EAI Endorsed Transactions
on Internet of Things

Research Article EAI.EU

Assessing the Effectiveness of MoSCoW Prioritization in Software Development: A Holistic Analysis across Methodologies

Suchetha Vijayakumar [1], Krishna Prasad K. [2], Raviraja Holla M. [3,*]

[1]Srinivas University, Mangalore and Assistant Professor, St Aloysius (Deemed to be University), Mangalore, Karnataka, India.
[2]Srinivas University Institute of Engineering and Technology, Mangalore, Karnataka, India.
[3]Department of Information and Communication Technology, Manipal Institute of Technology, Manipal Academy of Higher Education (MAHE), Manipal, 576104, Karnataka, India.

Abstract

Effective software Requirement Prioritization plays a pivotal role in the success of the Software Development process, ultimately contributing to the successful delivery of high-quality products. Among the various methods for Requirement Prioritization, the MoSCoW method has gained widespread adoption due to its ease of use. However, its overall effectiveness remains a subject of inquiry. This paper presents a rigorous assessment of the MoSCoW Requirement Prioritization technique, drawing insights from software developers who engage in the Prioritization process. Our evaluation encompasses a distinct perspective: that of the developers tasked with Prioritization. The feedback solicited from developers encapsulates a diverse set of criteria, shedding light on the method's efficacy. Additionally, we perform sentiment analysis on the user experience of the Prioritization task to corroborate the method's accuracy and efficiency. Our study unfolds through a practical exercise involving the Prioritization of a predefined set of requirements using MoSCoW principles. A mixed method approach is employed for the purpose of assessing the effectiveness of MoSCoW. The findings of our quantitative research underscore the method's limitations, indicating that it may not be as effective and precise as previously believed. Furthermore, through qualitative analysis, we are able to highlight the complexities and challenges associated with MoSCoW-based Prioritization. The insights gained from this analysis prompt contemplation regarding the potential introduction of an evolved Requirement Prioritization method, while leveraging MoSCoW as a foundational framework. This research aims to inform the ongoing evolution of Requirement Prioritization methodologies, ultimately enhancing the efficiency and accuracy of Software Development processes.

Keywords MoSCoW, Mixed method approach, Sentiment Analysis, Requirement Prioritization, Software Development.

Received on 04 May 2024, accepted on 03 September 2024, published on 28 October 2024

doi: 10.4108/eetiot.6515

1. Introduction

Software Development is a dynamic process, which is equally competitive and challenging. The outcome of the entire process depends on how the developers understand the Requirements. Therefore, Requirements Engineering is important and crucial for a successful development process. Software Requirements Prioritization is an important step in Requirements Engineering which helps developers to understand and sort the requirements in the order in which they have to be taken for implementation. There are many Software Requirements Prioritization methods each having its own set of Pros and Cons. Often, the changing requirements during the Development process poses as a challenge for Prioritization process [1]. An Effective Prioritization technique [2] helps the Software Development teams to:

[*] Corresponding author. Email: raviraj.holla@manipal.edu

• Concentrate on providing the most beneficial features and functionalities in line with user requirements on a priority basis that matches the corporate objectives.
• Make the best use of one's resources by devoting time and energy to projects that have the biggest payoff and importance.
• Control project scope and reduce risks by determining and resolving important constraints and dependencies.
• Promote cooperation and communication between stakeholders by outlining precise standards for trade-offs and decision-making.

The MoSCoW (Must-have, Should-have, Could-have, and Won't-have) method of Prioritization is one of the Prioritization methods that has gained popularity because of its simplicity. This method of Prioritization is very easy to understand and adapt [3]. Though MoSCoW provides an effective decision making framework with respect to the order of Prioritizing requirements, its effectiveness needs to be evaluated against real world scenarios. In order to provide a thorough understanding of its impact on project outcomes and stakeholder satisfaction, we aim to evaluate the efficacy of MoSCoW Prioritization in Software Development through a mixed-methods approach [4] in this research study. We do this by combining quantitative metrics with qualitative insights. The main aim of this Research paper is to evaluate and assess the performance of MoSCoW Requirements Prioritization Technique by adapting Mixed methods approach

The Research paper is organised as follows. Section 2 presents an appropriate and thorough Literature Review. All findings are recorded and presented. Section 3 of the paper presents a brief overview of MoSCoW Prioritization technique, formulating the Research problem by identifying the limitations and gaps in MoSCoW Prioritization process and also set a Research hypothesis. Section 4 briefs about the Research Methodology used in the evaluation process. A brief description about the Data Set and various Data Analysis methods that are used is also mentioned in this section. This section also includes Data collection method and experimental setup to carry out the analysis. The whole experimental setup and way of execution is presented in detail here. We have presented a detailed discussion on the results obtained after the analysis along with the inferences in Section 5. The paper is well concluded by mentioning the limitations and future scope of the work in Section 6.

