International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 10 Issue: 12 | Dec 2023 www.irjet.net p-ISSN: 2395-0072

Software Requirements Prioritization: What, Why, and How?

Narender Singh

Assistant Professor, Dept. of Computer Science, Govt. College, Ateli, Haryana, India.

Abstract - *Software requirements prioritization pushes the project towards success in any software development process. In contrast to general assumption, which claims that every software requirement is of equal importance, the software development teams actually restricted by time, money, and resource, which results in conflict between various stakeholders. To resolve this issue, we need a technique to prioritize software requirements, which implements the most essential requirements first, reduces the risk of project failure and ensures the success of software project.*

In this paper, an attempt is made to understand the terms software engineering, requirements engineering, software requirements prioritization, need of software requirements prioritization software requirements prioritization process, different aspects of software requirements prioritization, software requirements prioritization techniques, and challenges of software requirements prioritization.

***Key Words*: software engineering, requirements engineering, software requirements prioritization.**

1. INTRODUCTION

In recent decades, there is rapid progress in the field of science and technology with much advanced technologies, which leads to develop the complex software systems, hence includes high level methods and concepts. The gradually increasing complexity in present-day software systems, though, has surpassed the growth of software engineering concepts. Still, a lot number of projects have failed which is termed as “software crisis” [1].

The NATO Software Engineering Conference [2], which was held in 1968 has coined the term "Software Engineering" to address the software crisis [3]. Software professionals are able to develop high-quality software for computers by using a process, a number of tools, and a collection of approaches (practice) [4].

Only around 16% of software projects were successful, 53% had issues (cost or budget overruns, content inadequacies), and 31% were abandoned, according to the Standish Group in 1995 [5] [6]. Only 36% of all projects succeed when they are delivered on time, on budget, and with the required features and functions, 45% were challenged when they are late, over budget, and/or have fewer features and functions than required, and 19% failed when they are cancelled before completion or delivered but never used, according to the Standish Group's recently released report, "CHAOS Report 2015 [7]".

These statistics indicate that despite some progress have been made between 1995 and 2015; the current state of software development still falls far from the ideal. Issues such as budget overruns, missed deadlines, and incomplete functionality continue to plague the industry. Though, a lot of factors are there in the success and failure of a software project. But, the success heavily depends on three main factors: stakeholders’ participation, unambiguous software requirements specification, and executive management support. Similarly, inadequate user involvement and incomplete requirements are the two major factors which lead a software project to failure [5].

Further, no doubt, requirements are essential determinants of the success or failure of software engineering projects. Numerous software projects falter due to prevalent issues such as ambiguous or absent requirements, insufficient user engagement, and constantly evolving specifications [5, 8, 9]. Neglecting to address requirements comprehensively poses a significant risk to a project's overall success, regardless of the actions taken in subsequent phases.

In addition, the primary measure of a software system's success is the extent to which it achieves its primary objective. In general, identifying this objective is the process of requirements engineering for a software system. K. Wiegers defines it as “Requirements engineering is a discipline based on requirements development and requirements management. Requirements development is related to elicitation, analysis, specification and validation. The goals of these activities are gathering… software. Requirements management goal is to establish and maintain an agreement with the customer on the requirements for the software”. [10]

In this section, several terms are defined such as software crisis, software engineering, requirements engineering etc. The statistics of the two reports of Standish Group indicate that despite some improvements but still we need much more to achieve software systems’ success. If requirements are not handled properly at early stages of software development life cycle, they pose risk to project's overall success, no matter what the actions taken in later stages of it.

2. SOFTWARE REQUIREMENTS PRIORITIZATION

In any software project, requirements play the critical role that determines the success or failure of the project. If they are not addressed accurately, the project is likely to fail, regardless of what follows in subsequent phases of software development life cycle. Large scale software projects can have hundreds to thousands of software requirements, and the development teams do not have the sufficient budget, time, and resources to execute all of them. Furthermore, not all requirements are of equal importance. Hence, software requirements prioritization becomes necessary. It is the process of identifying and classifying requirements based on their importance, urgency, feasibility and their effect on the project's objectives to ensure that the most promising and critical requirements are addressed first. By using various tools and techniques, the development teams efficiently allocate the efforts and resources which results in minimizing the risk of project failure.

