritization technique called 'SRPTackle' to address the main challenges of the prioritization process such as scalability, unavailability of quantification and lack of automation.

A. Requirements Prioritization Techniques

1. Nominal scale

Under nominal scale category, requirements will be assigned to groups with different priorities but requirements within the same group are considered equal priority [9]. There are six different techniques of nominal scale such as "In or Out", three-level scale, MoSCoW, numerical assignment, Kano model and top ten requirements. MoSCoW, numerical assignment and Kano model are used in the proposed work for manual prioritization. The "In or Out" is considered the simplest technique among all prioritization techniques and involves two groupings only [1]. The stakeholders will be given a list of requirements and they decide which requirement will be included and which requirement is not included. Those requirements grouped as "In" are considered the same priority and to be implemented within a product. On the other hand, those requirements grouped as "Out" will be put aside for future consideration or removed completely. Three-level scale is another prioritization technique commonly used in which requirements are grouped into three different levels of priority, namely high priority, medium priority and low priority [1]. In order to make this prioritization process objective and precise, the stakeholders must agree and clearly define what high, medium and low priority means.

MoSCoW is a prioritization technique to group the requirements into four possible priority classification, namely Must, Should, Could and Won't [1]. Requirements that must be satisfied in the final product would be grouped in "Must". On the other hand, requirements grouped in "Should" is considered important but it is not compulsory. "Could" group contains requirements that are desirable by stakeholders and only implemented if time and budget is sufficient. The last "Won't" group consists of requirements that will not be implemented in the final product. Numerical assignment is also a commonly used prioritization technique where requirements are grouped into different priority groups and these groups are representation of what is demanded by the stakeholders [9]. For example, stakeholders may group requirements as "critical", "standard", "optional" and a percentage of requirements that can be placed for each group.

The Kano model explains how stakeholder satisfaction would change based on three different quality types, namely performance quality, basic quality and excitement quality [10]. If the product fulfills these requirements, there will be an increase in stakeholder satisfaction. When these requirements are not met, the stakeholders are more likely to be unsatisfied. Certain basic quality requirements are expected to be implemented in the product and stakeholders will take it for granted because these requirements are usually not clearly stated by the stakeholders. Even if the basic quality requirement is fulfilled, the stakeholder satisfaction level did not increase because these requirements are expected to be there. A decrease in the stakeholder satisfaction level will occur when these requirements are left out. When the excitement quality requirement comes together with the product, stakeholders will love it even if the stakeholders don't know or think about this requirement. The stakeholders are not affected if the product did not meet any excitement quality requirement.

2. Ordinal scale

For the ordinal scale, every requirement is prioritized in an ordered list. In other words, every single requirement has a different priority level. There are three common prioritization techniques under ordinal scale such as ranking, bubble sort, and pairwise comparison. Ranking technique which is being used in the proposed work allows the stakeholders to sort a list of requirements from number 1 (for the most important requirement) to number "n" (for the least important requirement). Sometimes, stakeholders may face difficulties sorting two similar requirements into the different ranks either "rank higher than" or "rank lower than". Bubble sort was introduced by [7]. This approach requires the stakeholders to compare two requirements and swap the requirements if the stakeholders found these requirements in the wrong priority order. The process will be repeated until there is no need to swap or the requirements are all in order of priority. The end list is the final ranked requirements. Pairwise comparison requires stakeholders to compare all possible pairs of requirements and make a decision which requirements in each pair have a higher priority. A matrix style tool is commonly used to support this pairwise comparison technique. At the end, the stakeholders make a rank order out of a list of requirements where the highest priority requirement refers to the requirement with the most appearance in the matrix.

3. Ratio scale

Ratio scale is considered as the viable among all the scales because it provides a relative distance between requirements on top of the ordering of requirements. One-hundred-dollar test and Analytical Hierarchy Process are examples of prioritization techniques utilizing ratio scale. In the hundred-dollar test, stakeholders will put a price on each requirement and the total value will come to a hundred dollars. The requirement that has the highest value is ranked first and so forth with the least important requirements will have the smallest value. Analytical Hierarchy Process (AHP) is a process to determine which requirements that have a higher priority and to what extent, all identifiable pairs of hierarchically classified requirements are compared and the importance are usually from a scale of one to nine with one representing equally important requirements and nine means it is an absolutely very important requirement. Then, the stakeholders will calculate the relative value and implementation cost of each requirement. A cost-value diagram is generated and the stakeholders evaluate the requirements based on this diagram. In short, AHP is highly time consuming.

B. Machine Learning in Prioritization Techniques

The existing semi-automated requirements prioritization in the market focused on the requirements ranks using various techniques. Machine learning in prioritization techniques is not only applicable in the ranking process but also can be implemented in classification approaches such as classifying the priority group of the software requirements. CBRank proposed by [11] uses the machine learning technique in processing stakeholder preferences of requirements ordering. With an intention to construct a strong classifier from a sample of weak classifiers using an ensemble technique, boosting method can create a very high accuracy rank prediction by using moderately accurate partial orderings and combining it linearly. Besides, classification techniques can be implemented to classify the priority group which is similar to the three-level scale in prioritization technique. According to [12], the classification approach in machine learning is mostly used in forecasting the group membership for data instances.

[12] conducted a research which focused on commonly used classification techniques such as Decision Tree, Bayesian Network (BN) K-Nearest Neighbor (kNN) and Support Vector Machines (SVM) which can be regarded as among the highly well-known and suitable technique for solving problems in connection to data classification, learning and prediction. According to [13], among challenges developers will face is using prior information to cluster stakeholders and requirements, but through clustering, patterns from ratings by already elicited stakeholders will be easier to find. SRPTackle proposed by [8] provides a semi-automated process based on an integrated formulation of requirement priority value function, such as multi-criteria decision-making method, clustering algorithms (K-means and K-means++), binary search tree, to increase efficiency and minimize human intervention.

