Modern Applied Science; Vol. 12, No. 2; 2018
ISSN 1913-1844 E-ISSN 1913-1852
Published by Canadian Center of Science and Education

Requirements Prioritization Techniques Comparison

Amjad Hudaib[1], Raja Masadeh[2], Mais Haj Qasem[1] & Abdullah Alzaqebah[2]

[1] University of Jordan, Computer Science Department, Amman, Jordan

[2] Computer Science Department, the World Islamic Science and Education University, Amman, Jordan

Correspondence: Mais Haj Qasem, University of Jordan, Computer Science Department, Amman, Jordan. E-mail: mais_hajqasem@hotmail.com

Received: December 29, 2017 Accepted: January 13, 2018 Online Published: January 16, 2018

doi:10.5539/mas.v12n2p62 URL: https://doi.org/10.5539/mas.v12n2p62

Abstract

Requirements prioritization is considered as one of the most important approaches in the requirement engineering process. Requirements prioritization is used to define the ordering or schedule for executing requirement based on their priority or importance with respect to stakeholders' viewpoints. Many prioritization techniques for requirement have been proposed by researchers, and there is no single technique can be used for all projects types. In this paper we give an overview of the requirement process and requirement prioritization concept. We also present the most popular techniques used to prioritize the software project requirements and a compression between these techniques. On the other hand, we spot the light on the importance of involving the non-functional requirements prioritization because of the great effects of non-functional on project success and quality; some approaches that used in prioritize non-functional requirements are discussed in this paper, in addition a general model is proposed based on reviewing the prioritization techniques in order to suggests a best suited technique for specific projects according to decision makers parameters.

Keywords: requirements analysis, requirement prioritization, prioritization techniques

1. Introduction

Requirement engineering (RE) is one of the earliest phases of software development lifecycle. In addition; its process contains several activities; feasibility study, requirements collection, classification, structuring, prioritization, validation, specification, and management (Sommerville & Sawyer, 1997). While the requirement prioritization is considered one of the most significant activities in the process to construct software project and deliver the good system as the customer need (Svensson et al., 2011). Most projects include a large number of software requirements which to be prioritized according to the limited resources in terms of time, budget and customer satisfaction which is the major purpose in software development. As all the requirements related to more than release and based on customer needs the software engineers do not know which requirements have higher priority and which are not. Thus, there are various stakeholders are participated in the system development in order to prioritize the requirements in the right way according to their importance, therefore, that requirements can be ordered in execution. Whereas, not all requirements can be executed simultaneous, requirements are not prioritized. Moreover, the stakeholders have various opinions as regards the priority of each requirement. In other words, the stakeholders' agreement of the priority of requirements should be taken into consideration for the requirement prioritization process.

In software engineering process, several techniques were employed for aggregation and selection the right requirements. Therefore, various techniques are adopted in the industry based on diverse criteria, such as time, cost, importance, etc., these factors may result in conflicts way due to the strong relationships between them.

In this study, several techniques are presented for the requirements prioritization. Some of them work very well with projects which have huge number of requirements; while other techniques incapable to give accepted results when transacting with projects that have large number of requirements. Thus, the most common techniques that are used for requirements are presented in this paper.

This paper is organized as follows: section II contains the requirements prioritization concept, while section III outlines the most popular techniques. Section IV shows the comparison between several techniques. Section V overviews the non-functional requirement prioritization approaches, advantages and disadvantages for reviewed techniques is presented in section VI, the proposed general model is discussed in section VII Finally, the last

section draws the conclusion of this study.

2. Requirement Prioritization

The meaning of requirements prioritization is seen from several angles. Summerville defined the requirements prioritization as one of the most significant task for decision makers (Greer & Bustard, 1997). While; Firesmith defines it as the major process in software engineering as it gives perfect implementation order of the requirements in order to planning software versions and supplying desirable functionality as early as possible or the process to define the priority of the requirements to stakeholders based on the requirements importance (Kousalya et al., 2012). Therefore, we can conclude that the requirements prioritization denotes the prioritization by importance or by implementation.

Implementing the most significant operations that leads to get incremental feedback from the customer, set schedule, solve mistakes and resolve any misunderstand between the customer and the corporation in premature phases that lead up to customer contentment. Moreover, it is valuable by eliminating needless requirements which may be inefficiently costly and choosing the most suitable requirements for each version; which leads to assist in future planning, reduce the risk of cancellation, evaluate the benefits, prioritize the investments and determine the financial effect with regards to the implementation of each requirement (Ibrahim & Nosseir, 2016).

Requirement prioritization is the most significant and critical portion of requirements analysis due to the restrictions in project resources. In other words, it is so hard to implement the whole requirements simultaneously due to the restrictions in resources whence of schedule, staff and budget. Moreover, to improve some projects may require many months or often several years, wherefore it is important to determine the requirement that should be implemented at the beginning. Furthermore, budget plays an important factor, especially when transaction with requirement prioritization process because budget is considered as small activity with regards to requirement engineering compared to other activities in software engineering. Lastly, as mentioned before concludes that requirements have various levels with regards to their importance and it is complicate to determine which one is the most important.

As mentioned before, the project stakeholders are the base of the prioritization process with respect to business and regulations factors because they have various viewpoints and each one must determine the highest priority for requirements in order to impose stakeholders to clearly gather all the relative importance requirements that guides to raise the communication between stakeholders, supplies a reasonable base for requirement negotiation and enables engineering to schedule the development activities in reasonably.

3. Requirements Prioritization Techniques

Numerous methods on how to prioritize requirements have been developed which some work best on a small number of requirements, others are better suited to very complex projects with many decision-makers and variables. Prioritizing methods guide decision makers to analyze requirements to assign numbers or symbols that reflect their importance. Many challenges are facing these techniques including budget, time, resources, technical constraints, the need for professional skills to implement these techniques, and the need to satisfy the clients' expectations.

Requirements prioritization techniques depend on experts in the field, need high communication with stakeholders, highly dependent on other requirements. This makes the job of proposing the right technique more difficult and makes improvements on these techniques highly needed. According to (Karlsson et al., 1998), a prioritizing session could be consisting of three consecutive stages:

1. **The Preparation Stage**: in this stage, according to the principle of the prioritizing technique that could be used, the person structures the requirements. Moreover, a team and a team leader are chosen for the session and supplied all substantial information.
2. **The Execution Stage**: in this phase, the decision makers define the actual prioritizing for the requirements based on the information that gained from the prior stage.
3. **The Presentation Stage**: where the execution's results are offered for the persons who involved.

In this study, the most popular techniques for requirements prioritization are presented. These techniques take the results of the lower level prioritization activities, then they do some computation to calculate the requirements ordering (Vestola, 2010). The prioritization techniques can be categorized into three general scales to present the results as nominal scale, ordinal scale, and ratio scale as illustrated in Figure 1.

