MCBRank Method to Improve Software Requirements Prioritization

An Empirical Investigation

Sabrina Ahmad[1], Riftika Rizawanti[2], Terry Woodings[3], Intan Ermahani A. Jalil[4]
Fakulti Teknologi Maklumat Dan Komunikasi, Universiti Teknikal Malaysia Melaka, Melaka, Malaysia[1, 4]
Fakultas Teknologi Informasi dan Komunikasi, Universitas Teknologi Mataram, Nusa Tenggara Barat, Indonesia[2]
School of Physics, Math and Computing, The University of Western Australia, Perth, Australia[3]

***Abstract*—Software Requirements Prioritization is an important issue that has a more profound effect on the overall quality of software development. Application of software requirements prioritization provides benefits to minimize risks in software development so that the most important and most impactful requirements are given priority. This paper presents a proposed software requirements prioritization method named MCBRank, which incorporates renowned MoSCoW Method and Case-Based Ranking to improve prioritization correctness. It elaborates on the implementation of MCBRank in an empirical investigation to determine software requirements prioritization for a potential e-library system. The investigation allows the software requirements prioritization process to be implemented by using the proposed MCBRank method. A role-playing empirical investigation with 30 respondents prioritized 31 software requirements, and the results were measured by Cohen Kappa. The kappa results show that MCBRank achieves a better agreement towards the Gold Standard with kappa value of 0.60. Therefore, the investigation results support that MCBRank improves the importance of ranking correctness, representing the stakeholders' wants and the organization's actual needs for the potential e-library system.**

Keywords*—Requirements prioritization; requirements engineering; software engineering; empirical software engineering

I. Introduction

Requirements play an essential part in software development as they lay a foundation for a system to be developed [1]. Among many factors influencing the success or failure of software development, prioritization seems vital as it is responsible for determining the value of various requirements proposed by many stakeholders based on specific criteria defined from their usefulness for the final software product [2]. Thus, requirements prioritization is an essential part of requirements management. It is responsible for identifying the subset from many candidate requirements to maximize the fulfilment of various limitations such as resources availability, constrained timeline, and budget.

Thus, the main objective of the prioritization is to ensure that the customers get the software system that satisfies their needs with utmost value within limited resources. This is supported by a study [3] that stated Software Requirements Prioritization (SRP) could affect several factors in software development such as value, time to market, cost, and, most importantly, user satisfaction. Using SRP, the requirements will be prioritized based on human choices and by analyzing several factors that can be the reason for a product to fail or succeed.

The importance of having the correct prioritized requirements which fulfil the stakeholders' needs [4] and the difficulty of getting a good set of requirements that satisfy the business value motivate this research to propose an enhanced MoSCoW prioritization method called MCBRank. The proposed method incorporates renowned MoSCoW method and Case-Based Ranking method to improve the prioritization correctness. In this paper, an empirical method is adapted to evaluate the MCBRank effectiveness to prioritize software requirements. Therefore, this paper presents the empirical investigation of the MCBRank prioritization method based on a case study. The empirical result showed that the MCBRank method improves the software requirements prioritization correctness which represent the stakeholders' wants and the organization's actual needs for the potential e-library system.

Following the introduction, Section II presents the literature review. Next, Section III presents the proposed prioritization method, and Section IV explains the empirical investigation methodology. This is followed by Section V that discusses the empirical investigation results. Finally, Section VI concludes the paper.

II. Literature Review

Requirements prioritization refers to the process of choosing the correct set of specifications from a mass of overlapping and contradictory demands from various stakeholders involved in a software development project. Prioritization is one of the critical steps in system development to make meaningful decisions to scrutinize essential requirements for realization. This is supported by a study which stated that requirements prioritization has an enduring partnership with several other important technical practices such as interaction with the requirements analysis, requirement engineering, and general software engineering practices [5].

Gilb and Maier [6] stated that "priority is the relative right of a requirement to the usage of restricted (or scarce) capital." In this description, "capital" includes all kinds of capital, including time, money, and human capital. In other terms, we can have anything of the infinite system; then, there is no need to consider the requirements. However, ventures usually face finite capital such as short deadlines, restricted budgets, limited human capacity, and fixed technologies. Consequently, proposals sometimes include more requests than what can be executed in one release period. Thus, requirements prioritization enables the project planners to choose the final requirements to be implemented under resource constraints.