2. Related Work

Voola, P., & Babu, A. V. (2013) have conducted an experiment with the three RP techniques: Numerical Assignment (NA), Analytic Hierarchy Process (AHP) and Extensive Numerical Assignment (ENA). Here different scales like ordinal, ratio and interval are taken into consideration respectively. It is also proved ENA is superior to NA and AHP [5]. Ali Khan, J., et al (2016) have evaluated seven software requirements Prioritization methods such as ANP, binary search tree, AHP, hierarchy AHP, spanning tree matrix, priority group and bubble sort through a case study. From their experiment and analysis of results, they conclude that ANP is the best techniques [6]. Hatton, S., (2008) have examined and found that volume of Requirements decreases as the development process progresses. Therefore, it is concluded that different Prioritization methods are required at different stages of development depending on the volume of requirements to be prioritized [7].

Marthasari, G., et al (2018) have used MoSCoW approach for development of Batu State Attorney library application project and came out with the fact that this approach will not be able to avoid potential delay time because of the presence of too many 'must have' phrases [8]. Babar, M. I., et al (2015) present the development and implementation of an expert system called "PHandler" that addresses the challenges of software requirements Prioritization. Further, the authors evaluate and prove that the system offers a scalable and flexible approach, integrating artificial intelligence and expert system techniques to support efficient decision-making in the requirements engineering phase of Software Development projects [9].

Achimugu, P., et al (2016) introduces and describes the "ReproTizer" tool, which is designed to facilitate the software requirements Prioritization process and address the challenges otherwise faced in the process. The tool automates and streamlines the Prioritization process, making it easier for stakeholders to collaborate and make well-informed decisions. The authors also demonstrate the effectiveness and practicality of the tool through case study [10]. Jahan, M. S., et al (2019) discuss the various challenges faced in traditional Prioritization techniques and prove that there is a need for an alternate and better approach. The new technique introduced by them proves to be effective and also advantageous through an experimental evaluation [11]. Ahmad, K. S., et al (2017) have presented a novel and effective approach which is a combination of fuzzy logic and the traditional MoSCoW method for prioritizing software requirements. Through this, the authors are able to handle uncertainty and imprecision in Prioritization decisions, providing a more flexible and nuanced way to rank requirements based on their importance [12].

3. MoSCoW Prioritization: An Overview

Software developers frequently employ the MoSCoW Prioritization technique to rank requirements, features, or tasks according to their significance and urgency. "MoSCoW" is an abbreviation [13] that stands for:

- Must-have: The requirements that have necessary specifications that are vital to the project's success and must be fulfilled on schedule. The project would fail or have a major negative effect on stakeholders if these elements weren't put into practice.

- Should-have: Significant prerequisites that are appealing but not necessary for the project's quick success. These add to the product's overall value and ought to be included after all essential features have been met.
- Could-have: Extra criteria that improve the product but aren't necessary for its main features. These things are not given priority over must-have or should-have items, but they may be accomplished if time and resources allow.
- Won't-have (or Would-have): Needs that are specifically left out of the project's present scope. These are either deemed low priority for the current development cycle or postponed to later versions.

With the help of the MoSCoW technique, stakeholders may work together to prioritise requirements and decide on project scope and resource allocation in an organised manner. Teams may manage expectations and trade-offs while concentrating on delivering the most important and valuable features by classifying objects into must-have, should-have, could-have, and won't-have categories.