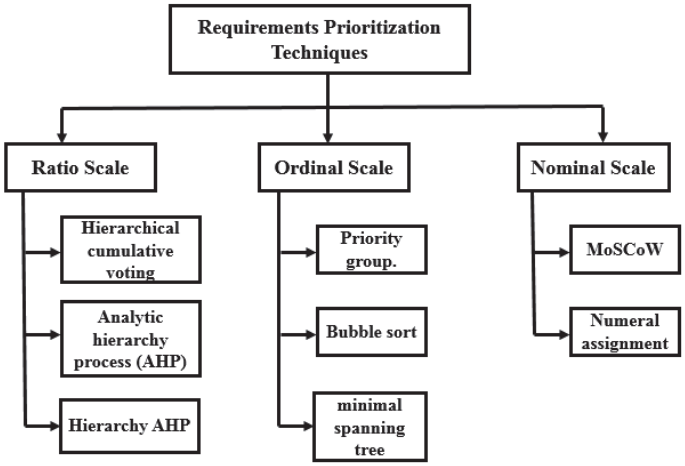

Figure 1. Requirements prioritization classification

3.1 Nominal Scale

Nominal scale prioritization mechanisms generate enumerate of classes to which objects can be categorized. Which means, requirements are classified into categories relies on their importance. Thus, all requirements that classified in the same category have the same priority (Sommerville & Sawyer, 1997). This type includes only two mechanisms which are Numeral assignment technique and MoSCoW technique.

3.1.1 Numerical Assignment Technique

The numerical assignment is considered as the traditional and popular prioritization mechanisms. In addition to that; it relies on clustering requirements into various classifications, where the number of priority groups can vary, but three groups are perhaps the most common division (Sommerville & Sawyer, 1997). As an example, requirements can be classified as critical, standard, and optional (Vestola, 2010). Each requirement could be assigned with a number scale from 1 to 3 to identify its importance. The numbers have the following meaning:

1) Does not matter (optional)
2) Rather important (standard)
3) Very important (critical)

The results of a numerical assignment classified as nominal scale. All requirements in same priority group denote equal priority. Thus, no more information presents that one requirement has more or less priority than others inside the same priority (Ma, 2009). The numerical assignment is discussed by a number of studies such as (Berander, Andrews, 2005), (Bradner, 1997), (IEEE-STD,1998), (Karlsson et al., 2006), (Leffingwell & Widrig, 2000) and (Sommerville & Sawyer, 1997).

3.1.2 MoSCoW Technique

Moscow technique is one of the easiest methods for requirement prioritization. MoSCoW technique is used by analysts and stakeholders for prioritizing requirements in a collaborative manner. MoSCoW method is the prioritization technique that is beginning from the dynamic software development method (DSDM), (Hatton, 2007) and (Tudor & Walter, 2006). According to MoSCoW mechanism, the list of requirements can be classified into the following four priority categories (Tudor & Walter, 2006), and illustrated in Figure 2:

- **M – Must have**. In this group, requirements must be contained in the project. Failure to deliver these requirements means the whole project would be a failure
- **S – Should have**. A high-priority feature that is not critical to launch. But it is supposed to be important and of a high value to users. Such requirements fill the second place on the priority list.
- **C – Could have**. This group contains the desirable requirement but not the necessary one. But these requirements are less important than the one in the “should have” group.
- **W – Won’t have**. A requirement that will not implement in a current development but may be included

in a future stage.

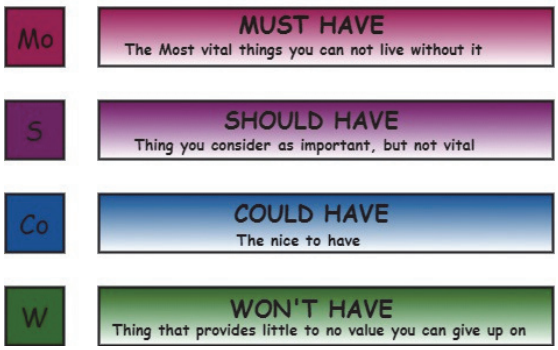

Figure 2. MoScoW Technique

Requirements that assigned to these groups based on the importance of implementing. The results of MoScoW are classified as nominal scale. All requirements in same priority group represent similar priority. No additional information will show one requirement is of higher or lower priority than another requirement inside same priority group.

3.2 Ordinal Scale

Ordinal scale prioritization techniques generate ranked lists of requirements. The ordinal scale can only show that one requirement is more important than another requirement, but not to what extent of it. Priority groups, Minimal spanning tree, and Bubble sort are techniques that involved in this section.

3.2.1 Priority Groups Technique

The priority groups technique was reported by (Karlsson et al., 1998). Priority groups do not actually create groups of requirements as a final result. Instead, the result is a ranked list of requirements. Priority groups technique is identical to numeral assignment technique, which assigns every requirement to one of three groups: low group, medium group, and high group. The difference between these two techniques is that the numeral assignment technique groups the requirements only once, where priority groups it groups the requirements repeatedly. priority groups technique is illustrated in Figure 3.

1) **In preparation stage**, all candidate requirements are outlined.
2) **In execution stage,** each of the requirements is put into one of the three groups: high, medium or low priority. In each group that have more than one requirement, three new subgroups are created, and the requirements are put into these groups and repeat this step recursively until there is only one requirement in all subgroup
3) **In presentation Stage**, they printed the requirements from left to right.

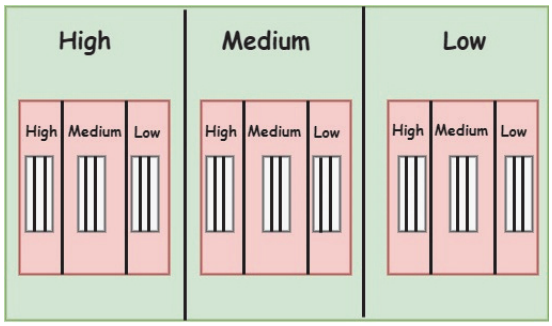

Figure 3. Priority Groups Technique

(Karlsson et al., 1998) are the only empirical study about priority groups in which total of 13 requirements was prioritized with the following five techniques: AHP, minimal spanning tree, binary search, bubble sort and priority groups. The study concludes that the priority groups' technique is the worst approach: it is quite slow to complete and hard to use. Furthermore, the priority groups got the lowest ranking when studying ease of use, reliability and

fault tolerance. Based on this study, the priority groups technique seems not to be suitable for prioritizing the small number of requirements.

3.2.2 Bubble Sort Technique

Bubble sort is one of the easiest and most primary methods for sorting elements with regard to a criterion that mentioned by (Mishra & Garg, 2009), and (Aho et al., 1983). Bubble sort technique is similar to AHP. Both techniques needed a number of pairwise comparisons, they need n x (n-1)/2 pair-wise comparisons. But, the decision maker in bubble sort has to decide which of the two requirements have higher priority, not to what extent as in AHP (Vestola, 2010), (Mishra & Garg, 2009), and (Pergher & Rossi, 2013).

Bubble Sort performs Requirement Prioritization on the following step (IEEE-STD,1998), and (Mishra & Garg, 2009):

1) **In preparation stage**, all requirements are outlined in a vector.
2) **In execution stage**, all the requirements are compared according to bubble sort algorithm to discover which requirement is the most important. If the lower requirement is more important than the higher one, exchange their positions.
3) **In presentation stage**, the sorted vector is outlined. Proces result is a vector where the initial order of the requirements has changed. Top of the vector will contain the least important requirement, while the bottom of the vector will contain the most important requirement is at the bottom of the vector.

The result of a bubble sort is ranked requirements based on their priority on the ordinal scale, the most important requirement is at the top of the vector, while the least important requirement is on the bottom of the column. a Bubble sort algorithm is illustrated in Figure 4.

Figure 4. Bubble Sort Technique

(Karlsson, 1996) are the only empirical of bubble sort techniques. In that study, they conclude that bubble sort was one of the best methods by comparing the time consumption and subjective measures. In time consumption terms, bubble sort was faster than AHP but slower than the minimal spanning tree. Bubble sort was the simplest method to use and produced both reliable and fault tolerant results. Bubble sort seems to be viable for a small number of requirements.