Various requirements prioritization techniques exist, but using the same technique to assign requirements for multiple stages may result in restricted benefits [7]. Different requirements prioritization techniques involve various properties, and therefore selecting the most optimal technique will optimize the gains gained at specific points [8]. These techniques enable the software development team to prioritize software requirements with high priority [9]. According to a study [5], there are many methods used to classify requirements. These prioritization methods may be classified as nominal scale, ordinal scale, ratio scale, machine learning, and hybridization, as shown in Fig. 1.

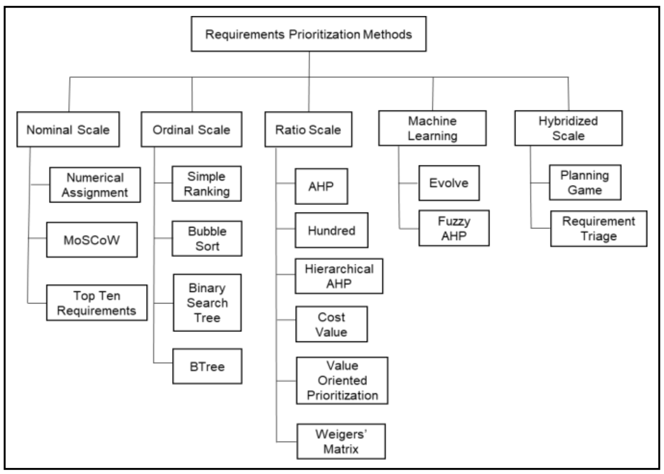

Fig. 1. Requirements Prioritization Method [19].

Other than methods, the elements being considered during the prioritization process are also essential. Hujainah et al. [10] provide a comprehensive investigation of the interrelated elements that need to be considered to formulate practical requirements prioritization technique. The findings reveal four interrelated elements that should be considered in developing RP techniques to secure an effective prioritization process. These elements are criteria for requirements, stakeholders, procedures, and implementation, which play important roles in prioritizing requirements. To incorporate the interrelated elements for prioritization, further research [11] proposed a semi-automated scalable prioritization technique to improve the prioritization efficiency for a large-scale project.

The fact that stakeholders' preferences need to be considered is also supported by Yaseen et al. [12]. It is noted that large numbers of requirements are likely to be developed based on the customers' preferences stated in the early stage of the requirements engineering phase. However, diverse stakeholders need to be considered since specific requirements are important for particular customers but not others [13]. Furthermore, customers may generally understand what they want but do not have a specific picture of precisely what is needed for a well-functioning software system. Therefore, requirements prioritization may assist the development team in shortlisting the requirements because essential requirements should be presented to the stakeholders as quickly as possible. In addition, should conflicting requirements surface, requirements prioritization can be performed to resolve the conflicts. This is also supported by Ma [7], who stated that requirement prioritization refers to the process of choosing the correct set of specifications from a multitude of overlapping and contradictory demands from various stakeholders involved in a software development project.

Achimugu et al. [14] performed a systematic review of 48 Software Requirements Prioritization methods and found that MoSCoW is the most cited and utilized prioritization method. Miranda [15] noted that the MoSCoW is a more straightforward method of obtaining information from customers, meaning that customers better understand what is being asked of them and thus provide the development team with more meaningful and valuable information. Moreover, the MoSCoW is suitable for iterative development such as "agile software development" [16, 17, 18], which allows collaborative requirements prioritization between stakeholders [19]. This collaborative effort provides the customers with a product of a maximized business value [20]. On the other hand, the MoSCoW needs more time as detailed information is required to provide a relative value for each requirement. Besides MoSCoW technique is a numerical assignment technique that needs more effort to solve conflicts between analysis and stakeholders' viewpoints [21].

Meanwhile, Avesani et al. [22] proposed a framework called Case-Based Ranking that can reduce the acquisition effort by combining human reference elicitation and automatic preference approximation. The result shows improvement in requirement prioritization accuracy and a trade-off between experts' elicitation efforts. During the requirements analysis process on deciding which requirements to develop, different methods are used to select the correct requirements due to the system development team's preferences and work nature.

While MoSCoW and CBRank have received a great deal of attention in the literature, MCBRank method is designed to emphasize the strength of both techniques to improve software requirements prioritization.

