JOURNAL OF SOFTWARE ENGINEERING & INTELLIGENT SYSTEMS
ISSN 2518-8739
31 August 2020, Volume 5, Issue 2, JSEIS, CAOMEI Copyright © 2016-2020
www.jseis.org

# USING CUMULATIVE VOTING AND PRIORITY GROUPS TO PRIORITIZE FUNCTIONAL REQUIREMENTS: ODOO ERP AS CASE STUDY

**MUHAMMAD YASEEN [1], AIDA MUSTAPHA [1], ZAHID ALI [2], SHAMS UZ ZAMAN [3], SYED WAJID KAMAL [3]**

[1] *Faculty of Computer Science and Information Technology, Universiti Tun Hussein Onn Malaysia*
[2] *Engineering Research and IT Services Provider, Peshawar, Pakistan.*
[3] *Exceed IT Solution, Peshawar, Pakistan*
Email: yaseen_cse11@yahoo.com [2]

OPEN ACCESS

**Academic Editor:** Farrukh Zeeshan, COMSATS, Pakistan.
**Published:** 31 August, 2020
**Funding:** Authors have no funds.
**Article Type:** Research Article

**ABSTRACT**

Requirements prioritization is an important activity during requirements collection and management phase of software engineering. Prioritization of small size requirements is not an issue but its importance increases when requirements are large in size. Different prioritization techniques are suggested to prioritize different types of requirements but none of the suggested techniques are applied to prioritize software requirements based on dependencies in functional requirements (FRs) from the perspective of software developers. This study addresses this research gap by applying Cumulative Voting (CV) and Priority Groups based prioritization approach to prioritize FRs. Using spanning trees, all FRs are first inter-related and then CV is applied first to assign priority values to each requirement belonging to different spanning trees. Using priority groups, prioritized requirements belong to different spanning trees are assigned to priority groups such that all FRs inside each priority group have similar priority. To do so, FRs of ODOO ERP as case study are considered for prioritization. Prioritized FRs can be easily managed by parallel development team members without having dependencies in FRs of priority group.

**Keywords:** software requirements prioritization; functional requirements; cumulative voting; priority groups; spanning tree; requirements dependencies;

## 1. INTRODUCTION

Requirement prioritization (RP) is an important activity during requirements management and implementation phase in requirement engineering [1][2]. RP is about giving priority to requirements for better time planning during implementation [3][4]. Without requirements prioritization, development phase could be costly as it takes more efforts to complete all the requirements within the specified timeframe [5]. Existing techniques from the literature have shown that they are not scalable for large set of requirements especially in dealing with dependency issues between the functional requirements. There are three types of requirements; (1) business requirements which deals with benefits and cost issues of requirements along with time constraint, (2) functional requirements (FRs) which are necessary for software system to develop, and (3) non-functional requirements that are not directly demanded but are necessary for ensuring quality product such as security and performance issues. Along this line, research on requirements prioritization techniques depends on the type of requirement under study whether at business level [6][7], non-functional level [8] or at functional level [9]. Meanwhile, some of the techniques are able to cater for all types of requirements [10]. In addition to the types of requirements, requirements prioritization techniques also depend on the size of the requirements data. For example, AHP is suitable for a small set of functional requirements but often fail on large requirements due to high time consumption. Technique such as cumulative voting (CV) works well for medium size requirements and easy in use. A lot of work is done and dozens of prioritization techniques are presented but none of them are applied based on inter-dependencies in FRs. As a case study, FRs of ODOO ERP are considered for prioritization. In our previous research studies, FRs of ODOO ERP are considered [11][12].

## 2. LITERATURE REVIEW

Analytical Hierarchical Process (AHP) is an organized decision-making method that is intended to compute complex multi-criteria decision problems [17]. AHP is technique that is also applied efficiently in many other fields such as biology and social sciences for prioritization. In fact, AHP is the utmost frequently discussed prioritization technique within decision making in requirements engineering. AHP is led by comparing all possible