Khan et al. [11] depicts that the complexity of requirements engineering occurs due its ability to adapt and engage numerous stakeholders which includes customers (clients and end-users), system analysts, and system designers, developers, and project managers. The success and failure of any software project heavily depends on requirements and it needs to handle them precisely and effectively. However, due to the global nature of software development, stakeholders in the requirements engineering process face a variety of challenges, such as communication barrier, which can take place when software organizations are spread out across different geographic locations [11, 12].

In the era of globalized world, to ensure the development of high-quality software within time and budget, the requirements engineering is a critical process that must be carried out thoroughly and accurately to succeed. *"The hardest single part of building a software system is deciding what to build…. No other part of the work so cripples the resulting system if done wrong. No other part is more difficult to rectify later"* [13].

During requirement engineering process, it is not an easy task to prioritize software requirements. Almost all the customers want that all their requirements be considered first and development teams are constrained by budget, time, and resources. Hence, effective combination of analytical and interpersonal skills is a must for development teams to perform this. From the above discussion it is clear that not all requirements can be fulfilled [14], and hence, it is essential to prioritize the most critical requirement first.

3. NEED OF SOFTWARE REQUIREMENT PRIORITIZATION

Software requirements prioritization can be interpreted from different viewpoints. It is a significant task for decision-making [15]. It emphasis on handling the essential features first with efficient planning of software versions and timely delivery of software product with all essentials features [16]. Furthermore, it allows stakeholders to understand the importance of requirements and prioritize them accordingly [17]. This helps in future planning, reduces the possibility of project cancellation, and aids the estimation and ranking of funds, as well as determining the economic impact of implementing each requirement [18]. Due to limited resources such as budget, time, and schedule; software requirement prioritization becomes the most critical and challenging [19]. Prioritizing software requirements is an essential activity in any software project development that ensures that the high-quality software project is delivered that meets all user needs. There are a lot of reasons why software requirement prioritization is needed, which are as follows:

- Improving Resource Utilization: Setting requirements in order of their importance makes resource allocation more efficient and ensures that the most essential features are developed first. Ultimately, it leads to reduce time, money, and efforts in long term by focusing on the most essential features of the software.

- Decreasing Scope Creep: Scope creep in software project development is a common problem where the project's original limits are frequently being surpassed. It can be avoided by prioritizing software requirements.

- Increasing Quality: Software requirements prioritization ensures that the most essential requirements are handled first and more time and efforts are dedicated to them. It guarantees that these features are developed with the highest quality possible. As a result, the software product is of a higher quality and satisfies more user needs.

- Faster Time-to-Market: Software requirements prioritization allows the development teams to focus on the most essential features first, which speeds up the software delivery process. It is needed significantly in dynamic markets where sudden ideas and decisions are critical.

- Improving Stakeholder Satisfaction: Software requirements prioritization ensures that the software product will satisfy the essential needs

of its stakeholders, which improves acceptance rates and stakeholder satisfaction.

- Decreasing Maintenance Costs: Software requirements prioritization guarantees that the most essential features are handled first with maintainability in mind, which helps in decreasing maintenance costs.

- Better User Experience: Software requirements prioritization guarantees that the software is developed with users in mind, which improves user experience. It raises user happiness, acceptance rates, and the performance of the system.

- Minimizing Consequences of Changing Requirements: There are many factors like new business needs or competitive circumstances, which may lead requirements to change during software project development. Software requirements prioritization ensures that the most essential features are handled first, hence minimizing the consequences of changing requirements.

- Trade-offs: Establishing requirements based on order of priority supports in balancing requirements that are conflicting. It is easier to determine what requirements need to be implemented first and which of them may postpone or be canceled totally while requirements are actually prioritized.

In brief, prioritizing software requirements becomes crucial for improving resource utilization, decreasing scope creep, increasing quality, faster time-to-market, improving stakeholder satisfaction, decreasing maintenance costs, better user experience, handling risks, enhancing business value, dealing with resource constraints, controlling changing requirements, and establishing accurate trade-offs between requirements that conflict.