III. THE PROPOSED METHOD

According to Yaseen et al. [23], it is necessary to recognize requirements' importance and priority to assist the developers in expediting the system development process. MCBRank enables two-layer prioritization to improve the correctness in terms of importance ranking. In this research design, the importance is represented by the key stakeholders' cumulative needs for the best of the system to be developed. Fig. 2 gives an overview of the MCBRank. Firstly, all candidate requirements which come from multiple stakeholders are listed. Then the key stakeholders are required to classify each requirement based on the adapted MoSCoW method on five points scale. The MoSCoW classification of M (Must have), S (Should have), C (Could have), and W (Would have) are assigned with numbers as listed below. 'Must not have this' is added into the scale to allow stakeholders to indicate the requirements they do not want to be realized.

5 - Must have this.

4 - Should have this if at all possible.

3 - Could have this if it does not affect anything else.

2 -Will not have this time but would like in the future.

1 - Must not have this.

Following that, the classification score based on the scale will be used to classify all the requirements into M (Must have), S (Should have), C (Could have), and W (Would have). At this point, the unwanted requirements will be discarded.

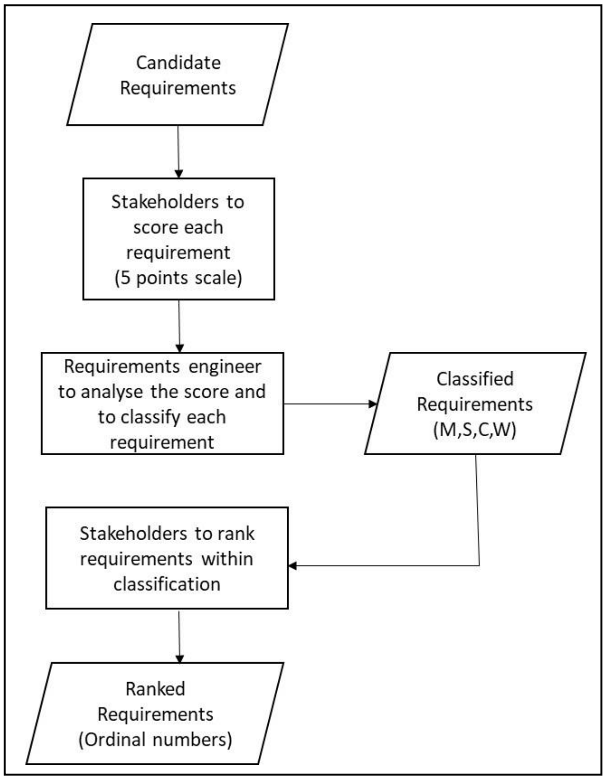

Fig. 2. Overview of MCBRank.

Next, within the classification, each requirement will be ranked using ordinal numbers. The majority rank will be the position of the requirement. If the majority rank is the same for two or more requirements, a smaller accumulative score given by the participating stakeholders will be in a higher rank. Finally, a new ranked requirements list is produced.

This ranking allows stakeholders to prioritize the requirements, delineate and narrow the scope of work to acquire focus. Priorities make it possible to measure how important the stakeholders feel about each requirement concerning a software solution to meet their needs. The prioritized requirements successfully narrow the focus, which helps in group agreement. Through priority [12], if it is impossible to develop all project requirements, it is feasible to discriminate the most critical requirements to the stakeholders. This means that a project that does not meet all of its requirements can still be of high value if it meets the stakeholders' most important requirements.

A. *An Example*

Assume that there are ten requirements proposed by three key stakeholders numbered as Req. 1 until Req. 10.

Step 1: The key stakeholders to score the listed requirements based on importance (5 points scale).

Requirements	**Stakeholder 1**	**Stakeholder 2**	**Stakeholder 3**
Req 1	5	5	4
Req 2	4	4	5
Req 3	5	4	5
Req 4	5	3	5
Req 5	3	3	4
Req 6	4	5	5
Req 7	5	2	1
Req 8	3	4	3
Req 9	2	3	3
Req 10	1	2	1

Step 2: The requirements engineer to analyze the score and to classify each requirement. In order to do this, the requirements engineer need to analyze the requirements and take into consideration the technical knowledge on top of just importance classified by the stakeholders. This is required to ensure the essential services for the said system is not neglected.