3.2.3 Binary Search Tree (BST) Technique

The binary search tree is one of the methods that using for sorting elements (Aho et al., 1983). There are two children at most of each node in a binary search tree. (Karlsson et al., 1998) and (Karlsson et al., 2006) include this technique to the requirements prioritization field for ranking requirements. Binary search tree method is illustrated in Figure 5.

The main idea of binary search tree techniques is that each node outlines a requirement, all requirements in the left subtree of a parent have lower priority than their parent priority, and all requirements in the right subtree of a parent

have higher priority than their parent priority. When implementing the binary search tree method, we choose one requirement to be the top node. Then, we select one unsorted requirement to compare with the top node. If that requirement priority is lower than the top node, it searches the left subtree, but if that requirement priority is higher than the top node, it searches the right subtree. The process is repeated until no more node needs to be compared, then the requirement can be inserted into the right position. The average complexity for binary search tree is O (n log n).

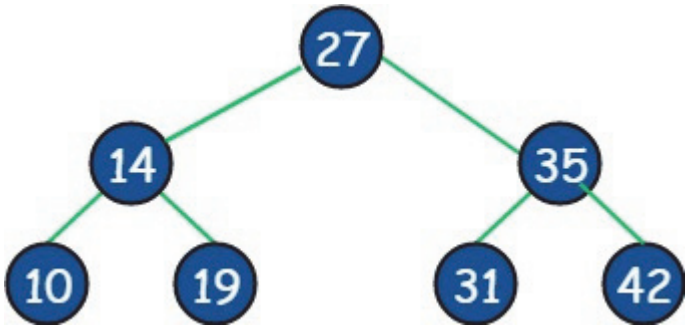

Figure 5. Binary Search Tree Technique

Binary search tree performs Requirement Prioritization by the following step (Mead, 2006):

1) **In the preparation stage**, outlined all candidate requirements of the system.
2) **In execution stage**, select one requirement at a time and a binary search tree is built.
3) **In presentation stage**, the binary search tree is traversed using in order traversing and nodes are attached to a list. The lowest priority requirements come first on the list. Then the list is printed.

3.3 Ratio Scale

Ratio scale prioritization techniques generate ranked lists of requirements. Ratio scale methods results can provide the relative difference between. Analytic hierarchy process (AHP), Hierarchy AHP, Minimal spanning tree, Cumulative voting (CV) and Hierarchical cumulative voting (HCV) are involved in this section.

3.3.1 Analytic Hierarchy Process (AHP) Technique

Analytic hierarchy process is the most popular requirements prioritization technique (Saaty, 1980) and it is created for complex decision making and may support the decision maker to set priorities and make the best decision (Golden et al., 1989).

First, AHP identifying the attributes and alternatives for every requirement and uses those to build a hierarchy to allow comparisons in a pair-wise fashion (Forman & Selly, 2001). Second, the users specify their favorite to every pair of the attributes by assigning a decision scale which is generally 1 to 9, where 1 means equal value and 9 means the farthest value. The scale is shown in Table 1. After that AHP converts the user's evaluations to numerical values and a numerical priority is determined for every element of the hierarchy. AHP is used to analyze software requirements and decide which one is the highest priority and to which degree; the number of pairwise comparisons will be n (n-1)/2 (Vestola, 2010), (Dabbagh & Lee, 2014), (Wang & Yang, 2012). Therefore, the complexity of AHP is O (n^2) (Kousalya et al., 2012), (Ma, 2009).

Table 1. Fundamental scale used for AHP (Karlsson, 1996).

How Important	**Description**
1	Equal importance
3	Moderate difference in importance
5	Essential difference in importance
7	Major difference in importance
9	Extreme difference in importance
Reciprocals	Reciprocals If requirement i has one of the above numbers assigned to it when compared with requirement j, then j has the reciprocal value when compared with i.

AHP perform Requirement Prioritization in the following step (Ma, 2009):

1) **In the preparation stage**, it outlined all unique pairs of requirements.
2) **In execution stage**, all outlined pairs of requirements compared by applying the AHP scale.
3) **In the presentation stage**, the relative priority of each requirement is estimated, then it calculated the consistency ratio of the pair-wise comparisons. Methods that mentioned above was used to calculate the consistency ratio of the pair-wise comparisons. The consistency ratio is an indicator of the reliability of the resulting priorities, and therefore also an estimate of the judgmental errors in the pair-wise comparisons.

AHP is divided into three different stages, based on the example that illustrated in Figure 6:

1) Create an $n \times n$ matrix where n is the number of requirements and insert n requirements in the rows and columns of the matrix.
2) For every unique pair of requirements, for example, A and B, insert their relative intensity of importance in the position where the row of SR-1 meets the column of SR-2. At the same time, the reciprocal values are inserted d in the transposed positions (e.g. if cell $SR-1/SR-2=8$ then cell $SR-2/SR-1=1/8$).
3) Finally, to notice the relative priority of every requirement, the eigenvalues of the resulting comparison matrix are calculated. The final result is the relative priorities of the requirements.

	SR-1	SR-2	SR-3	SR-4	SR-5
SR-1	1	8	1/5	3	1
SR-2	1/8	1	1/5	1/7	1/7
SR-3	5	5	1	1	2
SR-4	1/3	7	1	1	1/2
SR-5	1	7	1/2	2	1

Figure 6. An example matrix created with AHP (Ma, 2009).

(Karlsson et al., 1998), and (Karlsson & Ryan, 1997) empirical studies confirm that AHP is time-consuming. Some techniques try to reduce the number of comparisons to reduce the time consumed. Hierarchy AHP and minimal spanning tree have been generated for that purpose.

3.3.2 Hundred Dollar (100$) Technique

Cumulative Voting (CV) which is another name of hundred-dollar method that was proposed by (Berander, & Andrews, 2005), and (Hatton, 2007). Hundred-dollar is considered as a simple and straight forward technique for preferring requirements. The main idea of this technique is that stakeholder who has participation for prioritization are granted constant number of specific unit (such as 100 dollars, 1000 points) that the stakeholders asked to assume their unit to distribute to the requirements. Thus, the number of unit that is given to the requirement represents the priority of this requirement. The outcomes are given on a ratio scale which can provide the information on how much one requirement.

This method has a drawback when there are too many requirements, then this method will not perform well, and prioritization miscalculated, and the points do not add up to 100. It can also be difficult to keep track of how much has been assigned and what amount is left to dispose of.

(Ahl, 2005) compares hundred dollars to other four prioritization techniques, which are binary search tree, AHP, Planning Game and Planning Game combined with AHP. The total numbers of requirements that are used to prioritize are 13. When preferring these requirements, the test subjects proved that the hundred-dollar method is a simple technique to use and is considered as one of the most precise technique. In addition, it is one of the fastest techniques; however, this technique is not suitable for processing huge number of requirements.

(Regnell et al., 2001) contacted that the stakeholders prioritized 58 requirements with imaginary $\$100{,}000$. This study promoted some solicitude about hundred-dollars. Firstly, an overview might lose by the stakeholders when the number of requirements grows. Secondly, hundred-dollars might be susceptible to so called “shrewd tactics”

which means that the points that is distributed to the requirements by the stakeholders depend on how do others think will proceed it to obtain high primacy to their preferable requirements.