By applying collaborative algorithms to clustered requirements and stakeholder, more effective and accurate prediction will be achieved. Clustering will help extend the framework enabling a scope of improvement. Particle Swarm Optimization (PSO) which was adopted in CDBR by [14]. PSO is swarm intelligence meta-heuristic where a population based state of algorithm is continuously modified until a termination criterion is met. DRank used RankBoost algorithm to generate subjective requirement prioritization formula, generate requirement dependency graphs (RDGs) based upon the contribution dependencies and business dependencies among the requirements, and analyze the contribution order to calculate the contribution of every requirement by using PageRank algorithm to finally integrate the final requirements prioritization [15]. SAFFRON used social networks and collaborative filtering to prioritize the stakeholders and their importance based on roles of the stakeholders [13]. Subsequently, it merges the matrix and uses correlation analysis to find similarity among the requirements for predicting new requirements prioritization.

C. Existing Semi-Automated Prioritization Approaches and Framework

Few existing semi-automated approaches and framework for requirements prioritization are being reviewed in the section. Collaborative Dependency Based Ranking approach (CDBR) is a collaborative requirements prioritization method based on semi-automated dependency proposed by [14]. This approach concentrates on minimizing the differences in opinion among stakeholders and developers, to ensure an effective collaboration can be done and for the preferable approximation from the result of the final prioritization that is agreeable to the stakeholder and developer. The CDBR targeted three major constraints which are usually overlooked in an existing work namely requirements dependencies, communication between stakeholders and developers, and the issue of scalability. Particle Swarm Optimization algorithms are used for the automation process whereby ranking is calculated in the order of the difference that stakeholders and developers have and being processed to produce a final priorities of implementation

agreeable by all stakeholders. CDBR however doesn't consider incorporating new requirements or change to existing requirements which shows lack of flexibility for the prioritization process.

[15] proposed a semi-automated prioritization method called DRank and this method is based on preferences and dependencies. It has seven steps: (i) select ranking criteria; (ii) selected evaluation attributes (SEAs) as based in selecting a scale value for each requirement; (iii) prioritize sampled requirements pair (Srps); (iv) create a subjective requirement prioritization (SubjRP); (v) generate requirement dependency graphs (RDGs); (vi) analyze contribution order (CO); and (vii) merge final requirement prioritization (FinalRP). A semi-automated DRank consists of both manual and automatic processes. The manual steps involved stakeholder's participation which includes selection of ranking criteria, ranking the requirements according to SEAs and selecting pairs of requirements in order to compare it using pair-wise. Stakeholders will then make an allocation of each pair relative priority afterwards. Finally, DRank integrated the final requirements prioritization as FinalRP. Same as CDBR, DRank did not consider incorporating new requirements or the changing of existing requirements. Hence, rank reversal issues will arise.

[13] proposed a Semi-Automated Framework for soFtware Requirements priOritization (SAFFRON), which predicts stakeholders rating to reduce human interactions. There are eight steps in SAFFRON; (i) collection of stakeholders ratings; (ii) calculating how stakeholders roles and individual influence can influence the project; (iii) every stakeholder's importance for each requirement from ratings and project influence will be calculated; (iv) requirements are prioritized based on total importance; (v) ratings are collected from a group of stakeholders for new requirements; (vi) merged rating matrix will provide gathering of user-requirement relation matrix and merging both matrices from previous requirements and new requirements; (vii) determine which users that will predict ratings; (viii) using collaborative filtering to predict ratings and use the new predicted rating. Stake-Rare approach is a two-step approach with an objective of finding similarity between requirements through correlation analysis and using a technique of model-based collaborative filtering to predict a rating from a certain stakeholder. The result of estimating the new requirements will be the final priority rank which will solve the rank reversal issue. As proposed by [11], CBRank is a SRP through Machine Learning Approach using Case Base Ranking. CBRank uses the machine learning technique in processing stakeholder preferences of requirements ordering. With the use of this method, preference information given by effort from humans can be reduced but at the same time, perpetuate the accuracy of the final ranking approximation. This method also allows exploitation of domain knowledge encoded as an incomplete order relations described over the requirement attributes, resulting in support of an adaptive elicitation process. CBRank however doesn't consider incorporating new requirements or change to existing requirements, thus inviting the issue of rank reversal.

D. Gap Findings

CDBR, DRank, CBRank and SAFFRON did not consider the utmost feature of requirements prioritization, which is providing the final rank of software requirements prioritization. CDBR, DRank and CBRank failed to consider the rank reversal, especially where the new requirements or a change of existing requirements occurs. SAFFRON did consider this issue but it did not consider classifying the priority group of the requirements. A Semi-Automated Requirements Prioritization framework (SARiP), which implements both manual and automatic process in requirements prioritization is proposed. SARiP concentrates on the areas related to prediction of requirements priority group and ranks requirements using classification tree and ranking algorithm. SARiP concentrates on the areas related to prediction of requirements priority groups and ranks. It will adopt machine learning classification techniques using decision tree and ranking algorithm for conducting the prediction of new requirements prioritization. The comparison of features for the requirements prioritization approaches is summarized in Table 1.