Req.	**Stakeholder 1**	**Stakeholder 2**	**Stakeholder 3**	**Group**
Req 1	5	5	4	M
Req 2	4	4	5	S
Req 3	5	4	5	M
Req 4	5	3	5	M
Req 5	3	4	4	S
Req 6	4	5	5	M
Req 7	5	2	1	W
Req 8	3	4	3	C
Req 9	2	3	3	C
Req 10	1	2	1	discard

Step 3: Within each classification group, the key stakeholders to rank requirements based on ordinal numbers. The majority will be the rank of the requirements. Note that, the ordinal number will follow through based on classification group priority. For example, Req 2 is ranked as 1 within Group S, then the number must be after the last number in the group M. Therefore, Req 2 is ranked as 5.

Group	**Req**	**S 1**	**S 2**	**S 3**	**Rank**
M	Req 1	2	1	2	2
	Req 3	1	2	1	1
	Req 4	3	4	3	3
	Req 6	4	3	4	4
S	Req 2	1	2	1	5
	Req 5	2	1	2	6
C	Req 8	2	2	1	8
	Req 9	1	1	2	7
W	Req 7	1	1	1	9
-	Req 10	-	-	-	-

IV. Empirical Investigation Methodology

An empirical investigation method was carried out to evaluate the effectiveness of the proposed MCBRank Method. It is a role-playing investigation to test if the requirements prioritization is improved following the implementation of MCBRank.

A. The Underlying Concept

The underlying concept includes an explanation and rationale for the terms and approaches applied.

- Stakeholders are terms that refer to any person or group directly or indirectly affected by the system. Stakeholders include end-users who interact with the system and everyone in the organization who may be affected by its installation. Other system stakeholders may be engineers who develop or maintain related systems, business managers, domain experts, and union representatives. These stakeholders are the key people representing the interests of their group. They may include end-users, system owners, and managers who work together and are actively involved in decision-making to reach mutually satisfactory agreements. However, it is neither appropriate nor possible to have all system stakeholders in acquiring requirements. Therefore, during the role-play investigation, this project will involve key stakeholders to represent users and administrators.

- The Gold Standard (GS) is a term used to describe the theoretical idea of the best requirements set that can exist for a system. In this requirements prioritization study, the 'ideal' set represents the right ranked requirements based on importance, representing the cumulative key stakeholders' needs and the actual need of the organization. This is because a sound system should fulfill what the stakeholders want and what the customers describe is what they want and what the organization needs from the system to be developed. However, there is no way to test this benefit in the short term. That is why this study presents the GS to represent a set of right ranked requirements based on importance from the domain experts' point of view. The domain experts are familiar with the crucial need of the system to fit the system's purposes of existence and understand the system's business value. The GS is determined by the candidate requirements and the collaborative efforts of researchers and several domain experts in the field. The experts are library practitioners ranging from thirty-five to fifty years old, with more than ten years of experience working in the library for Universiti Teknikal Malaysia Melaka. It is established as a benchmark for the best possible requirements with the correct importance rank for the particular case study; e-library system. The results from the investigations that will be carried out will be measured against the GS.

B. Gold Standard Formulation

This subsection describes how the researcher obtained the Gold Standard (GS). The GS is an ideal set of ranked requirements based on importance from the point of view of domain experts. The GS is obtained through an expert judgment technique that involves a group of domain experts. GS is a term used to describe the theoretical idea of the ideal set of well-ranked requirements based on importance for a system. GS was developed carefully by identifying some requirements that contain all the primary requirements required for the system. Researchers are aware that it is impossible to find a precise solution, but estimates can be determined based on the analysis of several experts.

Subjects involved in determining GS are experts who are library practitioners with criteria aged 30-50 years, have more than ten years of experience in related fields, and come from the Universiti Teknikal Malaysia Melaka. The procedures carried out are:

- In the first step, the researcher prepares a set of instruments containing an introduction, instructions, e-library scenario description, and a decision form that contains 31 candidate requirements to collect expert opinions.

- In the second step, screening and determining experts based on predetermined demographic components were conducted. This study involved six experts in determining GS.

- In the third step, the experts were given three days to examine the GS determination instrument to get familiar with the introduction, instructions, descriptive scenario, and a list of candidate requirements. Experts will get an overview of the background system from which the requirements were obtained through descriptive scenarios. Then the experts were asked to provide a rating with a Likert scale of 1-5 on the decision form. The scale of the assessment technique used in this study was adapted from the MoSCoW method.