3.3.3 Minimal Spanning Tree Technique

Minimal spanning tree is another technique for requirement prioritizations that is proposed by (Karlsson et al., 1998). Where the main idea of this method is that in case the decisions are made absolutely constant, the redundancy will not occur; thus, the number of rapprochement will decrease to only (n -1) where n indicates to the number of requirements. This means that to generate a minimal spanning tree in a directed graph is the least effort that requested by a decision maker. In the directed graph that can be built by the comparisons supplied, there is at least one path between the requirements not pair-wise compared (Karlsson et al., 1998). For instance, if there are three requirements A, B, and C; where A is defined to be a higher priority than B and B is defined to be a higher priority than C; thus, requirement B should be a higher priority compared to C.

Minimal spanning tree implements Requirement Prioritization on the following step (Vestola, 2010), (Ma, 2009):

1) **In the preparation stage**, $(n - 1)$ unique pairs of requirements are outlined to construct a minimal spanning tree.
2) **In execution stage**, all outlined pairs of requirements are compared using the scale in Table 1.
3) **In presentation stage**, the missing intensities of importance are computed by taking the geometric mean of the existing intensities of all possible ways in which they are connected. After that, AHP is used as usual.

The minimal spanning tree approach is supposed to be extremely fast as it significantly minimizes the number of pairwise comparisons. Furthermore, it is more sensitive to judgmental errors since all redundancy has been removed.

(Karlsson et al., 1998) deduced that the minimal spanning tree approach is considered as the fastest technique; however, this technique provided the minimum reliable outcomes and fault tolerance is powerless as measured by subjective measures from practitioners. According to this study, it demonstrates that the minimal spanning tree approach is not the most suitable technique for a software system that has a small number of requirements. This technique is fast; however, in case fault tolerance and reliability are more significant than time-consuming, better technique exist such as AHP. Because of better scalability, the minimal spanning tree approach is suitable for prioritizing a huge number of requirements. Further, no empirical studies prop this hypothesis.

3.4 Data Mining and Machine Learning Techniques

In requirements prioritization literature there are rarely found literatures about Data mining and Machine learning techniques for requirements prioritization process.

(Duan et al., 2009) proposed a prioritization method that employs data mining and machine learning techniques for requirements prioritization based on the business goals, stakeholder's concerns, and collaborative interests that effects directly on the overall project; such as security or performance requirements.

(Avesani et al., 2005) proposed a machine learning technique in order to cope with scalability problem; the proposed technique is case-based ranking framework. In experimental results in proposed work shows comparison be-tween their results with AHP technique in terms of two factors: the required efforts from experts and the accuracy of requirements prioritization process.

(Perini et al., 2013) They described the Case-Based Ranking (CBRank) requirements prioritization method in another viewpoint by make a combination between the preferences of project's stakeholders and the requirements ordering approximations by employing machine learning techniques. According to their results compared with AHP technique; they concluded that their approach performs better than AHP, and keep into ac-count the trade-off between the efforts of experts and the accuracy of prioritization process; since it similar as AHP technique.

(Tonella et al., 2013) proposed an Interactive Genetic Algorithm (IGA) for requirements prioritization, in their work they take into consideration the effectiveness, efficiency, and robustness to the errors that may occurs in decision making process. As they assessed their technique they concluded that the IGA performs better than Incomplete Analytic Hierarchy Process (IAHP).

4. Evaluation of Requirements Prioritization

4.1 Question Comparison

Based on the discussion given above for the 8 techniques, this study has compared all the techniques with five questions for each method. The result is presented in the table 3. The four questions are organized as follow:

1) **The first question** that asked was "how easy they thought that the method was?"

The result of that question is depending on the scale from 1 to 8 that used for the measurement, where 1 indicates the worst and 8 indicates the best. The result is presented in Figure 7.

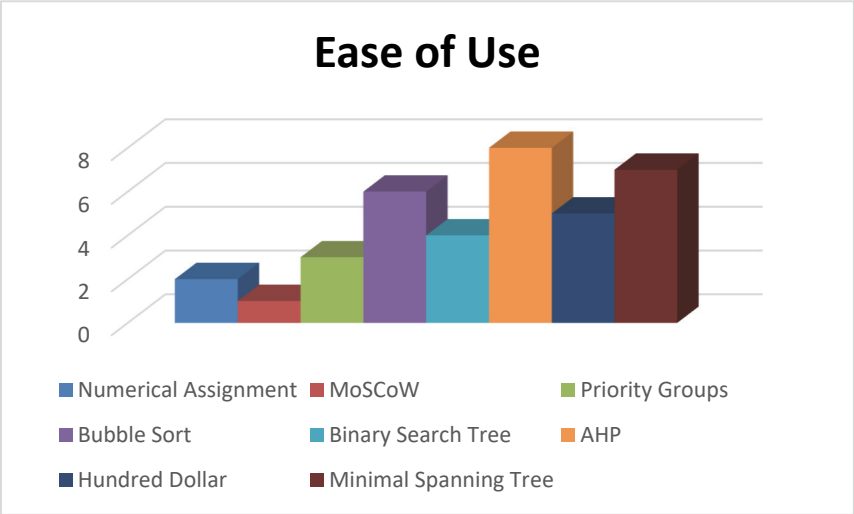

Figure 7. Ease of use comparison among the methods

2) **The second question** that asked was "how certain they were about the end result obtained from the methods under consideration?"

This question is depending on consistency of each method that we used. We answer this question with yes or no.

3) **The third question** that asked was "how long time it took to complete the prioritization process with the method under consideration?"

Also, the result of that question is depending on the scale from 1 to 8 that used for the speed measurement. The result is presented in Figure 8.

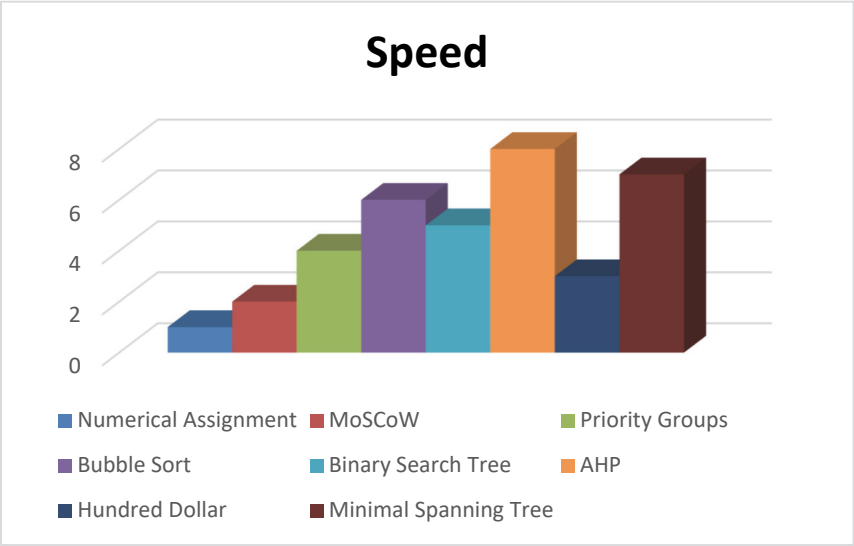

Figure 8. Speed comparison among the methods

4) **The fourth question** that asked was "how each method they believe that the methods would work with many more requirements?"

This question is depending on scalability of each method that we used. We answer this question with more scalable, not scalable, and medium scalable.

5) **The fifth question** that asked was "which of the method result is accurate than the other one?"

This question is depending on the accuracy of each method that we used. We answer this question with high accuracy, medium accuracy, and less accuracy.