***Corresponding Author:*** Muhammad Yaseen
***Email Address:*** yaseen_cse11@yahoo.com

pairs of hierarchically categorized entities such as requirements as well as stakeholders for obtaining comparative priorities for all objects. For each pair to compare, the prioritizing person estimates the importance relationship between the objects on a nine-level scale, where 1 means equal importance and 9 is the maximum difference. If requirement i is assigned an importance value when compared to requirement j, then j has the reciprocal value when compared to I. This shows that each pair of requirements only needs to be compared once with total of n (n-1)/2 (where n is the number of requirements) comparisons. Cumulative voting (CV) is a method where stakeholders are given 100 dollars and they have to distribute on all possible requirements just like voting mechanism [10]. This technique is also known as 100 dollars method. The requirement with high votes will be given more priority. This method is very simple for assigning priority to requirements but disadvantage of using this technique is that votes distribution of large size requirements become difficult. Besides, it stakeholders have to assign score to requirements manually. Priority Groups or Numerical Assignment is technique suggested where instead of prioritizing individual requirements, requirements are assigned to different priority groups and groups are then assigned priority values. This technique is suitable for large size requirements [13]. Prioritization techniques such as Goal-based Prioritization [7] is suggested to prioritize user requirements based on goals and business requirements. In Value Based technique, requirements are prioritized on the basis of weights of core values of business [14]. Company stakeholders or managers use ordinal scale to score business values that are necessary for organization. For example for banking system, the value of security is very high as compare to other values [15]. Prioritizing requirements on the basis of its cost and how much it has advantages is necessary. Techniques such as benefit and cost prediction [16] is suggested to prioritize requirements on the basis of cost and befits of particular requirements for their business. In his research study, author [17] describe prioritization based on cost of the requirements and benefits it provide to customers. Case-Based Ranking (CBR) a machine learning approach is presented to reduce the efforts during prioritization which combines stakeholder preferences with requirement ordering approximations computed through machine learning approaches. The framework provides an iterative prioritization process that can handle single and multiple human decision makers (stakeholders) and different ordering criteria [18].

## 3. RESEARCH METHOD DESIGN

Design of research method consist of the following steps as shown in Figure 1. In first step, all FRs are inter-related using Directed Acyclic Graph (DAG) and then converted to possible number of spanning trees as discussed in section 3.1 below. In second step, all FRs belong to different spanning trees are prioritized using CV technique as discussed in section 3.1. In third step, using NA, all prioritized requirements are assigned to possible number of priority groups. In last step, results are analysed using number of dependencies in FRs inside each priority group. Reducing number of dependencies in FRs inside each priority group shows importance of prioritization. All steps of prioritization are explained below.

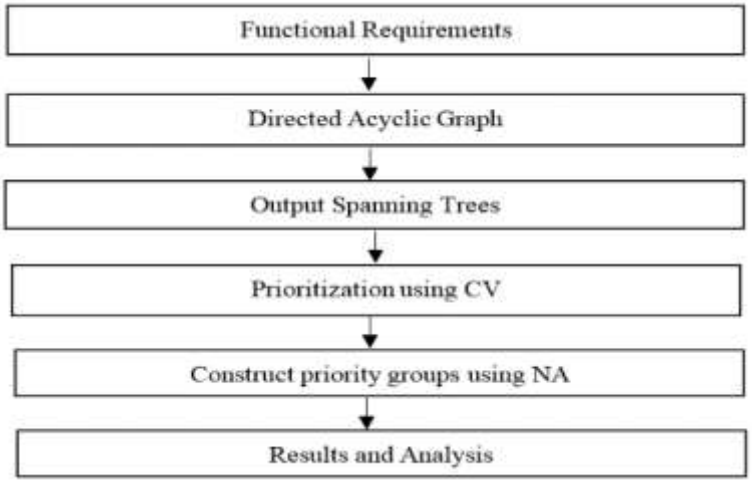


**Figure. 1** Step by step research design process

### 3.1 Output spanning trees

A graph G = (V;E) consists of a finite set of vertices V and a finite set of edges E. Graphs are useful for the representation of any kind of data in particular sequence. This research uses directed acyclic graphs (DAG) rather than cyclic graphs. A directed graph E is a set of ordered pairs of vertices (u; v). The arrows in the graph indicates the dependency of a requirement on another requirement. The requirement generates arrow and points to another requirement indicating that it is necessary or required for another requirement. For example, R1◄— R2 indicates

that R1 is depended on R2 or R2 is required for the completion of R1. Figure 2 shows the graph representation of requirements through directed acyclic graph.