Following the three steps above, the researcher gets the results of the requirements that have been ranked according to the highest to lowest rated. These properly ranked requirements are determined as GS. The value of each requirement is obtained from the total rate given by the six experts. Sorting has no problem when a requirement has different total values and does not get an equal value. For the first and second ranks, there were no obstacles. As for rank three and four, the total value for requirements R6 and R29 are the same, but since R29 has a higher number of five-point rating which means that more people rate R29 as 'Must have, R29 is placed in the third rank followed by R6 in the fourth rank. This also applied to the requirements R7, R5, and R22; R9 and R27; R17 and R23. As for requirements R26 and R12, which have the same total value and the same number of five-point ratings, R26 was decided to be in a lower rank than R12 because there exists a lower point rating for R26. Fundamentally, for the rest of the requirements, if the total value is the same and if the five-point rating is the same, then a lower point rating, if it exists for one of the requirements, will determine the ranking of the said requirements.

The formation of the GS in this investigation was done carefully since the GS represents an ideal set of requirements with the right importance ranking (theoretically represents both the stakeholders' wants and the organization's needs). The right rank of importance will determine the most crucial requirements to be realized to reduce the timeline or budget shortage. Thus, the output of the MCBRank will be measured against the GS to evaluate its effectiveness.

C. Investigation Method

1) Investigation materials: The candidate requirements are derived from the e-Library descriptive scenario. Ideally, the candidate requirements are proposed by independent stakeholders (or by a group of stakeholders) [24], represented by participants who play roles as a system's stakeholders in this investigation. However, the investigation focuses on the requirements prioritization effort with the proposed method and the researchers' candidate requirements prepared in advance. Requirements are built on the experience of researchers in using the system. They are improved by academics who previously have years of experience in developing systems and working on requirements. For prioritization purposes, candidate requirements are constructed to represent the interests and needs of various stakeholders. Candidate requirements include essential system features and additional features with a mix of quality values. In addition, the requirements also meet SMART concept; (S)pecific, (M)easurable, (A)greed, (R)ealistic, and (T)ime-bonded [3]. The number of candidate requirements is balanced and adjusted to the interests of specific key stakeholders. This is generally based on the purpose of the investigation. The number of candidate requirements was set for the experiment considering its time to rank effectively.

2) Population and sample: The investigation was carried out with 30 people, and the role-playing technique was exercised. Each was given a role as a user (students, lecturers, university staff) and administrators (librarians). Participants who carried out the investigation had a background in software engineering knowledge and were between 20 to 30 years old. Several of them have work experience in software development. To avoid bias, the roles they played in the investigation were randomized. Also, role-playing investigations always attend to whether the participants are actually playing a role or are merely including their judgments. It is expected that each participant will be more committed to individual priorities. Participants are given an exact role to minimize that possibility and given instructions and guidelines on playing system stakeholders' positions. Since participants may also have limited knowledge to precisely rank all candidate requirements concerning their impact on the acceptance of the software system [19], scenario descriptions and candidate requirements have been given to them in advance. This reading material is helpful because the scenario description describes the system needs and concerns of different stakeholders. In addition, the candidate requirements are carefully tailored to the needs of specific stakeholders.

3) The investigation protocol: For investigation, participants were divided into two groups. Each group consisted of thirty participants to rank candidate requirements based on the role they were playing. One group was a control group to exercise the MoSCoW prioritization method, and the other group exercised the MCBRank prioritization method. The results of the investigation are a prioritized list of software requirements.

Initially, during the investigation process, all participants were given scenario descriptions of potential e-libraries. Furthermore, a briefing was conducted on the experimental background knowledge through online communication. This is followed by instructions supported by an overview of examples of step-by-step activities. The assignment of participants is tailored to the role that will be played during the investigation. The participants were given a day to understand the roles and descriptions of the scenario prepared for them. The next day, participants were given a google form link and asked to prioritize the candidate requirements based on the MoSCoW method and MCBRank method. The participants' prioritization effort was then collected and analyzed. When all the participants had completed the assignment, a feedback form was given to them to learn about the strengths and weaknesses of the investigation for future references.