Table 2. Question comparison of Requirements Prioritization Techniques

Methods	Q1	Q2	Q3	Q4	Q5
Numerical Assignment	2	Yes	1	Not Scalable	Less Accuracy
MoSCoW	1	Yes	2	Not Scalable	Less Accuracy
Priority Groups	3	Yes	4	More Scalable	Medium Accuracy
Bubble Sort	6	No	6	Not Scalable	Less Accuracy
Binary Search Tree	4	Yes	5	More Scalable	High Accuracy
AHP	8	Yes	8	Not Scalable	Medium Accuracy
Hundred Dollar	5	Yes	3	Medium Scalable	High Accuracy
Minimal Spanning Tree	7	No	7	Medium Scalable	Less Accuracy

Table 2 shows that priority group technique is the easiest method compared with the other approaches. After each requirement was placed into one of the three groups (i.e., high, medium, and low priority), classifying the requirements of each group on the basis of importance becomes easy and simple. Meanwhile, AHP is the most difficult method compared with the other approaches. AHP uses a pairwise comparison matrix to compute the cost or value of requirements that are relative to one another. Thus, AHP is difficult to use in a large number of requirements.

In comparison with the other methods, AHP requires the largest number of decisions and the longest computation time in terms of speed because it encounters problems in matrix computation time. Moreover, when n requirements need to be prioritized, $n * (n-1)/2$ pairwise comparisons are required for the AHP method; thus, solving this problem takes considerable time. Numerical method has the least number of computations compared with the other techniques.

All results obtained from the methods can be considered except for those from bubble sort and minimum spanning tree, which provided the least reliable results. Fault tolerance was also poor, especially when more requirements are added. Reliability and fault tolerance are more important than time consumption.

4.2 Techniques Scalabilities

The number of requirements for software projects varies according to the size, stakeholders, nature …etc. of software, and when talking about requirement prioritization the number of requirements play an important role in selecting the suitable technique for prioritization process. Since the size of the requirement sets varies according to each project the term scalability is used here to describe the ability of each technique to scale up with increasing number of requirements.

From this point and according to the previously discussed techniques for requirements prioritization the main factor to describe the scalability for each one is the complexity with respect to human efforts for each technique to conclude the comparison, three major classes will be used in order to describe the scalability factor: low, medium and high are used to make the comparison; since low means the technique is not suitable with increasing number of requirements, medium means its scalable with increasing number of requirements to some extents and becomes not scalable for high number of requirements and finally high means the technique is scalable with increasing number of requirements.

- **Numerical Assignment Technique**: since this technique is based on prioritizing the requirements into three groups: critical, standard, and optional, so here is simple complexity of this technique but it needs human effort to specify group for each requirement that means when number of requirement is increased this technique becomes a not scalable technique, so this technique will be classified as low scalability.
- **MoSCoW Technique**: this technique is based on collaboration between analysts and stakeholders to grouping the requirements in four groups: must have, should have, could have and won't have. Here the complexity is simple but needs human efforts with conflicts between analysts and stakeholder's viewpoints, so this technique will be classified as low scalability.

- **Priority Groups Technique**: this technique is similar to numerical assignment technique instead of repeat grouping recursively for each subgroup, for that this technique is complex and not easy to adopt it, so this technique will be classified as low scalability.
- **Bubble Sort Technique**: this technique employs the bubble sort algorithm for prioritizing the requirements according to pair-wise comparisons like AHP technique, but it is better than AHP because the decision maker was included to decide which the highest priority one from every two requirements. So, the complexity of this technique as we discussed above is $O(n^2)$ and less of human efforts so this technique will be classified as low scalability.
- **Binary Search Tree (BST) Technique**: this technique is based on binary search tree algorithm; it contains a tree of two leaves at most and each requirement will be added to tree after examining the prior position according to binary search tree structure. The complexity of this technique will be $O\ (n\ log\ n)$ as the binary search tree algorithm and it needs little human effort, for that this technique will be classified as high scalability.
- **Analytic Hierarchy Process (AHP) Technique**: this technique builds a hierarchy of attributes and alternatives for each requirement to allow pair-wise comparisons, and then preference scale will be built according to user preference decision. Since this technique will make pair-wise comparisons and create n*n matrix the complexity will be $O(n^2)$and medium range of human effort for building the preference scale, so this technique will be classified as low scalability.
- **Hundred Dollar (cumulative voting) Technique**: by its name $100\$$ this technique makes prioritization of requirements it gives the stakeholders a constant number of units like $100\$$ at let them distribute these values to requirements according to their priority then ratio scale will be created according to the given value. For that the complexity is simple but it needs big human efforts for creating the ratio scale and keeps tracking of the scale's values; so, this technique will be classified as low scalability.
- **Minimal Spanning Tree Technique**: in this technique the spanning tree architecture was adopted and each node represents requirements, the main aim in this technique is pair-wise comparison and reduce the number of comparisons by removing the redundancy between requirements so it's a very fast technique for prioritizing the requirements and on the other hand this technique did not need huge amount of human effort, but because of removing the redundancy it affects the reliability and the fault tolerance ratio it becomes not good technique for so this technique will be classified as medium scalability.

Table 3 shows the result of comparisons between the discussed 8 techniques and according to analyze each technique from complexity with respect to human efforts.

Table 3. Comparison of the effect of number of requirements in each technique

Methods	Scalability
Numerical Assignment	Low
MoSCoW	Low
Priority Groups	Low
Bubble Sort	Low
Binary Search Tree	High
AHP	Low
Hundred Dollar	Low
Minimal Spanning Tree	Medium

4.3 Stakeholders Comparison

Identifying the software project stakeholder at the beginning stage of software development process is important because the process begins by identifying stakeholders and their relationships and they assist software engineers to determine the issues from the earlier phase of development, through planning phase and at the execution of the project. Thus, stakeholders should be aware of what are project functions and deliverables, including the scope of the project and their milestones that lead to goals.

Often their various types of stakeholders in the project, the number of stakeholders plays an important role to consider the stress and complexity of the project. The stakeholder involvement in the project determines the degree of stakeholder's influence on the project outcomes. In addition to the number of stakeholders and their level of

investment, the stakeholder's agreements or disagreements during software development influence the project's complexity. There many types of stakeholders such as primary, secondary, direct, indirect, internal and external stakeholders.

Primary stakeholders have a major concern in the success of a project because they have direct influences on the project's outcome. Examples of primary stakeholders are Customers and end users are as well as some project sponsors, project managers, and team members. While the Secondary stakeholders assist to finish the project. Instead of their role aren't primary, they help with administrative processes, financial, and legalities. So, the communication between primary and secondary types of stakeholders will ensure that everyone is working toward the same goal. Thus, the lack of communication between them can lead to break-down the project.

Project managers are involved in project development directly, so they are internal stakeholders. They have authority to manage the project by making decisions and handling the responsibility of work performance, managing, and planning; in order to ensure accuracy and efficiently phases to be done.

On the other hand, the Vendors, suppliers, and outside organizations are external stakeholders because they supply the needs for a project's success, so they should be stay communicated at all times on goals, milestones and deliverables.

The day to day activities of a project are the major concerns of direct stakeholders, the team members are considered as direct stakeholders as their workloads are scheduled around the project each workday. Indirect stakeholders like customers and end users; are not affected directly by the project because those not affected, and their concern is with the finished project. This would be the quality of products, price, packaging, and availability.

Numerical Assignment Technique as discussed above is a base technique for requirement prioritization in which requirements are prioritized in three groups and then each requirement will be assigned to one of these groups according to their priority. Thus, this technique needs decision makers to specify a group for each requirement. In other words, the stakeholders play important role in this technique and they involved in a high level in the prioritization process. In addition to that, the same thing for MoSCoW technique is a special case of numerical assignment technique, but needs more effort in solving the conflicts between analysts and stakeholder's viewpoints to assign the group for each one; here 4 groups will be identified.