Table 1. Comparison of semi-automated framework for prioritization

	CDBR [14]	**DRank** [15]	**SAFFRON** [13]	**CBRank** [11]	**SARiP**
Manual Prioritization	Numerical assignment, Dependency Flow Graph	Ranking, Pairwise comparison	Ranking	Pairwise comparison	MosCoW, Numerical assignment, Kano Model
Automatic Prioritization	Particle Swarm Optimization	RankBoost, PageRank	Social Networks, Collaborative filtering, Merge matrix, Correlation analysis	RankBoost	Classification tree, Ranking algorithm

Most of the semi-automated requirements prioritization still follows the manual process flow in which these methods maintain the preparation process manually and automate the execution and presentation processes. The preparation process is done manually since this process requires input from the stakeholders and developers as initial priority requirements. This information will be also used as training data in the automation process to calculate the final requirements prioritization and predict the requirements prioritization for new requirements. The proposed Semi-Automated Requirements Prioritization framework (SARiP) will implement the preparation process manually, e.g: data collection from stakeholders and project team. Combination prioritization techniques of MoSCoW, Numerical Assignment and Kano model will be applied by the project team and stakeholders to classify the requirements criteria and scale the requirements based on selection criteria. The automated part for SARiP will be done using a classification approach which includes a decision tree in machine learning technique. This approach is conducted to forecast the priority group of the requirements based on the scoring of the requirements. SARiP will also adopt a ranking algorithm to rank the requirements. Both approaches will be conducted when the need to predict new requirements prioritization or change of existing requirements prioritization occurs. The proposed method will focus on the classification of prioritization grouping and rank reversal issues which involve prediction of the new requirements by the stakeholders.

III. METHODOLOGY

This section proposed the research methodology to be used. There are five phases of research activities involved in implementing SARiP. Requirements are gathered and analyzed. Semi-automation method is then being proposed and SARiP is developed for the requirements prioritization process. The input and output for each phase in research methodology with the activities involved are represented in Figure 1.

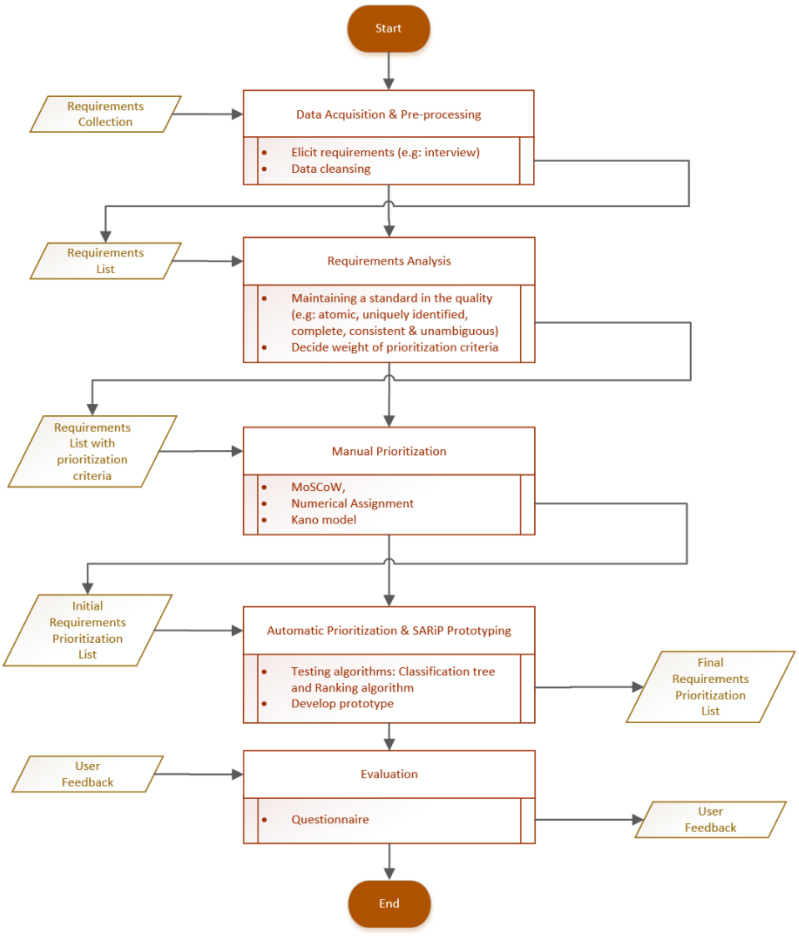


Figure 1. Proposed Methodology

A. Data Acquisition & Pre-Processing

Qualitative research method is adopted to elicit the requirements from a project team in government sector, where the primary data is gathered and analyzed. An in-depth interview with the government officers is conducted in order to gauge their views, expectations, and understanding on the current and future services demands with regard to adopting the semi-automated software requirements prioritization framework in government enterprise architecture. The pre-processing of the primary data gathered and the requirements is done manually to ensure that only the relevant information will be taken for analysis purposes. The data set used consists of functionality and quality requirements for i-Tegur project. The data will go through cleansing processes such as eliminating the noisy data, handling the missing values (filling in the values or eliminated if not important), or resolving the inconsistencies in the data before it is transformed into an understandable format. The cleansing processes were done manually.

B. Requirements Analysis

Requirements analysis is conducted according to data gathered in the early phase and requirements generated shall be atomic, uniquely identified, complete, consistent and unambiguous. Requirements analysis phase is important to ensure all the requirements are clear, clean and ready for the prioritization process. The prioritization criteria for requirements selection are also determined in this phase. The project team and stakeholders will decide on the weight of each prioritization criteria. The proposed SARiP framework prototype prioritization criteria are represented in Table 2.

Table-2. Prioritization criteria and details information

Criteria	**Details**
MoSCoW	Measure the significance of requirements using MoSCoW technique
Cost	Measure the cost involved in developing the application
Time	Measure the time sensitivity to develop application
Risk	Measure the risk while developing application, the risk if development of requirements delayed
Complexity	Measure the complexity of the requirements
Importance	Measure the importance of requirements using Kano model

The pre-processing of the primary data gathered and the requirements is done manually to ensure that only the relevant information will be taken for analysis purposes. The project team and stakeholders will then proceed with the process of identifying the prioritization criteria which involves deciding the weight of each prioritization criteria. The prioritization criteria for the proposed SARiP prototype include MoSCoW, cost, time, risk, complexity and importance. The stakeholders will decide the weight for each requirement criteria. The total weight of the criteria will be 100 percent. Table 3 shows the example of weight criteria for the proposed SARiP prototype requirements prioritization.