D. Justification of Results Measures

A study [25] introduced the concept of a perfect set of requirements (here termed the 'Gold Standard'), and measurements of progress towards such an ideal are made. Cohen's Kappa [26] is used to measure agreement between two assessors. Cohen's Kappa is an inter-rater reliability index commonly used to measure the level of agreement between two sets of dichotomous ratings or scores. It is generally a more robust measure than simple percent agreement calculation since it considers the agreement occurring by chance. Here, Cohen's Kappa is used to measure the agreement between requirements prioritized using MCBRank and the Gold Standard. The two raters are the group exercising the MCBRank and the domain experts who provide the Gold Standard. Kappa values show the movement of the agreement towards the Gold Standard. The nearer the agreement to the Gold Standard, the better quality it is. Therefore, the suitability of this metric is discussed, and its relevance to this study is stated.

Each participant prioritizes each requirement by using the MoSCoW or MCBRank methods. The prioritized requirements resulting from both methods were evaluated against the GS using the Cohen Kappa. Cohen's Kappa is used to measure the distance or degree of agreement between the GS and a set of requirements obtained through the application of the MCBRank in the investigation. If the agreement between results from the MCBRank and GS is better than the agreement between MoSCoW and GS, then the MCBRank method has succeeded in improving the level of correctness in requirements prioritization.

V. Results and Discussion

The empirical investigation evaluates whether the enhanced MoSCoW method named MCBRank can improve software requirements prioritization in terms of the correct rank of importance. Role-playing technique as key stakeholders to the e-library system was exercised in this study where the participants were to prioritize the candidate requirements using the proposed method. The purpose of this study is to evaluate the MCBRank method.

A control group was formed to exercise the MoSCoW method to be compared with the proposed method (MCBRank). The results obtained from the MoSCoW method and MCBRank method were both calculated. The level of agreement of both methods with GS was calculated using the Cohen Kappa formula.

The formula to calculate Cohen's kappa for two raters is:

$$k = \frac{P_{o-} p_e}{1 - p_e}$$

where:

P_o = total number of requirements that have the same rank position in GS.

p_e = total number of expected similarities that would occur if the observers were statistically independent.

Cohen's kappa measures the agreement between two raters who classify several items into several mutually exclusive categories. Landis and Koch [27] mention that labels with appropriate kappa ranges should be used to maintain consistent terminology when describing the relative agreement strength associated with kappa statistics. Table I presents the agreement strength based on Cohen Kappa.

The prioritized requirements obtained from the MoSCoW method were calculated, and the agreement to the GS is presented in Table II.

Based on the guideline adapted from Landis & Koch [27], a kappa (κ) of .067 represents a poor strength of agreement. Furthermore, since p = .042 (which means p > .0005), our kappa (κ) coefficient is not statistically significant. Meanwhile, Table III presents the measure of agreement between prioritization obtained from the MCBRank method and GS.

Meanwhile, a kappa (κ) of .600 represents a moderate strength of agreement. Furthermore, since p = .000 (which means p < .0005), our kappa (κ) coefficient is statistically significant.

Kappa values show the movement of the agreement towards the Gold Standard. The nearer the agreement to the Gold Standard, the better quality it is. The investigation was carried out to determine whether there was an agreement between the prioritized requirements achieved through the MoSCoW method to the GS and those achieved through the MCBRank to the GS. There is a poor agreement between the MoSCoW method and the GS, κ = .067, p> .0005. On the other hand, the kappa value shows a moderate strength agreement between the MCBRank method and the GS, κ = .600, p <.0005. Through the investigation, this research demonstrates that the set of prioritized requirements obtained using the MCBRank method moves closer towards the ideal Gold Standard compared to the prioritized requirements obtained using the MoSCoW method. Therefore, requirements prioritization in terms of the correct ranked requirements based on importance is better achieved through MCBRank.

A. Limitations

The research was conducted in a limited time frame, and therefore there are several limitations in the investigation conducted as stated below:

- It is ideal to have candidate requirements proposed by the participants. However, due to limited execution time and the need to focus on the prioritization exercise, the researchers carefully prepared the candidate requirements based on a specific case study. Ample time is given for the participants to understand the requirements before the prioritization activity.
- It is best to have a complete set requirement for at least an industrial small size system to investigate the requirements prioritization performance [28]. However, the candidate requirements were reduced to allow sufficient time to exercise prioritization activity. The researchers carefully selected the right combinations of requirements which consist of importance variety to allow the requirements prioritization activity to happen.