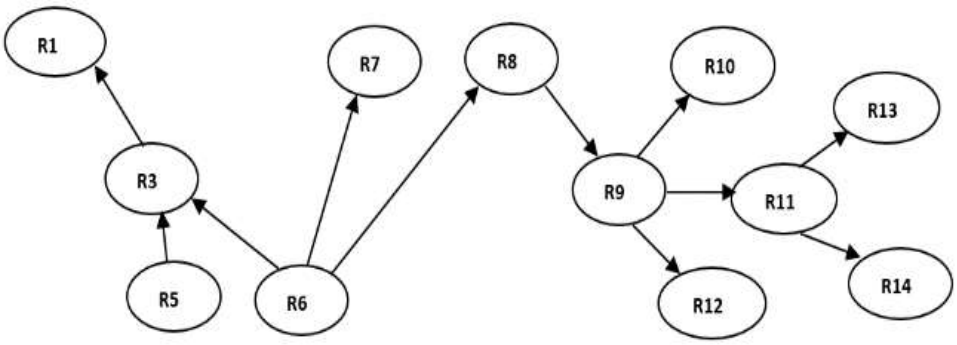


**Figure. 2** Directed graph inter-relating FRs

Resulted spanning trees from DAG of Figure 2 are shown in Figure 3 below. Spanning trees are produced from directed graph easily by following depth first search (DFS) method as discussed in [11].

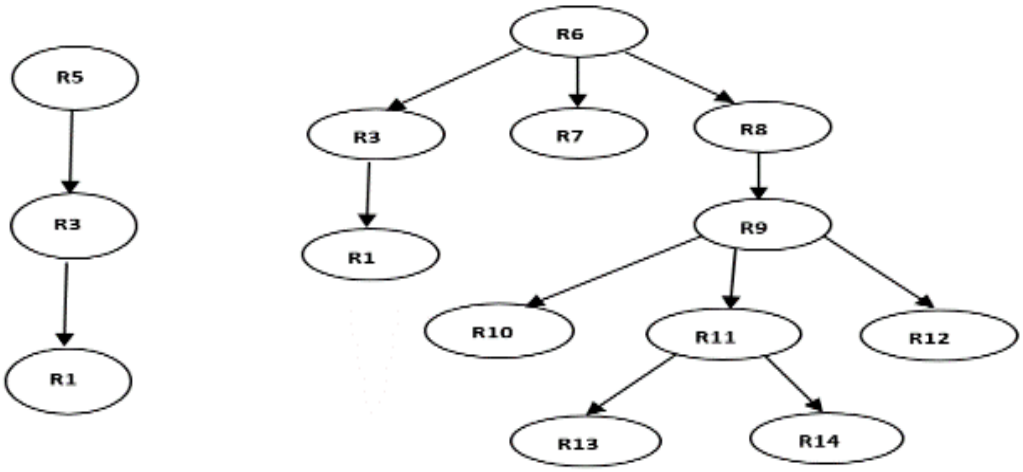


**Figure. 3** Resulted spanning trees from Figure 2

### 3.2 Prioritization using cumulative voting

Requirement that is required for other requirement will get high votes e.g. In Figure 3, R6 will get high priority votes as compare to R3, R7 and R8 while R8 will get high priority votes as compare to R9. Similarly, R9 will get high priority votes as compare to R10, R11 and R12. In this way, R11 will get high priority votes as compare to R13 and R14. As an example, priority votes assign to each requirement are given in Table 1 of both trees. Sum of priority values as result of prioritization of both trees will be equal to 100.

**Table. 1** Prioritized FRs of two spanning trees of Figure 3

| Requirement | Votes | Requirement | Votes |
|---|---|---|---|
| R6 | 20 | R11 | 4 |
| R3 | 10 | R12 | 4 |
| R7 | 10 | R13 | 2 |
| R8 | 10 | R14 | 2 |
| R1 | 8 | R5 | 10 |
| R9 | 8 | R3 | 5 |
| R10 | 4 | R1 | 3 |
| **Total Votes** | **100** | | |