Table 3. Example of weight criteria for requirements prioritization

Criteria	**% Weight**
MoSCoW	10
Cost	10
Time	20
Risk	10
Complexity	25
Importance	25
Total	100

C. Manual Prioritization

The manual prioritization process involved participation of the project team and stakeholders. The first process in this phase involved the project team who will use MoSCoW technique to identify and classify the requirements accordingly. MoSCoW is used as a guideline to ensure the basic importance of the requirements. The project team will rate and score the requirements based criteria of the scoring method. The scoring method for MoSCoW technique is depending on the criteria (e.g: 1 - the least important priority, 4 - the highly important priority) stated in Table 4.

Table 4. Scoring criteria for MoSCoW technique

Criteria	Score
Must	4
Should	3
Could	2
Won't	1

The project team and the stakeholders will then use numerical assignment techniques to rate the scoring for other prioritization criteria as in Table 5. The scoring method for these criteria is based on the criteria in Table 5 with the scale of 1 to 10 (e.g: 1 - the least important priority, 10 - the highly important priority).

Table 5. Scoring criteria for cost, time, risk and complexity

Criteria	Score
Cost	1-10
Time	1-10
Risk	1-10
Complexity	1-10

The final process in manual prioritization involved stakeholders where the stakeholders will rate the scoring for importance criteria based on the Kano model. The basic quality and performance in the Kano model will be taken into consideration and the excitement part in the Kano model will be a guideline for an added value to the requirements. The added value indicates that it's nice to have but not important. Thus, it will be considered after the important requirements were being prioritized. The scoring method for importance criteria is based on the criteria (e.g: 1 - the least important priority, 3 - the highly important priority) in Table 6.

Table 6. Scoring criteria for importance

Criteria	Score
Basic	3
Performance	2
Excitement	1

D. Automatic Prioritization

The automation process adopts machine learning techniques in generating the priority groups and final priority ranks of requirements which consist of two parts, prioritization and prediction. Accuracy testing is done while choosing the suitable classification model for the proposed prototype. Thus, Decision Tree and Naïve Bayes are selected in conducting accuracy tests for classification models. The function of this classification model is to predict the priority groups. Results of the accuracy testing between two classification models chosen are shown in Figure 2.

Decision Tree Classification model

```
Confusion Matrix and Statistics

          Reference
Prediction High Low Medium
    High      8   0      0
    Low       0   2      0
    Medium    0   0     11

Overall Statistics

               Accuracy : 1
                 95% CI : (0.8389, 1)
    No Information Rate : 0.5238
    P-Value [Acc > NIR] : 1.267e-06

                  Kappa : 1
 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: High Class: Low Class: Medium
Sensitivity                1.000    1.00000        1.0000
Specificity                1.000    1.00000        1.0000
Pos Pred Value             1.000    1.00000        1.0000
Neg Pred Value             1.000    1.00000        1.0000
Prevalence                 0.381    0.09524        0.5238
Detection Rate             0.381    0.09524        0.5238
Detection Prevalence       0.381    0.09524        0.5238
Balanced Accuracy          1.000    1.00000        1.0000
```

Naïve Bayes Classification model

```
Confusion Matrix and Statistics

          Reference
Prediction High Low Medium
    High      0   0      0
    Low       0   0      0
    Medium    8   2     11

Overall Statistics

               Accuracy : 0.5238
                 95% CI : (0.2978, 0.7429)
    No Information Rate : 0.5238
    P-Value [Acc > NIR] : 0.5874

                  Kappa : 0
 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: High Class: Low Class: Medium
Sensitivity                0.000    0.00000        1.0000
Specificity                1.000    1.00000        0.0000
Pos Pred Value               NaN        NaN        0.5238
Neg Pred Value             0.619    0.90476           NaN
Prevalence                 0.381    0.09524        0.5238
Detection Rate             0.000    0.00000        0.5238
Detection Prevalence       0.000    0.00000        1.0000
Balanced Accuracy          0.500    0.50000        0.5000
```

Figure 2. Comparison of two classification models

From the results shown in Figure 2, Decision Tree is chosen as classification model for SARiP prototype. The input of automatic prioritization is the initial set of prioritized requirements with the top-level of selection criterion. The prioritization part will start by calculating the score based upon the initial set of prioritized requirements by the project team and stakeholders. The weight criteria and scoring method identified in the previous phase will be used in calculating the score for requirements prioritization. The formula for calculating each criteria of the requirement are defined as Equations (1), (2), (3), (4), (5) and (6).

$$MoSCoW \rightarrow \frac{score}{4} * moscow\_weight \quad (1)$$

$$Cost \rightarrow \frac{score}{10} * cost\_weight \quad (2)$$

$$Time \rightarrow \frac{score}{10} * time\_weight \quad (3)$$

$$Risk \rightarrow \frac{score}{10} * risk\_weight \quad (4)$$

$$Complexity \rightarrow \frac{score}{10} * complexity\_weight \quad (5)$$

$$Importance \rightarrow \frac{score}{3} * importance\_weight \quad (6)$$

The weight of each requirement criteria is defined in the requirements analysis phase by the project team and stakeholders. Total score for requirements criteria will be calculated using Equation (7) when the criteria of each requirement are calculated.

$$Total\ Score \rightarrow MoSCoW + Cost + Time + Risk + Complexity + Importance \quad (7)$$

The second part of SARiP framework automatic prioritization is prediction. There are commonly used classification techniques in machine learning including Decision Tree, Support Vector Machines (SVM), K-Nearest Neighbour (kNN) and Bayesian Network [12]. Based on commonly applied classification techniques and accuracy testing done previously, the proposed SARiP prototype adopted Decision Tree algorithm to predict classification of priority groups which consist of three different priority groups (e.g: Low, Medium and High priority). The build classification or regression models of the decision tree are in the form of a tree structure. It works by disintegrating the dataset into smaller subgroups while concurrently; an associated decision tree is gradually developed. The output of it is a tree with decision nodes and leaf nodes. The highest node in a tree which has no parent is called root node. Decision trees can cope with both numerical and categorical data. A classification tree is applied to learn a classification function. This function forecasts the value of a target attribute given the values of the predictors (input) attributes. To build a decision tree, one needs to calculate impurity. Entropy is one of the techniques used to measure impurity. Formula to calculate entropy as Equation (8), which the logarithm is base 2.

$$E(S) = \sum_{i=1}^{c} -pi\ log\ 2\ pi \quad (8)$$

A pure table of entropy is zero which consists of a single class, where the probability is 1 and log(1) = 0. The entropy achieves its maximum value when entire classes in the table have perfect probability. The purpose of classification technique using decision tree is to predict the priority group classification. The classification of priority groups is divided by the total score as defined in Table 7.