Although MoSCoW Prioritization is a popular technique for prioritising requirements in Software Development, there are still a lot of restrictions and gaps in our knowledge on its usefulness and actual implementation. The following are some of the identified areas of concern:

- Subjectivity and Interpretation: A significant portion of the MoSCoW Prioritization process depends on stakeholders' subjective assessments in order to classify requirements. But how these categories are interpreted might differ greatly amongst people, which makes it difficult to decide which priorities to prioritise and emphasises the need for more precise rules and improvement.
- Absence of Formal Criteria: Although MoSCoW offers a framework for Prioritization, it is devoid of explicit standards or directives for judging the significance and immediacy of requirements. Because of its ambiguity, stakeholders find it difficult to make well-informed judgements, which emphasises the need for formalised criteria and improvement to increase its efficacy.
- Insufficient Stakeholder Involvement: All project stakeholders must collaborate and actively participate in order to effectively prioritise projects. On the other hand, incomplete or biased Prioritization decisions may result from some stakeholders' lack of engagement. Enhanced and supplementary channels of assistance are required to guarantee thorough stakeholder participation.
- Limited Adaptability: The MoSCoW's strict classification of requirements into four categories may not be able to take into account the changing needs of stakeholders or the dynamic nature of some projects. It needs to be improved and given more support systems in order to be more flexible and appropriate for a variety of Software Development scenarios.
- Absence of Empirical Evidence: Although MoSCoW Prioritization is widely used, there is a lack of empirical studies assessing its efficacy in practical projects. Current research frequently uses case studies or anecdotal evidence, emphasising the importance of the method, but does not sufficiently provide empirical evidence.

Based on the above limitations and also the research gaps identified through literature review, the following research hypothesis is framed:

Although fundamental, the MoSCoW Prioritization approach may not by itself ensure that requirements are prioritised accurately, that project management procedures are effective, or that stakeholders are satisfied with Software Development projects. But it can make a big difference in these results when combined with other guidelines that can be taken as a framework for all requirements that have to be prioritised.

4. Methodology

A mixed-methods approach [14] will be used in this study to assess how well MoSCoW Prioritization works in Software Development projects. This strategy efficiently addresses the study objectives and offers to provide a clear result for the research problem by combining quantitative and qualitative methodologies. The mixed-methods approach, by itself, contributes for the integration of numerical data with detailed contextual and situational understanding thus enhancing the robustness of the findings.

The quantitative methodology involves the analysis of survey data with statistical analysis to identify trends, patterns, and correlations, providing a comprehensive overview of the effectiveness and impact of the Prioritization technique. On the other hand, the qualitative methodology will provide rich, narrative insights to complement the numerical findings on applications of MoSCoW Prioritization.

The adapted methodology seeks to provide a more comprehensive and nuanced knowledge of MoSCoW Prioritisation through the triangulation of data from both quantitative and qualitative sources. This methodology guarantees that the study takes into account both the quantifiable results and the subjective experiences. It is anticipated that the combination of these techniques would produce useful insights that can bring out best practices and influence upcoming Software Development Prioritisation process.

4.1. Data Set and Data Analysis Methods

The Data Set used here is a real time Data which consists of eleven Prioritization orders as understood by the respondents followed by Level of Ease, Completion Status, Total number of comparisons done, Average time taken for every decision (in minutes), Satisfaction level and Drawbacks [15].

i. **Quantitative Data Analysis:** In order to compile important metrics like Satisfaction level, Completion Status, and the amount of time needed for Prioritization, quantitative survey results were analysed using descriptive statistics [16]. The results of the same are presented and interpreted in the sections that follow.

ii. **Qualitative Data Analysis:** Qualitative data analysis [17] is done to compile themes and patterns found in open-ended responses about stakeholders' opinions of MoSCoW Prioritization. The gathered data is subjected to a variety of analyses, including sentiment, content and thematic analysis [18].

4.2. Data Collection

In this Research, Quantitative data is collected through structured surveys distributed to stakeholders involved in Software Development projects who have a knowledge about MoSCoW Prioritization. The survey consists of questions that is meant to collect information about stakeholders' perceptions of MoSCoW Prioritization effectiveness, satisfaction levels, completion of Prioritization process, and other relevant information. Open-ended survey questions and semi-structured interviews is used to gather qualitative information about stakeholders' experiences, opinions, and difficulties regarding MoSCoW Prioritization.