### 3.3 Priority groups

Next, the prioritized list of FRs will be assigned to different priority groups. This approach assigns requirements into priority groups, whereby instead of prioritizing individual FRs, groups are prioritized such that high priority FRs belong to different trees with same priority will be assigned to high priority group and low priority FRs will be assigned into low priority group. Low priority group FRs cannot be implemented until high priority group FRs are not implemented. E.g. Figure 4 shows order of priority groups while requirements are

assigned in order of priority votes in descending order. When number of requirements is large, it is difficult to prioritize especially when software requirements are to be implemented by parallel development. If two depended requirements are assigned to parallel developers, this can cause problem as one requirement can only be implemented when its pre-requisite requirement is implemented first. Priority groups help in reduction of dependencies in requirements of parallel developers as inside priority group, all requirements are same or near to same in priority and if requirements are near to same in priority shows these requirements are not depended because depended requirements are assigned different votes during prioritization.

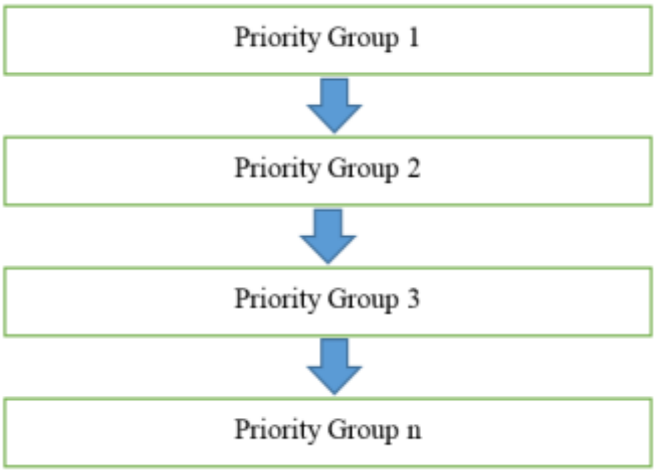


**Figure. 4** Requirements assigned to priority groups

## 4. CASE STUDY

Table 2 shows FRs of ODOO ERP. Column requirement shows all FRs belong to different modules of ODOO ERP while column required for shows all requirements for each particular requirement is required. Table 2 shows requirements belong to different spanning trees from Table. Requirements belong to each spanning tree are categorized into different levels and requirements of each level will be assigned same priority points.

**Table. 2** FRs belong to different modules of ODOO ERP

| Requirement | Required for | Requirements | Required for | Requirements | Required for |
|---|---|---|---|---|---|
| R1 (Employee) | R2, R4, R10, R11, R12, R17, R18, R20, R21, R22, R23, R25, R67, R81 | R33 (customer detail) | R24, R35, R36, R39, R55, R61, R64, R73, R90, R55 | R65 (supplier ledgers) | |
| R2 (Public information's of employee) | | R34 (products detail) | R35, R42, R60, R66, R70, R71, R91, R61 | R66 (stock ledgers) | |
| R3 (Employee personal info) | | R35 (sale) | R32, R51, R61, R62, | R67 (HR expense management) | |
| R4 (Contact info) | | R36 (customer refund) | | R68 (purchase return view) | |
| R5 (Job position) | | R37 (Sales persons) | R35, R36, R58, R63, | R69 (sale return view) | |
| R6 (Department) | R5, R81, R67 | R38 (customer receipts) | | R70 (Transfer In) | |
| R7 (Job information's) | | R39 (customer payment) | R38, R55 | R71 (Transfer out) | |
| R8 (Manager) | R5, R24, R67 | R40 (supplier receipts) | | R72 (order to suppliers) | |
| R9 (Coach) | | R41(supplier detail) | R42, R44, R52, R60, R65, R72 | R73 (order from customer) | |
| R10 (Contract information's) | | R42 (purchase) | R51, R59 | R74 () | |
| R11 (Contract reference information's) | | R43 (Sales man) | R42, R44 | R75 (Balance sheet) | |
| R12 (Salary generation) | R21, | R44 (supplier refund) | | R76 (compose message) | R79 |