Table 7. Priority group score

Priority Group	Score
High	> 75
Medium	<= 75
Low	< 50

The next step conducted in the SARiP framework is the prediction of requirements ranks based on score and priority group which involves a ranking algorithm in machine learning using order.scores function. A new data frame is created for ranks, and order.scores function is applied to get the approximated ranks based on the total scores and priority group for the requirements. When there are new requirements coming in or a change of existing requirements, there is a need to predict the new ratings for those new requirements. Indirectly, it will be able to minimize human interactions and shorten the time of requirements prioritization activities. Prediction of new requirements and change of existing requirements uses both techniques as in the previous step, classification using decision tree and ranking algorithm. Classification is performed using decision tree and ranking algorithm whereby the original data will be partitioned into two sections; training data and testing data. The training data consists of 70% of the original data (existing requirements) and the other 30% of the data (new requirements) will be used as testing data. Once the training process is completed the prediction of the priority group for new requirements starts. Then, both training and testing data will be combined to get the new ranks for the final approximation prioritization ranks.

E. SARIP Prototyping

A prototype of SARiP, a semi-automated software requirements prioritization framework is developed to visualize the prioritization process as well as to obtain feedback and recommendation for the framework implementation. Prototyping also helps in determining the direction of the entire project, and its eventual success. The phase starts with the design of template of the requirements prioritization. The template is used for initial requirements prioritization, which is done manually in the preparation stage by both project team and stakeholders. The template is designed and adjustments will be made to the data to ensure that the data fits the template accordingly.

F. Evaluation

The evaluation of the SARiP framework is conducted to validate the requirements prioritization group and ranks. The evaluation process is done by the government officials as a case study. The process starts with the testing of the prototype and the users will be given an evaluation form to evaluate the tested prototype. This is to ensure that the framework is developed according to the user requirements and the final ranks with the prediction of new priority ranks are correct and accurate. Correctness and accuracy are defined by the users as in the user requirements and validated during the evaluation phase.

IV. PROPOSED SOLUTION

The proposed SARiP framework begins with data acquisition and pre-processing, followed by requirements analysis. When both phases are done, a requirements list will be issued. The requirements list will be used in SARiP for prioritizing the requirements. The execution starts with the manual prioritization, where the requirements will be prioritized manually by the project team and stakeholders. The output of this process is the initial prioritization list. Then, the initial prioritization list will be used as an input for automatic prioritization. The final output of the SARiP framework is the list of requirements with classification of priority grouping and the prioritization rank. Finally, the approximated requirements prioritization priority group and rank will be evaluated. Overall architecture of the SARiP framework is represented in Figure 3 and Figure 4 depicted the overall process flow of the SARiP framework.

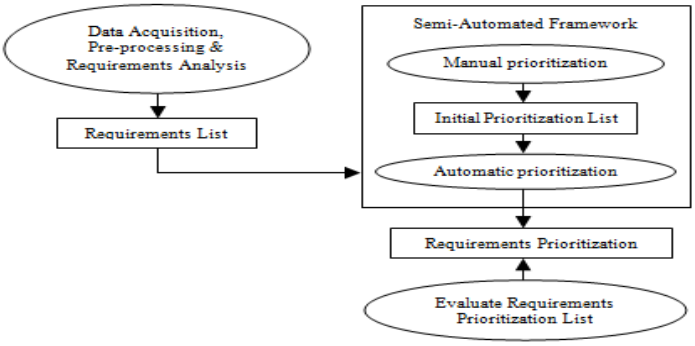


Figure 3. Overall architecture of SARiP framework

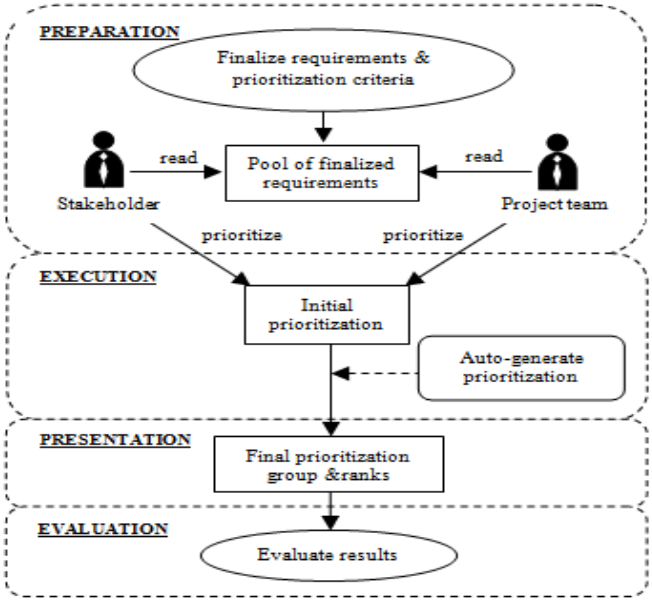


Figure 4. Process flow of SARiP requirements prioritization

Use case diagram of SARiP framework which captures the main functionalities of SARiP is represented in Figure 5. Two main user groups were identified, namely project team and stakeholders. Both user groups can use the SARiP framework prototype for six different use cases such as set requirements prioritization weight, upload initial requirements prioritization list, view prioritization priority and ranking, edit requirements prioritization weight, upload new initial requirements prioritization list, view prioritization priority and re-ranking. Class diagram is a static image that depicts the static view of an application. The class diagram is used as a visual description and documentation on different aspects of an application. Class diagram of the SARiP framework is represented in Figure 6.