4. THE PROCESS OF PRIORITIZING REQUIREMENTS

The process of prioritizing requirements provides support for the following activities [20, 21, 22, 23]: to decide what are the core software requirements of a software system for stakeholders; to arrange and choose the most suitable set of software requirements to be implemented in a series of releases; in order to negotiate the intended project's scope against constraints like schedule, budget, time, resources, and quality; to assess each requirement's cost along with its business benefit; to assess each requirement's impact on the software architecture, the product's successive development, and the related costs; to produce a system that will satisfy the customer(s) by selecting a limited number of the requirements; to predict customer satisfaction; to gain a technical edge and optimize the market opportunity; to minimize both the rework and project schedule slippage; to resolve conflicting requirements, it's necessary to prioritize and negotiate such requirements and to resolve disagreement (if any) among different stakeholders; and finally, to deliver a high quality software product at low cost by assigning relative degree of significance to each requirement [24].

5. ASPECTS FOR SOFTWARE REQUIREMENTS PRIORITIZATION TECHNIQUES

A critical process that ensures that the most essential requirements are handled first is to prioritize software requirements. Different techniques are used to prioritize requirements, and the choice of the most suitable technique is influenced by various factors [25]. These factors include the complexity of the project, the urgency of the requirements, the resources available, and the stakeholders' needs. The interpretation of the prioritized requirements varies among stakeholders, as they perceive the ordering of requirements according to their individual perspectives. For instance, some stakeholders may prioritize time over budget, while others may have different priorities [26] [27]. Several factors affect software requirement prioritization, including stability, dependencies, risk, penalty, benefit, time, sensitivity, and regulatory/policy compliance [28]. Among these factors, cost and benefit are two key considerations that are directly related to software requirement prioritization [28].

In addition to cost and benefit, technical and commercial aspects are also crucial in software requirement prioritization. In the commercial realm, aspects like sales, marketing strategies, competition, long-term business goals, customer retention, product simplicity, innovation, resource allocation, client-focused features, and availability are of paramount significance. These commercial considerations need to be carefully weighed during the software requirement prioritization process, as they directly impact the overall business success [26]. At technical frontage, software requirements prioritization must address factors such as risk management, complexity analysis, value assessment, cost estimation, effort estimation, speed of implementation, technical complexity, time limitation, conflicting requirements, dependency among requirements, changing requirements, and overall project understanding and importance [28]. These are the critical factors that must be considered to prioritize the software requirements.

In software project development, an approach to requirement prioritization has been introduced by [29], which emphasizes the significance of multiple perspectives and is built on three key perspectives: the customer perspective, the business perspective, and the technical perspective. Customer perspective in software requirement prioritization may differ from one individual to another. On the other hand, business perspective takes into account the specific priorities and objectives of the organization by product manager. There is a distinct criterion which is most suited defined by each organization to prioritize its requirements to meet strategic goals [30]. Also, the technical perspective to prioritize software requirements focuses on the aspects such as cost, time, and risk. It requires that the development team members collaborate to each other to optimize resources and lessen the potential challenges in the software development process.

6. SOFTWARE REQUIREMENTS PRIORITIZATION TECHNIQUES

Bukhsh, Faiza, et al. [31] categorized software requirements prioritization techniques as follows:

- Software Requirements Prioritization based on Analytical Hierarchy Process (AHP), which uses a pair-wise comparison matrix for determining the relative importance of each requirement based on multiple criteria.
- Software Requirements Prioritization based on B-Tree, which uses a decision tree to visualize and compare the importance of requirements based on specific criteria.
- Software Requirements Prioritization based on fuzzy logic which uses a linguistic scale for representing the importance or preference of each requirement, and later they are transformed into a numerical value for prioritization.
- Software Requirements Prioritization based on stakeholders' preferences, which collect and aggregate preferences from different stakeholders for determining the relative importance of each requirement.
- Software Requirements Prioritization based on numerical assignment, which assigns a numerical value to each requirement based on specific factor or criteria.
- Software Requirements Prioritization based on advanced data processing, which uses machine learning algorithms, neural networks, and other advanced data processing techniques to predict the relative importance of each requirement.
- Software Requirements Prioritization based on binary inputs, which collects binary inputs from stakeholders for determining the relative importance of each requirement.
- Software Requirements Prioritization based on cost-value approach, which evaluates the cost and value of each requirement for determining its relative importance.
- Software Requirements Prioritization based on mutation testing and dependability analysis, which analyze consequences of each requirement on the system's dependability and robustness.