4.3. Experimental setup and execution

This study was conducted by circulating a set of Requirements among students of Post-Graduation Studies. The number of respondents are 172 as this type of response requires a thorough knowledge of Software Development and Prioritization methods. The experiment is somewhat similar to an experiment conducted by Javed Ali Khan et al [2] for evaluating other type of Requirement Prioritization techniques . For our experiment, a set of Requirements of an Online Food Ordering system was given for Prioritization which is as follows:

Req 1. The system must let a Customer who is logged into the Food ordering system to place an order
Req 2. The system should confirm that the Customer is a registered Customer
Req 3. The System can prompt the Customer for an order through the menu
Req 4. The Customer may specify whether the order is to be delivered or picked. By default it is delivered
Req 5. The system will display a menu of that particular day only with available food items.
Req 6. When the Customer is done with the orders, the System should display the orders along with individual food prices and also the total payable amount
Req 7. The System will ask for Confirmation which needs to be accepted by Customer
Req 8. The order given by the Customer could be accepted or rejected based on the number of orders being processed by the Food ordering system at the moment
Req 9. After the order confirmation, the system must ask to select payment confirmation
Req 10. When the payment is confirmed, the system should accept the order and process
Req 11. A confirmation mail/message may be sent depending on the availability of mail id or phone number

The following table explains the verbal meaning of each requirement.

Table 1. Verbal meaning of requirements

Requirement	Verbal meaning
1. The system must let a Customer who is logged into the Food ordering system to place an order	Necessity
2. The system should confirm that the Customer is a registered Customer	Advise
3. The System can prompt the Customer for an order through the menu	Possibility
4. The Customer might specify whether the order is to be delivered or picked. By default it is delivered	Low Possibility
5. The system will display a menu of that particular day only with available food items.	Certainty
6. When the Customer is done with the orders, the System should display the orders along with individual food prices and also the total payable amount	Advise
7. The System will ask for Confirmation which needs to be accepted by Customer	Certainty
8. The order given by the Customer could be accepted or rejected based on the number of orders being processed by the Food ordering system at the moment	Low possibility
9. After the order confirmation, the system must ask to select payment confirmation	Necessity
10. When the payment is confirmed, the system must accept the order and process	Necessity
11. A confirmation mail/message would be sent depending on the availability of mail id or phone number	Conditional certainty

The survey was conducted in two phases. Initially, the students (respondents) were given a presentation of MoSCoW method of Prioritization with few examples.

They were given a day’s time to study the concept and understand it. Then the requirements and the various factors that were supposed to be evaluated by them post prioritising the requirements were given. The whole exercise was done through Google Forms and the responses were recorded.

5. Results and Discussion

Python libraries were used to analyse the data obtained though the survey. The results of analysis done on various columns which will henceforth be called as Evaluation parameter are as follows:

Table 2. Descriptive statistics of various evaluation parameters

Evaluation parameter	**Count**	**Set of unique values**	**Top value**	**Frequency of the top value**
Satisfaction level	172	3	Moderately satisfied	145
Total number of comparisons done	172	27	11	45
Average time taken for every decision	172	27	3	30
Level of Ease	172	3	Confusing	126

i. Satisfaction Level: Satisfaction level is an evaluation parameter that defines the performance and also calls for improvements, if any. In our questionnaire, three satisfaction levels were given to respondents to choose from, namely highly satisfied, moderately satisfied and not at all satisfied. Analysis of satisfaction levels revealed that the majority of respondents (145 out of 172) reported being "Moderately satisfied", which happened to be the most frequent satisfaction level as shown in Table 2. Figure 1 gives a visual comparison of various satisfaction levels as put forth by the respondents.

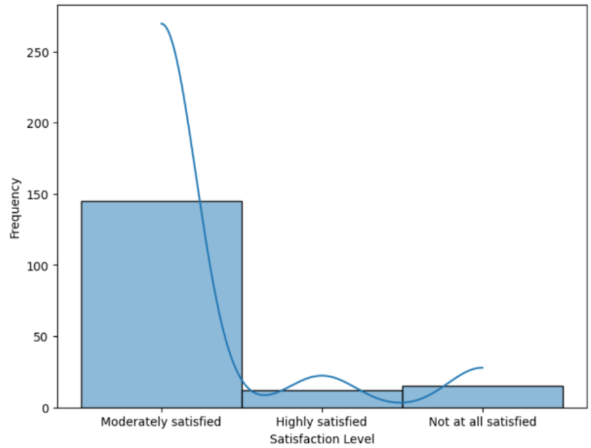


Figure 1. Visualisation of ‘Satisfaction level’

ii. Completion of Prioritization Process: This column in the dataset represents the status of completion of Prioritization Process. The completion rate of the Prioritization process was found to be 72.67%, indicating a substantial level of engagement among participants. This also supports the fact that MoSCoW can still be considered as the best option for prioritising requirements, though a small percentage of respondents were not able to complete the process. This is shown with a bar chart in Figure 2.