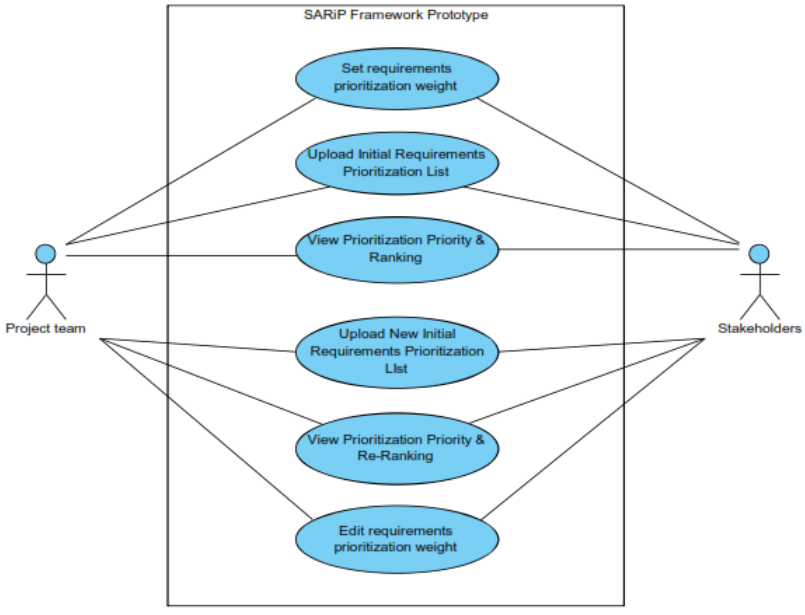


Figure 5. Use case diagram of SARiP framework

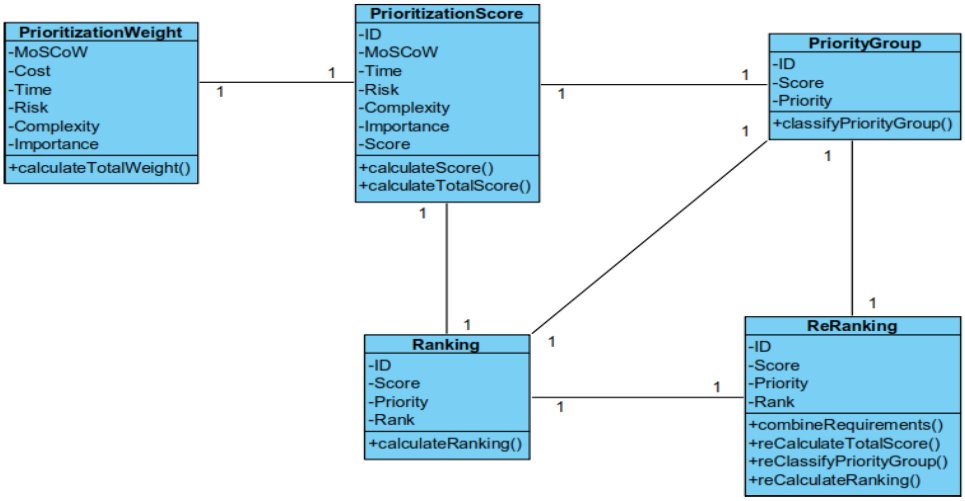


Figure 6. Class diagram of SARiP framework

The automatic prioritization requires an interface for interaction between manual process and automatic process. Thus, the Graphical User Interface (GUI) of the SARiP framework main page is designed as represented in Figure 7.

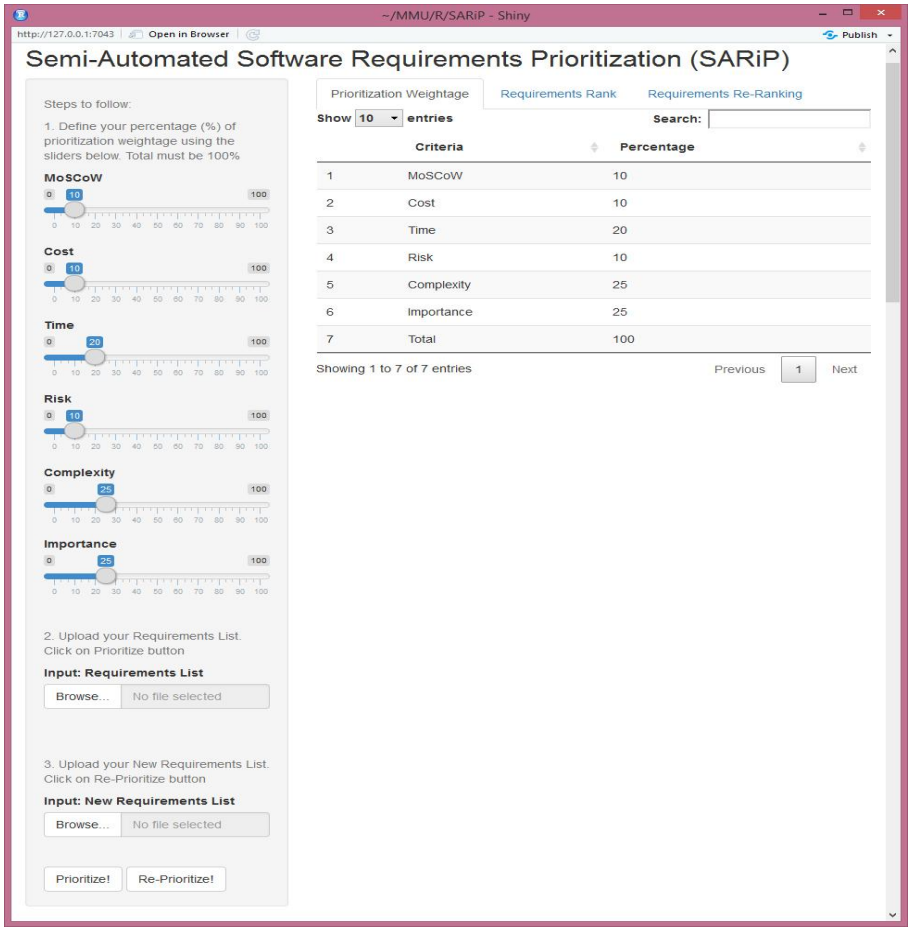
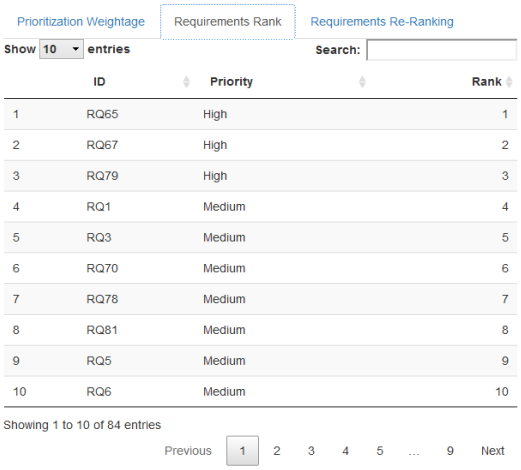