Due to various challenges, several software requirements prioritization techniques have been developed, which vary in terms of measuring scales, focus areas, and complexity [32]. Whereas, a few techniques present high-level insights into the process of software requirements prioritization, while some others present more extensive prioritization methods. It is clear that different techniques consider certain aspects while disregarding others. Therefore, it is not convenient to directly compare techniques on different criteria, as each technique focuses on distinct priority aspects [33].

It is essential to ensure that the resources are optimally utilized and all key stakeholders are kept engaged during software project development. Further, the development teams must try to deliver high-quality software product that addresses all stakeholders' needs while considering organization goals. The importance of software requirements prioritization techniques lies in their ability to facilitate the allocation of resources, reduce development time and risks, and finally contribute to the successful delivery of software products.

7. CHALLENGES

Mostly, all techniques of software requirements prioritization available in the literature are best suited for small projects with few requirements and become useless as the number of requirements increases in large projects. So, these techniques face numerous challenges such as failing to provide a feasible solution when scaling up from small to large projects, too time-consuming, producing erroneous or inaccurate results, and lacking the ability to recall previous results [34].

Conventionally, the process of software requirements prioritization has been regarded as an essential part of requirements analysis [35]. However, Lehtola et al. [36] have put forward the idea that software requirements

prioritization should be integrated into later stages of the software development life cycle. But, it highlights a common challenge which aspects to consider and to what extent [36]. It emphasizes the need for a more holistic approach to prioritize software requirements in software development.

8. CONCLUSION

Software requirement prioritization plays a critical role in the success or failure of any software project. So, it is critical to ensure that the resources are optimally utilized and all key stakeholders are engaged and well-informed about the project's importance on the most essential features during software project development. Further, it relies on many factors which includes budget, schedule, risk, resources, reward, time, stability, dependency, sensitivity, and compliance with regulatory and policy requirements. It includes techniques that are most frequently used are such as Bubbles Sort, Kano Analysis, Ranking, MoScoW Method, Numerical Assignments (Grouping), Hundred Dollar Method, and Analytic Hierarchy Process. It is a dynamic process to effectively prioritize software requirement that evolves throughout the project to ensure that the development efforts are focused on the most critical aspects to achieve project success.

In this paper, the author has made an attempt to answer the questions related to software requirements prioritization such as what is software requirements prioritization; why it is needed; and the techniques of it. Further, different aspects and challenges to prioritize software requirements are also discussed. Finally, the author has concluded that the software requirement prioritization plays a critical role in the success or failure of any software project.

REFERENCES

[1] R. Pressman, Software Engineering: A Practitioner's Approach, 3rd edition, McGraw Hill, 1992.

[2] P. Naur and B. Randell, "Software Engineering: Report of the Working Conference on Software Engineering," Garmisch, Germany, October 1968, NATO Science Committee, 1969. [Online] Available: http://homepages.cs.ncl.ac.uk/brian.randell/NATO/nato1968.PDF.

[3] Singh, Narender and Nasib Singh Gill. "Towards an Integrated AORE Process Model for Handling Crosscutting Concerns." International Journal of Computer Applications 37 (2012): 18-24.

[4] R. Pressman, Software Engineering - A Practitioner's Approach, 6th ed., 007-123840-9, McGraw Hill, 2005.

[5] The Standish Group, The Chaos Report (1995), Technical Report, Standish Group International, 1995. [Online] Available: http://www.it-cortex.com/Stat_Failure_Rate.htm#The%20Chaos%20Report%20(1995).

[6] Singh, Narender and Nasib Singh Gill. "Aspect-Oriented Requirements Engineering for Advanced Separation of Concerns: A Review." (2011).

[7] The Standish Group, The Chaos Report (2015), Technical Report, Standish Group International, 2015. [Online] Available: https://www.standishgroup.com/sample_research_files/CHAOSReport2015-Final.pdf

[8] I. Sommerville, P. Sawyer, Requirements Engineering - A Good Practice Guide, 978-0471974444, John Wiley, 1997.

[9] S. Dusire, M. Flynn, N. Dardenne, "Requirements Engineering: Applying Theory to Reality," In Proceeding of 10th Requirements Engineering Conference (RE'02), Essen, Germany, pp. 300-302, IEEE Computer Society, 2002.