TABLE I. Strength of the Agreement based on Cohen kappa

Kappa Statistic	Strength of Agreement
<0.00	Poor
0.00-0.20	Slight
0.21-0.40	Fair
0.41-0.60	Moderate
0.61-0.80	Substantial
0.81-1.00	Almost Perfect

TABLE II. Symmetric Measures of GS and MosCoW Method

		Value	Asymptotic Standard Error[a]	Approximate T[b]	Approximate Significance
Measure of Agreement	**Kappa**	.067	.055	2.033	.042
N of Valid Cases		31			

a. Not assuming the null hypothesis

b. Using the asymptotic standard error assuming the null hypothesis

TABLE III. SYMMETRIC MEASURES OF GS AND MCBRANK METHOD

		Value	Asymptotic Standard Error[a]	Approximate T[b]	Approximate Significance
Measure of Agreement	**Kappa**	.600	.090	18.298	.000
N of Valid Cases		31			

[a] Not assuming the null hypothesis

[b] Using the asymptotic standard error assuming the null hypothesis

VI. CONCLUSION AND FUTURE WORK

An enhanced method is commonly understood to provide better results, but it is merely an assumption if scientific data does not prove the claim. Therefore, this paper presents an empirical investigation to show that an enhanced MoSCoW method named MCBRank improves software requirements prioritization. The empirical results showed that the requirements could be better prioritized closer to the Gold Standard and represent cumulative importance values from multiple stakeholders of a software system.

Further investigation can be done on various requirements elements and attributes that influence the prioritization for future work. Besides, techniques to be embedded for automation are worth exploring to expedite the prioritization process to improve performance. However, careful measures to include a variety of stakeholders' perspectives must be taken care of.

ACKNOWLEDGMENT

This paper is sponsored by Universiti Teknikal Malaysia Melaka through a research grant numbered PJP/2020/FTMK/PP/S01774.

REFERENCES

[1] Abd Elazim, K., Moawad, R. and Elfakharany, E., "A framework for requirements prioritization process in agile software development," In *Journal of Physics: Conference Series* vol. 1454, no. 1, pp. 1-11, 2020.

[2] Bukhsh, F.A., Bukhsh, Z.A. and Daneva, M., 2020. "A systematic literature review on requirement prioritization techniques and their empirical evaluation," *Computer Standards & Interfaces*, vol.69, pp.1-18, 2020.

[3] Asghar, A. R., Bhatti, S. N. Tabassum, A. Sultan, Z. and Abbas, R. "Role of requirements elicitation & prioritization to optimize quality in Scrum Agile Development," *International Journal of Advanced Computer Science and Applications (IJACSA)*. vol.7, no. 12, pp. 300-306, 2016.

[4] Ahmad, S. Asmai, S.A. "Measuring software requirements quality following negotiation through empirical study" *International Journal of Applied Engineering Research.* vol. 11 no. 6, pp. 4190-4196, 2016.

[5] Olaronke, I., Rhoda, I. and Ishaya, G. "An appraisal of software requirement prioritization techniques," *Asian Journal of Research in Computer Science,* vol. 1 no. 1, pp. 1–16, 2018.

[6] Gilb, T. and Maier, M. "Managing Priorities: A Key to Systematic Decision-Making." *In Proceedings of INCOSE International Symposium*. Wiley. 2005.

[7] Ma, Q. "The effectiveness of requirements prioritization techniques for a medium to a large number of requirements: A systematic literature review." *In Dissertation, Auckland University of Technology.* 2009.

[8] Berntsson Svensson, R. and Torkar, R., "Not All Requirements Prioritization Criteria Are Equal at All Times: A Quantitative Analysis," arXiv e-prints, pp.arXiv-2104, 2021.

[9] Jahan, M.S., Azam, F., Anwar, M.W, Amjad, A., and Ayub, K. "A novel approach for software requirement prioritization," *In Proceedings of 7th International Conference in Software Engineering Research and Innovation*. IEEE. pp. 1-7, 2019.