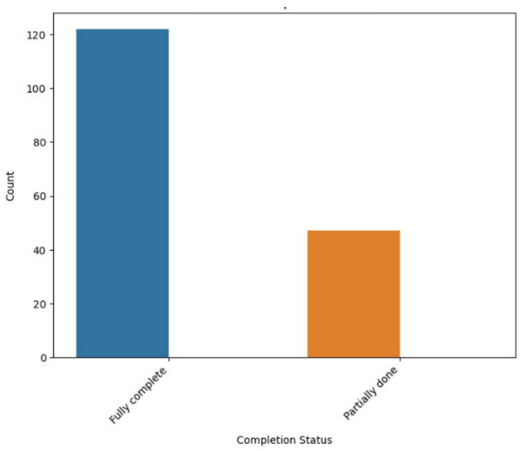


Figure 2. Visualisation of ‘Completion of Prioritization Process’

iii. **Total Number of Comparisons Done:** The number of comparisons done by the respondents shows the volatility of MoSCoW method of Prioritization. In our analysis (Table 2), we see that 11 comparisons are done by 45 respondents. Out of the total 11 requirements given, 45 people actually compared all the requirements, indicating the prioritization method is not volatile. Also Figure 3 shows a wide range of values and the high standard deviation suggests significant variability in the number of comparisons performed by respondents. The median value being lower than the mean indicates that the distribution of the data might be right-skewed, with some respondents performing a relatively large number of comparisons, leading to a higher mean. The minimum value of 0 suggests some respondents may not have performed any comparisons for various reasons including the confusion involved in the MoSCoW method. The quartile values provide insight into the distribution of responses, indicating that a significant portion of respondents performed relatively few comparisons, with a smaller proportion performing a larger number of comparisons. Overall, these descriptive statistics provide valuable insights into the distribution and characteristics of the total number of comparisons done by respondents in the Prioritization process.

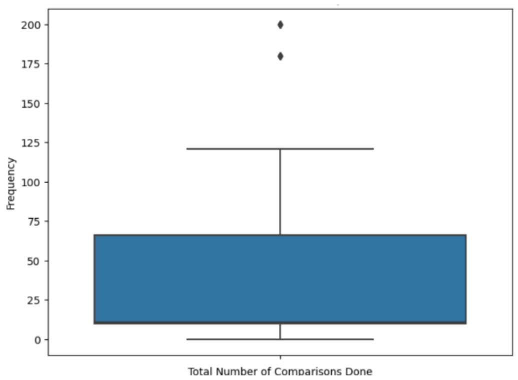


Figure 3. Visualisation of 'Total Number of Comparisons done

iv. Average Time Taken for Every Decision: The average time taken for making decisions during the Prioritization process ranged across 27 unique values. The most frequently reported time was 3 minutes, with 30 respondents indicating this duration (Table 2). The distribution of the average time taken for every decision in minutes was analyzed using a box plot also. The median time taken for each decision was approximately 10 minutes (Figure 4). The interquartile range (IQR), ranged from approximately 7 to 15 minutes, encompassing the middle 50% of the data. However, the distribution exhibited variability, with several outliers present beyond the whiskers of the box plot, indicating some decisions took significantly longer or shorter times compared to the majority of the dataset.

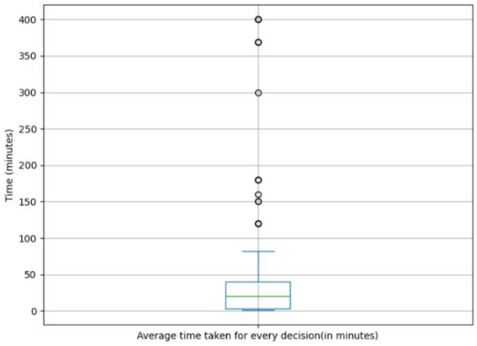
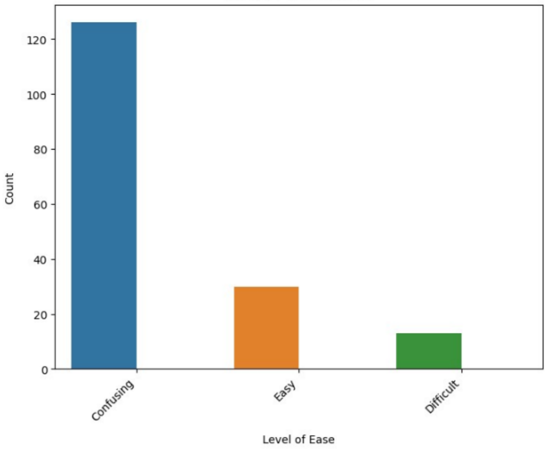