[10] K. Wiegers, Software Requirements, 2nd ed., 978-0735606319, Microsoft Press, 2003. [Online] Available: http://www.amazon.com/Software-Requirements-2-Karl-Wiegers/dp/0735618798

[11] H. Khan, A. Ahmad, C. Johansson and M. A. Alnuem, Requirements Understanding in Global Software Engineering: Industrial Surveys, International Conference on Computer and Software Modeling (ICCSM), Vol. 14, pp. 183-190, 2011.

[12] Khan, Hashim et al. "Requirements Understanding in Global Software Engineering: Industrial Surveys." (2011).

[13] Brooks, F. P. (1995) "The Mythical Man-Month" Essays on Software Engineering, Addison-Wesley Longman, Boston, MA, USA.

[14] Anand Krishnan, Associate Consultant, Maveric-Systems, Modern Analyst e-Journal - The Business Analysis & Systems Analysis Magazine, [Online] Available: https://www.modernanalyst.com/Resources/Articles/tabid/115/ID/3332/Techniques-to-Prioritize-Requirements.aspx

[15] Sommerville, I., & Sawyer, P. (1997). Requirements engineering: a good practice guide. John Wiley & Sons, Inc.

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

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

[18] 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.

[19] Hudaib, Amjad et al. "Requirements Prioritization Techniques Comparison", Mathematical Models and Methods in Applied Sciences 12 (2018): 62.

20] Karlsson, Joachim, Claes Wohlin, and Björn Regnell. "An evaluation of methods for prioritizing software requirements." Information and software technology 39.14-15 (1998): 939-947.

[21] Sommerville, Ian, and Pete Sawyer. Requirements engineering: a good practice guide. John Wiley & Sons, Inc., 1997.

[22] P. Berander, A. Andrews, Requirements prioritization, Engineering and managing software requirements, Springer, 2005, pp. 69–94.

[23] Yeh, Albert C. "Requirements engineering support technique (REQUEST): a market driven requirements management process." Proceedings of the second symposium on assessment of quality software development tools. IEEE Computer Society, 1992.

[24] Berander, P., Andrews, A. (2005). Requirements Prioritization. In: Aurum, A., Wohlin, C. (eds) Engineering and Managing Software Requirements. Springer, Berlin, Heidelberg. https://doi.org/10.1007/3-540-28244-0_4

[25] L Lehtola and M. Kauppinen, "Empirical evaluation of two requirements prioritization methods in product development projects," Software Process Improvement, pp. 161-170, 2004.

[26] F. Sher, D. N. Jawawi, R. Mohamad, and M. I. Babar, "Multi-aspects based requirements priortization technique for value-based software developments," in Emerging Technologies (ICET), 2014 International Conference on, 2014, pp. 1-6.

[27] A. Egyed, "A scenario-driven approach to trace dependency analysis," Software Engineering, IEEE Transactions on, vol. 29, pp. 116-132, 2003.

[28] Sher, Falak et al. "Requirements prioritization aspects quantification for value-based software developments." (2019).

[31] Bukhsh, Faiza Allah, Zaharah Allah Bukhsh, and Maya Daneva. "A systematic literature review on requirement prioritization techniques and their empirical evaluation." Computer Standards & Interfaces 69 (2020): 103389.

[32] Berander, P., Andrews, A. (2005). Requirements Prioritization. In: Aurum, A., Wohlin, C. (eds) Engineering and Managing Software Requirements. Springer, Berlin, Heidelberg. https://doi.org/10.1007/3-540-28244-0_4

[33] Berander, Patrik, Kashif Ahmed Khan, and Laura Lehtola. "Towards a research framework on requirements prioritization." SERPS 6 (2006): 18-19.

[34] M. I. Babar, M. Ramzan and S. A. K. Ghayyur, "Challenges and future trends in software requirements prioritization," International Conference on Computer Networks and Information Technology, Abbottabad, Pakistan, 2011, pp. 319-324, doi: 10.1109/ICCNIT.2011.6020888.

[35] Sommerville, I.: Software Engineering. 5 edn. Addison-Wesley, Wokingham, England (1996)

[36] Lehtola, Laura & Kauppinen, Marjo & Kujala, Sari. (2004). Requirements Prioritization Challenges in Practice. Proceedings of 5th International Conference on Product Focused Software Process Improvement. 497-508. 10.1007/978-3-540-24659-6_36