Figure 7. Graphical User Interface (GUI) of SARiP framework

The proposed SARiP framework offers a flexible setting of weight prioritization criteria, in case the stakeholders decide to change the weight criteria. For example, the final list of requirements prioritization with the priority group and ordered ranks has been issued, but the stakeholders suddenly decide to change the weight of prioritization criteria which the SARiP framework is able to handle in this situation. The project team will just be required to change the weight prioritization criteria accordingly and re-run the program. The final result of requirements prioritization with the priority group and ordered ranks will be re-calculated and displayed accordingly in the respective displayed tab. Hence, SARiP framework is able to handle different datasets as long as the dataset is using an understandable format. Example of requirements prioritization result is shown in Figure 8.


Prioritization Weightage | Requirements Rank | Requirements Re-Ranking

Show 10 entries Search:

	ID	Priority	Rank
1	RQ65	High	1
2	RQ67	High	2
3	RQ79	High	3
4	RQ1	Medium	4
5	RQ3	Medium	5
6	RQ70	Medium	6
7	RQ78	Medium	7
8	RQ81	Medium	8
9	RQ5	Medium	9
10	RQ6	Medium	10

Showing 1 to 10 of 84 entries

Previous 1 2 3 4 5 … 9 Next

Figure 8. Example of requirements prioritization result

V. USER EVALUATION

User evaluation was conducted and participated by the i-Tegur team. Five respondents, including project manager, system analyst, data analyst, developer and tester, were participating in the evaluation of SARiP framework. The participants tested the SARiP framework and filled in the evaluation form. The evaluation form is made of ten questions and evaluation results summary which includes Satisfaction, Meet the needs, Helps in requirements prioritization and SARiP characteristics are shown in Figure 9. 80% of the participants agreed that in overall, the proposed SARiP framework prototype is somewhat satisfied and another 20% thinks that it is very satisfied and effective. Participants are being asked whether the proposed SARiP framework prototype will help in prioritizing the requirements for their projects. Result shows that 80% of the participants agreed that the proposed SARiP framework prototype is extremely likely to help the participants in prioritizing the requirements for their projects. All participants agree that the proposed SARiP framework is reliable and useful. 80% of the participants agreed that SARiP framework produced an accurate result. 40% of the participants agreed that the prototype is unique and 20% thinks that the proposed SARiP framework is a high quality product which facilitates an effective prioritization process.

VI. CONCLUSION

Requirements prioritization is always a challenging task to solve the variance of needs between stakeholders. Much time and effort is used to select and prioritize the requirements in satisfying all stakeholders who might cause the delay of software development. A semi-automated framework is proposed to facilitate the requirements selection and ranking process for practical decision making. MoSCoW, cost, time, risk, complexity and importance are identified as the requirements prioritization criteria with flexible setting weight. The proposed framework is known as the SARiP prototype which performs both manual and automatic process of requirements prioritization. The manual process involved MoSCoW technique, numerical assignment technique and Kano model as a guideline for the project team and stakeholders. The automatic process implemented a classification tree to predict the requirements priority group. Ranking algorithm was applied to predict the ranks of the requirements. The proposed SARiP prototype also considered rank reversal issues for extension of new requirements. Finally, the proposed SARiP framework has been successfully evaluated in the government sector domain by the i-Tegur team from the Department of Information

Technology, Ministry of Housing and Local Government of Malaysia (KPKT). Currently, the proposed SARiP prototype uses excel file as input in generating the requirements prioritization results and does not store them in the database. Hence, storing the requirements and results for the prioritization shall be considered for future works. Other than that, traceability to trace the requirements changes would also be an added advantage in the future.