| | | | | | |
|---|---|---|---|---|---|
| R13 (Salary rules) | | R45 (supplier payment) | R40, | R77 (message inbox) | R80 |
| R14 (Salary structure) | R12 | R46 (bank statement) | R47 | R78 (message Draft) | |
| R15 (Salary categories) | R12 | R47 (bank detail) | R49, R50, R53 | R79 (sent messages) | |
| R16 (Registers) | R12, R13, | R48 (cash registers) | | R80 (message Searching) | |
| R17 (Apply for leave) | R19, R20, | R49 (put money in) | | R81 (Job position in recruitment) | |
| R18 (Allocation request) | | R50 (put money out) | | R82 (Job) | |
| R19 (Approval) | | R51 (Profit and lost) | | R83 (appraisal form) | |
| R20 (Leave summary) | | R52 (supplier payment) | | R84 (create a job position) | |
| R21(HR payroll) | | R53 (Journals accounts) | R54 | R85 (Recruitment form) | |
| R22 (HR Expenses) | | R54 (Chart of accounts) | | R86 (Job selection process) | |
| R23 (HR expenses) | | R55 (Analytic accounts) | R54 | R87 (Link tracker) | |
| R24 (Project management) | R26, 27, R28, R29 | R56 (company) | | R88 (Mass mailing) | |
| R25 (Add team members) | | R57 (region) | R58 | R89 (contacts) | |
| R26 (Extra information's) | | R58 (Area) | | R90 (business pipeline) | |
| R27 (Project stages) | | R59 (purchase view) | | R91 (manufacturing orders) | |
| R28 (View current task) | | R60 (purchase return) | R68, | R92 (fleet management) | R93, |
| R29 (create a task) | R31, | R61 (sale return) | R69 | R93 (Vehicle repairing) | |
| R30 (Extra information's) | | R62 (sale view) | | R94 (Directories for documents) | R96 |
| R31 (Tasks stages) | | R63 (salesman ledgers) | | R95 (Documents history) | R96 |
| R32 (customer invoice) | R36 | R64 (customer ledgers) | | R96 (Documents attachments) | |

Table 3 shows FRs belongs to different spanning trees. Column Tree shows all notations for particular spanning trees. Requirements in each spanning tree belong to different level e.g. in spanning tree T3, R8 is required for R5, R67 and R24. So, root node R8 will be kept in Level 1 category and R24, R5 and R67 will be kept in Level 2 category. Similarly, R24 is required for R26, R27, R28 and R29, so these requirements will be kept in Level 3 category. R29 is required for R31, so it will go to Level 4. Purpose of keeping requirements in different levels is distribute votes equally among them as these requirements are not depended on each other's. Moreover, the vote's distribution process will become easier. While 20 requirements are not inter-related to any other requirement.

**Table. 3** FRs belong categorized to different levels in spanning trees

| **Tree** | **Level 1** | **Level 2** | **Level 3** | **Level 4** |
|---|---|---|---|---|
| T1 | R1 | R81, R25, R2, R23, R67, R4, R10, R11, R12, R17, R18, R22 | R19, R20, R21 | |
| T2 | R6 | R5, R67, R81 | | |
| T3 | R8 | R5, R67, R24 | R26, R27, R28, R29 | R31 |
| T4 | R34 | R42, R60, R66, R35, R70, R71, R90 | R51, R59, R68, R32, R61, R62, R80 | R36, R69 |
| T5 | R43 | R42, R44 | R51, R59 | |
| T6 | R41 | R44, R65, R72, R42, R52, R60 | R51, R59, R68 | |
| T7 | R37 | R58, R63, R35 | R32, R61, R62 | R36, R69 |
| T8 | R46 | R47 | R49, R50 | |
| T9 | R33 | R73, R35, R64, R39 | R62, R61, R32, R69, R38 | R36, R69 |
| T10 | R16 | R12, R13 | R21 | |
| T11 | R54 | R53, R55 | | |
| T12 | R57 | R58 | | |
| T13 | R14 | R21 | | |
| T14 | R15 | R21 | | |
| T15 | R76 | R79 | | |
| T16 | R45 | R40 | | |
| T17 | R91 | R92 | | |
| T18 | R93 | R94 | R95 | R96 |

## 5. RESULTS AND ANALYSIS

Results of prioritization as result of prioritization are shown in Table 4 below in descending order or priority. Any number of priority groups can be constructed using NA. Top priority requirements will be assigned to high priority group will low priority requirements will be assigned to low priority groups.