The idea behind priority groups technique is similar to numerical assignment technique by assigning each requirement into one of the three groups: high, medium and low priority. However, this priority technique works recursively to grouping each subgroup, so it needs the stakeholders with the complex operation, thus a high level of involvement is required.

The simple ranking, bubble sort, and binary search tree techniques are employed in order to rank the requirements according to bubble sort and binary search tree algorithms. The simple ranking method is well known for people. Therefore, these techniques need the stakeholders at the beginning to determine the highest priority of each requirement and prioritize all of them, so the middle level of involvement is needed.

As mentioned before, the base of the hundred-dollar technique is that each stakeholder is asked to assume a person has $\$100$ to spread it to the requirements according to their importance. A ratio scale is used to present the results. Therefore, the stakeholder is the most significant factor in this technique with less complexity.

In AHP technique a pair-wise comparison is the basic of this technique to create $n * n$ matrix that describes the decision makers preferences, so as well this technique is best suited for small size projects, the stakeholders are involved at beginning then the technique will continue to find the final prioritized list, thus means low level of involvement.

Minimal spanning technique is working like AHP in pair-wise comparisons manner but employing the minimal spanning tree architecture to minimize the total number of comparisons by removing the redundant comparisons, thus as AHP, it needs a low level of stakeholder involvement.

As discussed above, the data mining and machine learning techniques the role of stakeholders is essential and very important to accomplish the prioritization process, so these techniques have a high stakeholder's participation.

Table 5 shows the comparisons between the 8 techniques according to stakeholder's participation in prioritization process, these participations are categorized into three groups: low, medium and high participation.

Table 4. Comparison of stakeholder participation in each technique

Technique	Low	Medium	High
Numerical Assignment			✓
MoSCoW			✓
Priority Groups			✓
Bubble Sort		✓	
Binary Search Tree		✓	
AHP	✓		
Hundred Dollar			✓
Minimal Spanning Tree	✓		
Datamining and machine learning			✓

5. Nonfunctional Requirements Prioritization Approaches

In software projects, the specific software is utilized using both functional and nonfunctional characteristics that describe the nature and restrictions on the development process, such as performance, security and usability constraints, and expectations, these characteristics for both functional and nonfunctional should be taken into consideration in order to deliver high-quality software (Chung et al., 2009).

The term non-functional requirements describe the nature and limitations on the project instead of its functionality, also this term describes the non-behavior aspects and attributes of the system including usability, portability, security, understandability, reliability, and modifiability. In general, the non-functional requirements highlight the requirements that describe "how good" the software (Chung et al., 2009).

The non-functional requirements determine the success of project functionality specifically when the project delivered and put to use by customers, for example, the user will feel bad when system's functionality works well but with a low level of security or low speed. So, the non-functional requirements prioritization is a challenging phase because of some reasons (Chopra et al., 2016):

- Usually, the non-functional requirements prioritization is done by developers instead of users, since there is a high relationship between functional and non-functional requirements.
- Generally, the non-functional requirements are considered as a base of the system since they did not affect the system functionality, so fewer concerns spend on these requirements.
- Usually, the prioritization process concerns with functional requirements instead of non-functional or the non-functional usually take priority less than functional requirements.
- When non-functional requirements prioritized, they are prioritized separately from the functional requirements, but in reality, the prioritization process should be balanced between these types of requirements.

(Chopra et al., 2016) discussed and experiments three approaches to non-functional requirements prioritization according to two main objectives: accuracy of prioritization for each approach and the software complexity impacts on the prioritization accuracy.

A. Approach 1 (A1)

This approach makes a prioritization of both types of requirements functional and non-functional requirements since the two types will have prioritized together the functional requirements will gain the highest priority over the non-functional requirements, from this the non-functional will lose the competition with functional so this is not good.

B. Approach 2(A2)

In this approach the functional and non-functional requirements will be prioritized separately, so there is no competition between them and this a good approach; in the other hand almost, the non-functional prioritization process is done by software developers instead of users, so implicitly the selection of non-functional requirements depends in somehow to the functional requirements selection.

C. Approach 3(A3)

In this approach a combination of A1 and A2 will do in order to get best and more accurate results, in this approach the non-functional requirements will be considered separately, with keep in account the result of prioritizing the functional requirements, this leads to ignoring the competition between them and the selection process of them are independent, so the base of selection is based on functionality.

In order to measure the research objectives of in (Chopra et al., 2016), they employ the Analytical Hierarchal Process (AHP) based on cost-value prioritization technique on different three complexity versions of same industry software projects, the chosen technique to evaluate the proposed approaches is a pair-wise comparison prioritization technique and its approximately accurate technique that found in the literature by Karlsson (1996), Karlsson et al. (1998), Perini et al. (2009).

As the results that are introduced by (Chopra et al., 2016), approach 1 (A1) shows decreasing in accuracy when the project complexity is increasing, since the non-functional requirements ignored and lose competition; on the other hand, approach 2 (A2) shows better results on accuracy from A1, but the results show when project's complexity increased the accuracy decreased too because when the project has large number of requirements the execution process of prioritization become complex. And as approach 3 (A3) considers non-functional prioritization that high coupling with the result of functional requirements prioritization, the results shows remarkable improvements in the prioritization process.

As shown in (Chopra et al., 2016), the make of consideration by associate non-functional requirement prioritization with the result of functional requirements prioritization has a great effect and improvement in prioritization process instead of considering both types of requirements separately.

(Gupta et al., 2017) proposed non-functional requirements prioritization approach based on Chopra et al. (2016) work, the proposed approach is considered prioritization of functional and non-functional requirements separately using AHP technique with considering aspects that are mandatory for the organization like business values, cost and time. The selection and re-prioritization of non-functional requirements are done according to their impact and dependencies with highest functional priority.

The proposed approach is working by firstly prioritize the requirements separately, then and based on dependencies of non-functional requirement with the highest priority functional with respect of usage count from history to find the highest priority non-functional requirements that related with functional requirements (Gupta et al., 2017).

As the results of (Gupta et al., 2017) work, they conclude that there is a high coupling relationship between functional and non-functional requirements in requirements prioritization process that totally affect the success and usability of the software projects, since the author uses AHP technique to evaluate the proposed approach, when the project size go larger that means huge number of requirements will put question marks on approach efficiency.

(Dabbagh & Lee, 2014) proposed a prioritization technique to prioritize functional and non-functional requirements simultaneously by using one decision matrix based on finding the effect of non-functional for a specific functional requirement by assigning value like importance, the result of the proposed work is a prioritize list of functional requirements with considering the effect of non-functional requirements.

- **Integrated Prioritization Approach (IPA).**

This proposed approach by (Dabbagh & Lee, 2014) is consists of 5 steps in order to generate a decision matrix that relates functional to non-functional requirements, first elicit functional and functional requirements and create an n*m matrix where n and m represent functional and non-functional requirements respectively; then the importance degree will be assigned to each non-functional requirement with respect to relationship with functional requirements by involving decision makers in this step; next step will be based on fuzzy numbers and alpha cut approach in order to specify the rank of non-functional requirement prioritization with consider the functional requirements and finally calculating the functional requirements priority vector and from this step the final ranking list will be produced.

The authors in (Dabbagh & Lee, 2014) compared their proposed approach with AHP and hybrid assessment method (HAM) in term of time-consuming for prioritizing set of requirements. As the results that shown in their work, the proposed approach outperforms both AHP and HAM in term of time-consuming, with showing that almost similar of prioritization agreement from decision makers.