Figure 4. Visualisation for Average Time taken for every decision

v. Level of Ease: Respondents' perceptions regarding the ease of the Prioritization using MoSCoW principle varied. The majority (126 / 172) found the process "Confusing", followed by 33 respondents who found it "Easy", and 13 respondents who found it "Difficult". The Descriptive statistics is as shown in Table 2 and the corresponding graph in Figure 5 illustrated the fact that respondents found the MoSCoW Prioritization to be confusing.

Figure 5. Visualisation of 'Level of Ease'



vi. Drawbacks: In this study, respondents were provided with an open-ended question to express their views on the drawbacks of the MoSCoW Prioritization method. This open-ended format allowed participants to freely articulate their concerns, challenges, and criticisms without being constrained by predefined response options. The responses collected provide rich qualitative data that can offer deeper insights into the perceived weaknesses and limitations of the MoSCoW method from the stakeholders' perspectives.

To analyze the qualitative data collected from the "Drawbacks" question, a combination of thematic analysis, content analysis and sentiment analysis were employed. Each of these methods provides a different lens through which to examine the data, allowing for a comprehensive evaluation of the feedback

a) **Thematic Analysis**: Thematic analysis is one of the most popular qualitative data analysis techniques used to understand the common themes or patterns within the responses [19]. Responses were systematically reviewed to detect recurring topics and ideas. Themes such as "confusion", "difficult", "time-consuming process" and issues with "multiple must-have items" were identified. A small Python code was written that helped in categorizing the responses into predefined themes and counting the occurrences of each theme and the results are recorded and a bar chart drawn for the resulting values as shown in Figure 6. The result of thematic analysis is documented in Table 3 which reveals the following key issues:

i. Confusion: Mentioned by 83 respondents, indicating a significant issue with understanding the MoSCoW method.

ii. Difficulty: Reported by 9 respondents, highlighting challenges in using the method.

iii. Time-Consuming: Noted by 12 respondents as a major drawback.

iv. Multiple Must-Haves: Identified by 7 respondents, pointing to the impracticality of having too many must-have items.

v. Other: Various other concerns were mentioned by 61 respondents.

Table 3. Result of Thematic analysis performed on 'Drawbacks'

Total Responses	Number of responses showing				
	Confusion/Confusing	Difficult	Time consuming	Multiple 'must'	Other
172	83	9	12	7	61

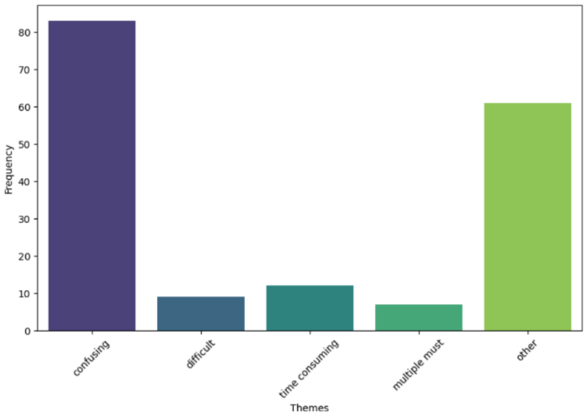


Figure 6. Visualisation of Thematic Analysis performed on 'Drawbacks'

b) Content Analysis: A research technique called content analysis can be used to find specific terms, themes, or ideas in a given set of qualitative data [20]. Content analysis was performed to quantify the presence of specific words, phrases, or concepts within the responses to determine the frequency of certain themes. This was done by using a tool like word cloud [21] generation to highlight the most commonly mentioned drawbacks. A word cloud thus generated visualized the most frequently mentioned terms in the open-ended responses about the drawbacks of the MoSCoW Prioritization method as shown in Figure 7 . In the word cloud, larger words represented terms that were mentioned more frequently by respondents.