**Table. 4** Prioritized FRs of all spanning trees as result of CV

| Requirement | Priority | Requirement | Priority | Requirement | Priority |
|---|---|---|---|---|---|
| R33 | 3.3 | R44 | 1 | R38 | 0.8 |
| R1 | 2.88 | R48 | 1 | R40 | 0.8 |
| R34 | 2.55 | R3 | 1 | R25 | 0.8 |
| R8 | 2.25 | R52 | 1 | R7 | 0.8 |
| R41 | 2.5 | R82 | 1 | R17 | 0.8 |
| R43 | 2 | R83 | 1 | R18 | 0.8 |
| R37 | 1.8 | R84 | 1 | R22 | 0.8 |
| R46 | 1.6 | R85 | 1 | R23 | 0.8 |
| R6 | 1.6 | R86 | 1 | R2 | 0.8 |
| R93 | 1.6 | R87 | 1 | R53 | 0.75 |
| R16 | 1.6 | R88 | 1 | R55 | 0.75 |
| R66 | 1.36 | R89 | 1 | R26 | 0.72 |
| R70 | 1.36 | R56 | 1 | R27 | 0.72 |
| R71 | 1.36 | R60 | 1 | R28 | 0.72 |
| R90 | 1.36 | R65 | 1 | R29 | 0.72 |
| R63 | 1.35 | R72 | 1 | R61 | 0.68 |
| R54 | 1.5, | R74 | 1 | R62 | 0.68 |
| R35 | 1.3 | R75 | 1 | R32 | 0.68 |
| R45 | 1.2 | R77 | 1 | R80 | 0.68 |
| R47 | 1.2 | R78 | 1 | R31 | 0.63 |
| R57 | 1.2 | R13 | 1 | R49 | 0.6 |
| R76 | 1.2 | R58 | 0.8 | R50 | 0.6 |
| R91 | 1.2 | R92 | 0.8 | R51 | 0.5 |
| R94 | 1.2 | R79 | 0.8 | R59 | 0.5 |
| R14 | 1.2 | R67 | 0.8 | R68 | 0.5 |
| R15 | 1.2 | R81 | 0.8 | R69 | 0.45 |
| R99 | 1.2 | R4 | 0.8 | R36 | 0.45 |
| R39 | 1.1 | R5 | 0.8 | R96 | 0.4 |
| R64 | 1.1 | R95 | 0.8 | R19 | 0.64 |
| R73 | 1.1 | R9 | 0.8 | R20 | 0.64 |
| R24 | 1.08 | R10 | 0.8 | R21 | 0.4 |
| R30 | 1 | R11 | 0.8 | R100 | R99 |
| R42 | 1 | R12 | 0.8 | | |

Table 5 shows different scenarios consider while assigning prioritized FRs of Table 3 to two priority groups of different sizes. From Table 5 we can see that total dependencies are reduced from 80 to 41 only when prioritized requirements are assigned to two priority groups of 7 and 93 requirements each. These 7 requirements are top priority requirements which reduce number of dependencies for other requirements of low priority group of 93 requirements. We can see that when prioritized FRs are assigned to two priority groups with 25 and 75 requirements each, total dependencies reduced to 29 only. Dependency reduction column shows number of requirements reduced with prioritization using priority groups for different scenarios considered.

**Table. 5** Different scenarios consider while assign FRs to Priority Groups

| Scenario | Requirements in first group | Requirements in second group | Total dependencies | Dependency reduction |
|---|---|---|---|---|
| Scenario 1 | 7 | 93 | 41 | 39 |
| Scenario 2 | 10 | 90 | 35 | 45 |
| Scenario 3 | 15 | 85 | 36 | 44 |
| Scenario 4 | 20 | 80 | 33 | 47 |
| Scenario 5 | 25 | 75 | 29 | 51 |

## 6. CONCLUSION AND FUTURE WORK

In this research study, FRs of ODOO ERP as case are efficiently prioritized using CV and Priority Groups with less time complexity. For developers, prioritization of FRs is necessary as one requirement can only be implemented when pre-requisite FRs are implemented first. Also, as in case of parallel development, prioritization of FRs is necessary in order to reduce dependencies among FRs of parallel developers so that timely implementation of requirements can be assured. In this research, 100 FRs of ODOO belong to different modules are prioritized. Results are analysed for different scenarios of two priority groups which shows significant reduction of dependencies in FRs inside each group. In future, this FRs prioritization approach using CV and

priority groups will be evaluated using parallel development scheduling models such as RCPSP by calculating time estimation of each requirements using effort estimation model and whole project. Significant difference in total time estimation of software project with prioritized and un-prioritized FRs will shows advantage of prioritization with technique such as CV with less time complexity.