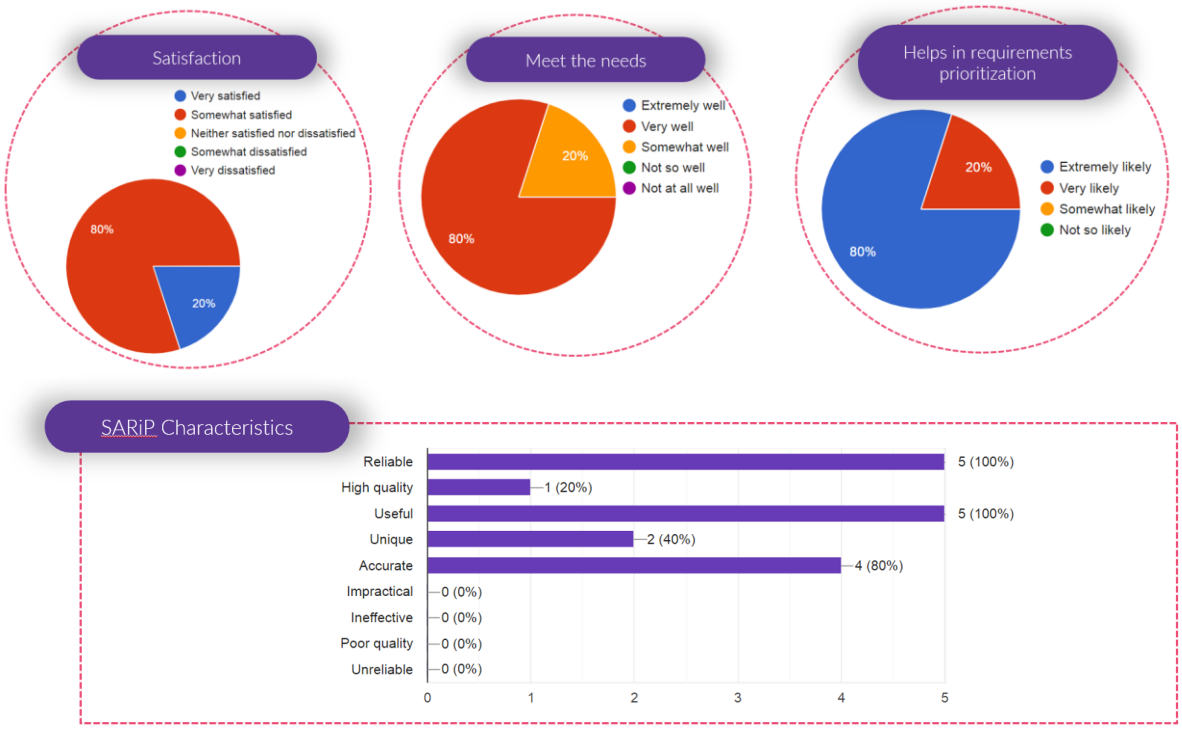


Figure 9. Results of user evaluation

ACKNOWLEDGEMENT

The authors received no funding from any party for the research and publication of this article.

REFERENCES

[1] K. Wiegers, and J. Beatty, "Software requirements: 3-rd edition", Microsoft Press, Washington, 2013.
[2] K. AbdElazim, R. Moawad, E. Elfakharany, "A framework for requirements prioritization process in agile software development", Journal of Physics: Conference Series, vol. 1454, no. 1, 012001, IOP Publishing, 2020.
[3] M. Schedlbauer. “Requirements Prioritization Strategies”, CEG, 2018. https://www.corpedgroup.com/resources/ba/ReqsPrioritization.asp
[4] M. Roy, N. Deb, A. Cortesi, R. Chaki, N. Chaki, "NFR ‐ aware prioritization of software requirements. Systems Engineering", vol. 24, no. 3, pp. 158-176, 2021. https://doi.org/10.1002/sys.21572
[5] P. Berander, A. Andrews, "Requirements Prioritization", Engineering and Managing Software Requirements. Springer, Berlin, Heidelberg, 2005. https://doi.org/10.1007/3-540-28244-0_4
[6] F. Sher, D. N. A. Jawawi, R. Mohamad and M. I. Babar, "Requirements prioritization techniques and different aspects for prioritization a systematic literature review protocol," Malaysian Software Engineering Conference (MySEC), Langkawi, Malaysia, 2014, pp. 31-36, doi: 10.1109/MySec.2014.6985985.
[7] J. Karlsson, C. Wohlin, B. Regnell, "An evaluation of methods for prioritizing software requirements", Information and software technology, vol. 39, no. 14-15, pp. 939-947, 1998.
[8] F. Hujainah, R.B.A. Bakar, A.B. Nasser, B. Al-haimi, K.Z. Zamli, . "SRPTackle: A semi-automated requirements prioritisation technique for scalable requirements of software system projects", Information and Software Technology, 131, 106501, 2021. https://doi.org/10.1016/j.infsof.2020.106501
[9] M. Narendhar, K. Anuradha, "Different Approaches of Software Requirement Prioritization", International Journal of Engineering Science Invention, vol. 5, no. 9, pp. 38-43, 2016.

[10] N. Kano, N. Seraku, F. Takahashi, S. Tsuji, “Attractive quality and must-be quality", The Journal of Japanese Society for Quality Control, vol. 14, pp. 39-48, 1984.
[11] A. Perini, A. Susi and P. Avesani, "A Machine Learning Approach to Software Requirements Prioritization," IEEE Transactions on Software Engineering, vol. 39, no. 4, pp. 445-461, 2013. doi: 10.1109/TSE.2012.52.
[12] A.A. Soofi, A. Awan, "Classification techniques in machine learning: applications and issues", Journal of Basic & Applied Sciences, vol. 13, pp. 459-465, 2017.
[13] S.A. Asif, Z. Masud, R. Easmin, A.U. Gias, "SAFFRON: a semi-automated framework for software requirements prioritization", arXiv preprint arXiv:1801.00354, 2017.
[14] A. Gupta, C. Gupt, "CDBR: A semi-automated collaborative execute-before-after dependency-based requirement prioritization approach", Journal of King Saud University-Computer and Information Sciences, vol. 34, no. 2, pp. 421-432, 2022. https://doi.org/10.1016/j.jksuci.2018.10.004
[15] F. Shao, R. Peng, H. Lai, B. Wang, "DRank: A semi-automated requirements prioritization method based on preferences and dependencies", Journal of Systems and Software, vol. 126, pp. 141-156, 2017.