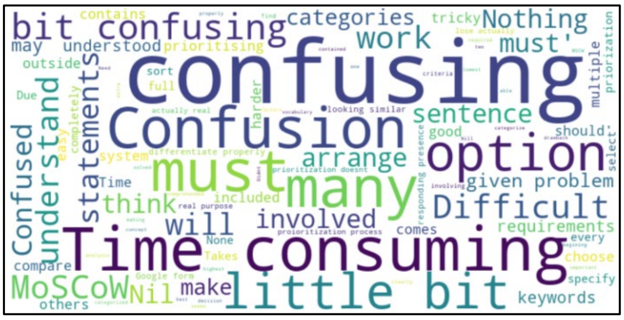


Figure 7. Word Cloud after performing Content Analysis of ' Drawbacks'

The word cloud visually summarized the main issues identified by users of the MoSCoW method. The prominence of terms such as "confusing", "confusion", "difficult," and "time-consuming" underscores significant challenges in the method's application, suggesting a need for additional training, clearer guidelines, and potentially supplementary tools to improve its effectiveness.

c) Sentiment Analysis: Sentiment Analysis is a handy NLP technique to determine the emotions and opinions expressed in a text having reviews or comments [22]. The result of this analysis could be either positive or negative based on the type of emotions it holds. This is sometimes referred to as Sentiment Orientation [23]. Sentiment analysis was performed on the Drawbacks expressed by the stakeholders while using MoSCoW method of Prioritization. In order to analyse the sentiments, two effective tools 'VADER'[24] and 'TEXTBLOB'[25] were used. The analysis proved that the sentiment carried in this column is mostly negative across both the tools used for analysis. Vader gave a result of 58.1% as negative sentiment and TextBlob gave a percentage of 41.9% as negative sentiment. The same is tabulated in Table 4.

Table 4. Result of Sentiment analysis performed on 'Drawbacks'

Tool	Positive	Negative	Neutral
VADER	4.7%	58.1%	37.2%
TEXTBLOB	24.4%	41.9%	33.7%

6. Conclusion and Future work

The comprehensive analysis of the MoSCoW Prioritization method reveals both its strengths and limitations. While the method provides a structured framework for categorizing requirements into must-have, should-have, could-have, and won't-have categories, the subjective nature of these classifications and the lack of formal criteria can lead to inconsistencies and challenges in practical application. Our findings indicate that, although MoSCoW can be effective in improving the accuracy of Requirement Prioritization and enhancing project management efficiency, it requires additional support and refinement to address its inherent ambiguities.

The sentiment analysis of stakeholder feedback on the 'Drawbacks' of MoSCoW, using both VADER and TextBlob, showed a predominant negative sentiment, highlighting the perceived challenges and dissatisfaction among users. Themes such as confusion, difficulty, and the time-consuming nature of the process were recurrent in the responses.

To enhance the effectiveness of MoSCoW, it is crucial to implement supplementary guidelines and criteria that provide clearer distinctions between Prioritization categories. Increased stakeholder involvement and iterative feedback mechanisms can also mitigate the subjective biases and improve the overall satisfaction levels. Future research should focus on empirical studies to validate these findings across different project contexts and explore the integration of MoSCoW with additional Prioritization techniques and rules to create a more robust and adaptable framework for Requirement Prioritization.

In conclusion, while MoSCoW is a valuable tool in the arena of Project Management techniques, its full potential can only be realized through continuous improvement and supplementary strategies so as to adapt to specific needs of Software Development projects.

References

[1] Licorish, S. A., Savarimuthu, B. T. R., & Keertipati, S. (2017, June). Attributes that predict which features to fix: Lessons for app store mining. In Proceedings of the 21st International Conference on Evaluation and Assessment in Software Engineering (pp. 108-117).

[2] Ali, A., Hafeezb , Y., Hussain, S., & Yang, S. (2020). Role of requirement prioritization technique to improve the quality of highly-configurable systems. IEEE Access, 8, 27549-27573.

[3] Kravchenko, T., Bogdanova, T., & Shevgunov, T. (2022, April). Ranking requirements using MoSCoW methodology in practice. In Computer Science On-line Conference (pp. 188-199). Cham: Springer International Publishing.

[4] Migiro, S. O., & Magangi, B. A. (2011). Mixed methods: A review of literature and the future of the new research paradigm. African journal of business management, 5(10), 3757-3764.