## REFERENCES


1. M. Aasem, M. Ramzan, A. Jaffar, and E. S. Islamabad, '*Analysis and optimization of software requirements prioritization techniques*', 2010.
2. M. Yaseen, A. Mustapha, and N. Ibrahim, '*An Approach for Managing Large-Sized Software Requirements During Prioritization*', *2018 IEEE Conf. Open Syst.*, pp. 98–103, 2019.
3. N. Misaghian and H. Motameni, '*An approach for requirements prioritization based on tensor decomposition*', *Requir. Eng.*, 2016.
4. M. Yaseen, N. Ibrahim, and A. Mustapha, '*Requirements Prioritization and using Iteration Model for Successful Implementation of Requirements*', *Int. J. Adv. Comput. Sci. Appl.*, vol. 10, no. 1, pp. 121–127, 2019.
5. M. Ramzan and M. A. Jaffar, '*Value Based Fuzzy Requirement Prioritization and its Evaluation Framework*', pp. 1464–1468, 2009.
6. N. Garg, M. Sadiq, and P. Agarwal, '*GOASREP : Goal Oriented Approach for Software Requirements Elicitation and Prioritization Using Analytic Hierarchy Process*', pp. 281–287, 2017.
7. M. A. A. Elsood and H. A. Hefny, '*A Goal-Based Technique for Requirements Prioritization*', 2014.
8. F. Dalpiaz, '*Contextual Requirements Prioritization and Its Application to Smart Homes*', vol. 1, pp. 94–109, 2017.
9. M. I. Babar, M. Ghazali, D. N. A. Jawawi, S. M. Shamsuddin, and N. Ibrahim, '*PHandler : An expert system for a scalable software requirements prioritization process*', *KNOWLEDGE-BASED Syst.*, 2015.
10. P. Chatzipetrou, L. Angelis, P. Roveg??rd, and C. Wohlin, '*Prioritization of issues and requirements by cumulative voting: A compositional data analysis framework*', *Proc. - 36th EUROMICRO Conf. Softw. Eng. Adv. Appl. SEAA 2010*, pp. 361–370, 2010.
11. M. Yaseen, A. Mustapha, and N. Ibrahim, '*Prioritization of Software Functional Requirements : Spanning Tree based Approach*', vol. 10, no. 7, pp. 489–497, 2019.
12. M. Yaseen, I. Journal, M. Yaseen, A. Mustapha, M. A. Salamat, and N. Ibrahim, 'International Journal of Advanced Trends in Computer Science and Engineering Available Online at http://www.warse.org/IJATCSE/static/pdf/file/ijatcse09912020.pdf. '*Prioritization of Software Functional Requirements : A Novel Approach using AHP and Spanning Tree*', vol. 9, no. 1, 2020.
13. J. Ali Khan, I. Ur Rehman, Y. Hayat Khan, I. Javed Khan, and S. Rashid, '*Comparison of Requirement Prioritization Techniques to Find Best Prioritization Technique*', *Int. J. Mod. Educ. Comput. Sci.*, vol. 7, no. 11, pp. 53–59, 2015.
14. N. Kukreja, S. Swaroop, B. Boehm, and S. Padmanabhuni, '*Value-Based Requirements Prioritization : Usage Experiences*', *Procedia Comput. Sci.*, vol. 16, pp. 806–813, 2013.
15. S. S. Payyavula and F. Si, '*Application of Value Based Requirement Prioritization in a Banking Product implementation*', pp. 157–161, 2012.
16. A. Herrmann and M. Daneva, '*Requirements Prioritization Based on Benefit and Cost Prediction : An Agenda for Future Research*', no. iv, pp. 125–134, 2008.
17. M. Daneva and A. Herrmann, '*Requirements Prioritization Based on Benefit and Cost Prediction : A Method Classification Framework*', pp. 240–247, 2008.
18. A. Perini, F. Ricca, and A. Susi, '*Tool-supported requirements prioritization : Comparing the AHP and CBRank methods*', *Inf. Softw. Technol.*, vol. 51, no. 6, pp. 1021–1032, 2009.


## AUTHORS PROFILE