From the previous studies that concern in integrating both functional and non-functional requirements in prioritization process and highlights the importance of involvement both types of requirements and their importance in the success of software projects and their effect on software quality.

We conclude that there is high coupling relationship between functional and non-functional requirements in term of delivering high-quality software for both functionalities of software and satisfying the restrictions and limitation on the project as non-functional requirements. From this point and according to researcher's experiments, the accuracy and user satisfaction will be increased, and the restriction and limitation will be kept in all software releases delivery.

6. Advantages and Disadvantages for Each Technique

According to the discussion about requirements prioritization techniques, each technique has advantages and disadvantages for complexity speed, the reliability of results, requirements size and easy to use.

Table 6 shows the concluded advantages and disadvantages for each technique that we discussed in this research.

Table 5. Advantages and Disadvantages for Each Technique

Methods	Advantages	Disadvantages
Numerical Assignment	Low complexity with High speed Cope with large size of requirements Easy to use	Low rate of reliability Low level of fault tolerance
MoSCoW	Easy to use for small size of requirements Medium level of Reliability High level of fault tolerant	Medium complexity with acceptable speed according to requirements size Not scalable for medium to large size of requirements
Priority Groups	Easy to use Reliable results High level of fault tolerant	Medium complexity with acceptable speed in some cases Not scalable for medium to large size of requirements
Bubble Sort	Good in small size of requirements	Not easy to use High complexity with low speed Not reliable results Low level of fault tolerant Not cope and become very complex with medium to large size of requirements
Binary Search Tree	Easy to use Reliable results High level of fault tolerant	Medium complexity with medium speed Not cope very well for medium to large size of requirements
AHP	Excellent in small size of requirements	Not easy to use High complexity with low speed Not reliable results Low level of fault tolerant Not cope and become inefficient with medium to large size of requirements
Hundred Dollar	Low complexity with high speed Easy to use Excellent in small size of requirements and good for medium size of requirements	Medium level of reliability medium level of fault tolerant Not cope with large size of requirements
Minimal Spanning Tree	Low complexity with acceptable speed Easy to use Reliable results High level of fault tolerant Excellent in small to medium size of requirements	Not efficient with large size of requirements
Data mining and machine	Reliable results	Complex with medium speed

learning	High level of fault tolerant Cope with all size of requirements	Not easy to adopt
Integration of functional and non-functional requirements	High quality software with more respect of restrictions and limitation. Reliable High accuracy results	Time consuming Additional complexity to involve non-functional requirements in prioritization process

7. The Suggested General Model for Best Suited Technique

As a result of this study, and after making comparisons between the highlighted prioritization techniques with general factors that the decision makers concern in step to adopt a prioritization technique as a base for requirement prioritization that is suited for a specific project with respect to its properties. General Model is proposed in this study in order to aid the decision makers to choose the best-suited prioritization technique according to the project properties.

The proposed model is concerned with general factors to specify the best-suited technique and clearly shown in table 7; these factors are:

- **Ease of use**: these values are in range between 1 and 8 as discussed previously, the smallest value means the high degree of ease of use and large value means low degree of ease of use
- **The speed of showing results**: in another word the technique's complexity and its given values from 1 to 8 since the smallest number means high speed to get results and vice versa.
- **Requirement set size**: it represents the size of requirements set for the project and the given values are small, medium and large.
- **Accuracy**: the degree of accuracy of results, these values are less accuracy, medium, and high accuracy according to the discussion above.
- **Stakeholder involvement**: this factor represents the participation level of stakeholders in prioritization process, the given values are low. Medium and high level.

Table 6. General Model matrix

General Factors	Ease to use	Speed	Requirement set size (Scalability)	Accuracy	Stakeholders involvement
Numerical Assignment	2	1	Small (low)	less	high
MoSCoW	1	2	small, medium (low)	less	high
Priority Groups	3	4	Small (low)	medium	high
Bubble Sort	6	6	small, medium (low)	less	medium
Binary Search Tree	4	5	small, medium, large (high)	high	medium
AHP	8	8	Small (low)	medium	low
Hundred Dollar	5	3	Small (low)	high	high
Minimal Spanning Tree	7	7	medium, large (Medium)	less	low
Data Mining	8	7	medium, large (Medium)	high	medium

According to table 7 the user can specify their concern factors in the project and the model suggests the best-suited technique for that project, and for dynamicity in the proposed model; the user can specify importance weight for each factor, and for static valued factors the input is as a list of these values.

This model calculates the overall scores for all techniques based on user selections and weights using Eq. (1):

$$\boldsymbol{Technique\ score\ = \sum\_(k=1)\hat{}n\text{▒}(Weight * N\_val)k} \quad (1)$$

Since the weight is the user weight a N_val is normalized value for the cell, since in ease of use factor the small number represents a high level of ease of use the normalized value is calculated by subtracting it from one after dividing it by 10 for normalization issues.

And after each technique gets its score based on equation 1, the technique with the highest value of score will be suggested as a best-suited technique for that project, equation 2 represents the selection process of best-suited

technique.

$$\textbf{Suggested technique} = MAX(\textbf{Technique1 scores}, \dots, \textbf{TechniqueN scores}) \quad (2)$$

Where the technique score is calculated using equation 1, and N represents the number of techniques.

Figure 9 shows an example of the proposed model, the user chooses the weight of 70% to ease of use, 75% for speed, large requirement size with high accuracy and medium stakeholder participation. And according to model calculations, it suggests the Binary Search Tree as best suited technique for this project with a weight of 3.795.

A	B	C	D	E	F
	easy to use	speed	req size	accuracy	stakeholders
Numerical Assignment	2	1	small	less	high
MoSCoW	1	2	small,medium	less	high
Priority Groups	3	4	small	meduim	high
Bubble Sort	6	6	small,medium	less	meduim
Binary Search Tree	4	5	small,medium,large	high	meduim
AHP	8	8	small	meduim	low
Hundred Dollar	5	3	small	high	high
Minimal Spanning Tree	7	7	meduim,large	less	low
data mining	8	7	meduim,large	high	meduim
user input	0.5	1	medium	high	meduim
			Suggested technique	weight	
			Binary Search Tree	3.8	

Figure 9. Example of proposed general model

8. Conclusion

This study presented the most popular techniques for requirements prioritization and their corresponding literature. In this paper, the comparisons are based on three criteria; first criteria based on some questions that related to the ease of use, scalability, consistency, accuracy, and the speed for each one of the methods. Second criteria based on stakeholder's participation in the prioritization process. Third criteria based on the number of requirements for different software projects. And according to of the literature, we conclude the importance of involvement the non-functional requirements in prioritization process and their effects on project success and quality of the software.

In this study general model was proposed to aid the decision makers to choose the best-suited technique for prioritizing the software requirements, additional techniques can be added to this model after studying and reviewing their properties and decide the general factors for each one.

References

Ahl, V. (2005). An experimental comparison of five prioritization methods - investigating ease of use, accuracy and scalability. Master's thesis, Blekinge Institute of Technology, Ronneby, Sweden, 2005.

Aho, A. V., Hopcroft, J. E., & Ullman, J. D. (1983). Data structures and algorithms. Reading, MA: Addison-Wesley.

Avesani, P., Bazzanella, C., Perini, A., & Susi, A. (2005, August). *Facing scalability issues in requirements prioritization with machine learning techniques.* In Requirements Engineering, 2005. Proceedings. 13th IEEE International Conference on (pp. 297-305). IEEE.