[5] P. &. B. A. V. Voola, "Comparison of requirements Prioritization techniques employing different scales of measurement.," ACM SIGSOFT Software Engineering Notes, vol. 38, no. 4 doi.org/10.1145/2492248.2492278, pp. 1-10, 2013.

[6] Ali Khan, J., Qasim, I., Khan, S. P., & Khan, Y. H. (2016). An Evaluation of Requirement Prioritization Techniques with ANP. International Journal of Advanced Computer Science and Applications, 7(7).

[7] Hatton, S. (2008, March). Choosing the right Prioritization method. In 19th Australian conference on software engineering (ASWEC 2008) (pp. 517-526). IEEE.

[8] Marthasari, G., Suharso, W., & Ardiansyah, F. A. (2018). Personal Extreme Programming with MoSCoW Prioritization for Developing Library Information System. Proceeding of the Electrical Engineering Computer Science and Informatics, 5(1), 537-541.

[9] Babar, M. I., Ghazali, M., Jawawi, D. N., Shamsuddin, S. M., & Ibrahim, N. (2015). PHandler: an expert system for a scalable software requirements Prioritization process. Knowledge-Based Systems, 84, 179-202.

[10] Achimugu, P., Selamat, A., & Ibrahim, R. (2016). ReproTizer: A fully implemented software requirements Prioritization tool. In Transactions on computational collective intelligence XXII (pp. 80-105). Springer Berlin Heidelberg.

[11] Jahan, M. S., Azam, F., Anwar, M. W., Amjad, A., & Ayub, K. (2019, October). A Novel Approach for Software Requirement Prioritization. In 2019 7th International Conference in Software Engineering Research and Innovation (CONISOFT) (pp. 1-7). IEEE.

[12] Ahmad, K. S., Ahmad, N., Tahir, H., & Khan, S. (2017, July). Fuzzy_MoSCoW: A fuzzy based MoSCoW method for the Prioritization of software requirements. In 2017 International Conference on Intelligent Computing, Instrumentation and Control Technologies

[13] Hudaib, A., Masadeh, R., Qasem, M. H., & Alzaqebah, A. (2018). Requirements prioritization techniques comparison. Modern Applied Science, 12(2), 62.

[14] Almalki, S. (2016). Integrating Quantitative and Qualitative Data in Mixed Methods Research--Challenges and Benefits. Journal of education and learning, 5(3), 288-296.

[15] Vestola, M. (2010). A comparison of nine basic techniques for requirements prioritization. Helsinki University of Technology, 1-8.

[16] Pentang, J. T., & Pentang, J. (2021). Quantitative data analysis. Holy Angel University Graduate School of Education: Research and academic writing. http://dx. doi. org/10.13140/RG, 2(23906.45764), 1.

[17] Lester, J. N., Cho, Y., & Lochmiller, C. R. (2020). Learning to do qualitative data analysis: A starting point. Human resource development review, 19(1), 94-106.

[18] Lochmiller, C. R. (2021). Conducting thematic analysis with qualitative data. The Qualitative Report, 26(6), 2029-2044.

[19] Vaismoradi, M., Turunen, H., & Bondas, T. (2013). Content analysis and thematic analysis: Implications for conducting a qualitative descriptive study. Nursing & health sciences, 15(3), 398-405.

[20] Bengtsson, M. (2016). How to plan and perform a qualitative study using content analysis. NursingPlus open, 2, 8-14.

[21] Heimerl, F., Lohmann, S., Lange, S., & Ertl, T. (2014, January). Word cloud explorer: Text analytics based on word clouds. In 2014 47th Hawaii international conference on system sciences (pp. 1833-1842). IEEE.

[22] Hussein, D. M. E. D. M. (2018). A survey on sentiment analysis challenges. Journal of King Saud University-Engineering Sciences, 30(4), 330-338.

[23] Taboada, M. (2016). Sentiment analysis: An overview from linguistics. Annual Review of Linguistics, 2, 325-347.

[24] Elbagir, S., & Yang, J. (2019, March). Twitter sentiment analysis using natural language toolkit and VADER sentiment. In Proceedings of the international multiconference of engineers and computer scientists (Vol. 122, No. 16). sn.

[25] Chaudhri, A. A., Saranya, S. S., & Dubey, S. (2021). Implementation paper on analyzing COVID-19 vaccines on twitter dataset using tweepy and text blob. Annals of the Romanian Society for Cell Biology, 8393-8396.