[10] Hujainah, F., Bakar, R.A., Alhroob, E., Al-haimi, B. and Nasser, A.B., "Interrelated Elements in Formulating an Efficient Requirements Prioritization Technique," *In 10th IEEE International Conference on Control System, Computing and Engineering (ICCSCE)* pp. 97-101, IEEE, 2020.

[11] Hujainah, F., Bakar, R.A., Nasser, A. B., Al-haimi, B. and Zamli, K. Z. "SRPTackle : A semi-automated requirements prioritisation technique for scalable requirements of software system projects," *Inf. Softw. Technol.*, vol. 131, 2020.

[12] Yaseen, M., Mustapha, A., Ibrahim, N., Rahman, A.U., Kamal, S.W. and Ijaz, A., "Importance of functional requirements prioritization: ODOO ERP as case study," *i-Manager's Journal on Software Engineering*, vol. 14, no. 1, 2020.

[13] Sasank, V.V.S., Prabha, B., Praveen, S. R. K., Hasane, S.K., Ahammad, G . Kumar, N.S. "Modern Driven Requirements Engineering with Requirements Prioritization in Software Testing" *Turkish Journal of Physiotherapy and Rehabilitation*. vol. 32, no. 2, pp. 3037–3043, 2021.

[14] Achimugu, P., Selamat, A., Ibrahim, R., Mahrin, M. N. "A systematic literature review of software requirements prioritization research," *Information and Software Technology*. vol. 56, no. 6, pp. 568–585, 2014.

[15] Miranda, E.,"Moscow Rules: A Quantitative Exposé," *International Conference on Agile Software Development*, pp. 19-34, 2022.

[16] Abdelazim, K., Moawad, R., and Elfakharany, E. "A Framework for Requirements Prioritization Process in Agile Software Development," *J. Phys. Conf. Ser.*, vol. 1454, no.1, 2020.

[17] Abdelazim, K., Moawad, R., and Elfakharany, E. "A Supporting Tool for Requirements Prioritization Process in Agile Software Development," *Futur. Comput. and Informatics J.*, vol. 5, no. 1, pp. 15–27, 2020.

[18] Tufail, H., Qasim, I., Masood, M. F., Tanvir, S. and Butt, W. H. "Towards the selection of Optimum Requirements Prioritization Technique: A Comparative Analysis," *5th Int. Conf. Inf. Manag. ICIM,* pp. 227–231, 2019.

[19] Saher, N., Baharom, F. and Romli, R., "A review of requirement prioritization techniques in Agile software development," *Knowl. Manag. Int. Conf.*, pp. 25–27, 2018.

[20] Jarzębowicz, A. and Sitko, N., "Agile requirements prioritization in practice: Results of an industrial survey," *Procedia Computer Science*, vol. 176, pp.3446-3455, 2020.

[21] Hudaib, A., Masadeh, R., Qasem, M.H. and Alzaqebah, A., "Requirements prioritization techniques comparison." *Modern Applied Science,* vol. 12, no. 2, p.62, 2018.

[22] Avesani, P., Ferrari, S. and Susi, A. "Case-based ranking for decision support systems," Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2689, pp. 35–49, 2003.

[23] Yaseen, M., Mustapha, A., Rahman, A.U., Khan, S. and Kamal, W. "Importance of requirements prioritization in parallel developing software projects," *International Journal of Scientific Research in Computer Science and Engineering*, vol. 9, no. 2, pp.171-179, 2020.

[24] Gerogiannis, V.C., Fitsilis, P., Kakarontzas, G. and Born, C., "Handling vagueness and subjectivity in requirements prioritization" *In Proceedings of the 22nd Pan-Hellenic Conference on Informatics,* pp. 150-155, 2018.

[25] Ahmad, S. "Measuring the effectiveness of negotiation in software requirements engineering" Thesis of The University of Western Australia, 2012.

[26] Cohen, J. "A coefficient of agreement for nominal scales," *Educational and Psychological Measurement*, vol. 20, no. 1, pp. 37-46, 1960.

[27] Landis, J. R. and Koch, G. G. Landis and Koch "Agreement of Categorical Data," *Biometrics,* vol. 33, no. 1, pp. 159–174. 1977.

[28] Gerogiannis, V.C., Tsoni, E., Born, C. and Iatrellis, O. "Software Features prioritization based on Stakeholders' Satisfaction/ Dissatisfaction and Hesitation" *In 46th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)* pp. 265-271, IEEE, 2020.