Berander, P., & Andrews, A. (2005). Requirements prioritization. In A. Aurum & C. Wohlin (Eds.), Engineering and managing software requirements (pp. 69-94): Springer Berlin Heidelberg.

Bradner, S. (1997). Key words for use in RFCs to indicate requirement levels. RFC 2119.

Chung, L., Leite, P., & Cesar, J. (2009). On non-functional requirements in software engineering. Conceptual Modeling: Foundations and Applications, pp. 363–379, Springer, 2009.

Dabbagh, M., & Lee, S. P. (2014). An approach for integrating the prioritization of functional and nonfunctional requirements. *The Scientific World Journal, 2014.*

Duan, C., Laurent, P., Cleland-Huang, J., & Kwiatkowski, C. (2009). Towards automated requirements prioritization and triage. *Requirements Engineering, 14*(2), 73-89.

Forman, E. H., & Selly, M. A. (2001). The Analytic Hierarchy Process and Expert Choice. World Scientific Book Chapters, 43-125.

Golden, B. L., Wasil, E. A., & Harker, P. T. (1989). The analytic hierarchy process. Applications and Studies, Berlin, Heidelberg.

Greer, D., & Bustard, D. W. (1997). SERUM-Software engineering risk: Understanding and management. The *International Journal of Project & Business Risk, 1*(4), 373-388.

Hatton, S. (2007). Early prioritisation of goals. In Advances in conceptual modeling – Foundations and applications (pp. 235-244).

Hudaib, A., Qasem, M. H., & Obeid, N. (2017, September). FIPA-Based Semi-centralized Protocol for Negotiation. In Proceedings of the Computational Methods in Systems and Software (pp. 135-149). Springer, Cham.

Ibrahim, O., & Nosseir, A. (2016). A Combined AHP and Source of Power Schemes for Prioritising Requirements Applied on a Human Resources. In MATEC Web of Conferences (Vol. 76, p. 04016). EDP Sciences.

IEEE-STD 830-1998. (1998). IEEE recommended practice for software requirements specifications. IEEE Computer Society.

Kadhum, M., Qasem, M. H., Sleit, A., & Sharieh, A. (2017, April). Efficient MapReduce Matrix Multiplication with Optimized Mapper Set. In Computer Science On-line Conference (pp. 186-196). Springer, Cham.

Karlsson, (1996). Software requirements prioritizing. in Requirements Engineering. Proceedings of the Second International Conference on, 15-18, 110 –116.

Karlsson, J., & Ryan, K. (1997). A Cost-Value approach for prioritizing requirements. *IEEE Software, 14*(5), 67-74.

Karlsson, J., Wohlin, C., & Regnell, B. (1998). An evaluation of methods for prioritizing software requirements. *Information and Software Technology, 39*(14-15), 939-947.

Karlsson, L., Host, M., & Regnell, B. (2006). Evaluating the practical use of different measurement scales in requirements prioritisation. Proceedings of the 2006 ACM/IEEE International Symposium on Empirical Software Engineering (ISESE'06), 326-335.

Kousalya, P., Reddy, G. M., Supraja, S., & Prasad, V. S. (2012). Analytical Hierarchy Process approach–An application of engineering education. *Mathematica Aeterna, 10*, 861.

Leffingwell, D., & Widrig, D. (2000). Managing software requirements - A unified approach. Upper Saddle River: Addison-Wesley.

Ma, Q. (2009). The effectiveness of requirements prioritization techniques for a medium to large number of requirements: A systematic literature review (Doctoral dissertation, Auckland University of Technology).

Masadeh, R., Alzaqebah, A., Hudaib, A., & Abdel Rahman, A. (2018). Grey Wolf Algorithm for Requirements Prioritization. *Modern Applied Science, 12*.

Mead, N. R. (2006). Requirements prioritization introduction. Software Eng. Inst. web pub., Carnegie Mellon Univ. Retrieved from http://www.dii.unisi.it/~mocenni/Note_AHP.pdf

Mishra, A. D., & Garg, D. G. (2009). Selection of best sorting algorithm for a particular problem (Doctoral dissertation).

Mohammad, D., & Lee, S. P. (2014). An Approach for Integrating the Prioritization of Functional and Nonfunctional Requirements. *Scientific World Journal, 2014*, 13 pages, Article ID 737626,

Pergher, M., & Rossi, B. (2013, July). Requirements prioritization in software engineering: a systematic mapping study. In Empirical Requirements Engineering (EmpiRE), 2013 IEEE Third International Workshop on (pp. 40-44). IEEE

Perini, A., Susi, A., & Avesani, P. (2013). A machine learning approach to software requirements prioritization. *IEEE Transactions on Software Engineering, 39*(4), 445-461.

Qaddoura, R., Abu-Srhan, A., Qasem, M. H., & Hudaib, A. (2017). Requirements Prioritization Techniques Review and Analysis. 2017 International Conference on New Trends in Computing Sciences (ICTCS). https://doi.org/10.1109/ictcs.2017.55

Qasem, M. H., & Qatawneh, M. (2017, September). Parallel Matrix Multiplication for Business Applications. In Proceedings of the Computational Methods in Systems and Software (pp. 24-36). Springer, Cham.

Qasem, M. H., Al Assaf, M. M., & Rodan, A. (2015). Data mining approach for commercial data classification and migration in hybrid storage systems (Doctoral dissertation, The University of Jordan).

Qasem, M. H., Faris, H., Rodan, A., & Sheta, A. (2017, April). Empirical Evaluation of the Cycle Reservoir with Regular Jumps for Time Series Forecasting: A Comparison Study. In Computer Science On-line Conference (pp. 115-124). Springer, Cham.

Raj, K. C., Varun, G., & Durg, S. C. (2016). Experimentation on accuracy of nonfunctional requirement prioritization approaches for different complexity projects. *Perspectives in Science, 8*, 79-82. Elsevir.

Regnell, B., H¨ost, M., och Dag, J. N., Beremark, P., & Hjelm, T. (2001). An industrial case study on distributed prioritisation in market-driven requirements engineering for packaged software. *Requirements Engineering, 6*(1), 51–62.

Saaty, T. L. (1980). The analytic hierarchy process. McGraw-Hill, New York.

Sommerville, I., & Sawyer, P. (1997). Requirements engineering - A good practice guide. Chichester: John Wiley and Sons.

Svensson, R. B., Gorschek, T., Regnell, B., Torkar, R., Shahrokni, A., Feldt, R., & Aurum, A. (2011, August). Prioritization of quality requirements: State of practice in eleven companies. In Requirements Engineering Conference (RE), 2011 19th IEEE International (pp. 69-78). IEEE.

Tonella, P., Susi, A., & Palma, F. (2013). Interactive requirements prioritization using a genetic algorithm. *Information and software technology, 55*(1), 173-187.

Tudor, D., & Walter, G. A. (2006). Using an agile approach in a large, traditional organisation. Proceedings of AGILE 2006 Conference (AGILE'06), 367-373. Retrieved from https://blog.ganttpro.com/en/prioritization-techniques-and-methods-for-projects-with-advantages-of-moscow-model/

Varun, G., Shivam, L., Deniz, Ç., & Hye-jin, K. (2017). Non-functional Requirement Prioritization Approach. *International Journal of Software Engineering and Its Applications, 11*(1), 61-66.

Vestola, M. (2010). A comparison of nine basic techniques for requirements prioritization. Helsinki University of Technology.

Wang, Q., & Yang, Z. (2012). A method of selecting appropriate software architecture styles: quality Attributes and analytic hierarchy process.