ADAPTIVE FUZZY HIERARCHICAL CUMULATIVE VOTING: A NOVEL APPROACH TOWARD REQUIREMENT PRIORITIZATION

Bhagyashri Jawale[1], Ashish T. Bhole[2]

[1]Research Scholar, Department of Computer Engineering, SSBT's College of Engineering and Technology, Jalgaon, Maharashtra, India

[2]Associate professor, Department of Computer Engineering, SSBT's College of Engineering and Technology, Jalgaon, Maharashtra, India

Abstract

In Software Engineering (SE) process Requirement Engineering (RE) is considered as an important part in Software Development Life Cycle (SDLC). Requirement Prioritization is very useful for making good decisions about product plan but most of the times it is ignored. In many cases it is seem that the product fails to meet its core objectives because lack of proper prioritization. Increased emphasis on requirement prioritization and high changing requirements makes management of composite services time consuming and a complicated task. When a project has tight schedule, restricted resources and high customer expectations, it becomes necessary to deploy the most critical and important features as early as possible. The problem can be solved by prioritizing the requirements. In Software Engineering process numbers of requirement prioritization methods are already present. This paper shows the comparison of some of these techniques and based on its advantages and disadvantages a new technique 'Adaptive Fuzzy Hierarchical Cumulative Voting' is proposed. Fuzzy logic is used with adaptation mechanism, to target the situations where composite service behaviour can be deviated from customer expectations and to also deal with uncertainty and ambiguity. The proposed Adaptive Fuzzy Hierarchical Cumulative Voting includes the analysis of different self-adaptive properties such as self-heal, self-configure, self-optimize, self-protect and the addition to Fuzzy HCV, in order to increase the coverage of events that can occur at runtime. It may be useful to prioritize the requirements at run time.

Keywords: *Requirement prioritization, Fuzzy system, Fuzzy adaptation, HCV, Fuzzy HCV*

---***---

1. INTRODUCTION

Requirement Engineering (RE) is a crucial part of Software Development Life Cycle; and is also regarded as a vital division of software engineering. This phase includes an identification of requirements (often termed as Elicitation/Gathering), their analysis, documentation, validation; and management of requirements.

Requirement Engineering includes an important part termed as Requirement prioritization under Requirement Analysis phase. Its main objective should be to identify the requirement. When a project has inadequate resources, tough execution plan and too high customer expectations, its most significant aspects must be organized beforehand. For this purpose, prioritization of requirements becomes essential and many associates are engaged in this process [1].

A variety of personnel like Development representatives, Key customer representatives, project manager and other stake-holders are engaged in prioritization process. Both customers and developers do play a vital role in requirement prioritization. Business benefits from each function have to be balanced with its cost and implication in product design.

Adaptive mechanisms offer features like Self-protect, Self-configure, self-heal, self-optimize, etc. to the software systems, taking into consideration the factors like reasons for adaptation, reaction of system to modifications, objectives to be achieved by system and impact of adaptation over the system [2].

The proposed work presents the use of fuzzy logic with adaptive mechanism; to target the situations where complex project behavior can be deviated from customer expectations and to deal with uncertainty and ambiguity as well. The technique is named as Adaptive Fuzzy Hierarchical Cumulative Voting (Fuzzy HCV) which includes the analysis of different self-adaptive properties such as Self-protect, Self-configure, self-heal, self-optimize and the addition to Fuzzy HCV, in order to increase the coverage of events that can occur at runtime.

1.1 Requirements Prioritization

The software product quality is generally considered as the ability of it to suit and fulfill the customer needs. The prospect of a successful product or project is increased by finding, choosing and planning the appropriate releases having suitable functionality [3] [4]. When the requirements are developed in wrong or improper manner; and if customers defy the use of the product, it gains no credit how good other components of the development have been done. During the product development phase, decision-makers usually come across the challenge of exceeding number of candidate requirements than their competency to realize various allocated constraints (e.g. resources, time and cost)

[3]. Under such circumstances, it becomes essential to differentiate the significant requirements from relatively lesser significant ones, so as to minimize the business value by and large, by fulfilling various key requirements, vital stakeholders' preferences and technical constraints [5].

With identification of most vital, least precarious, least expensive, etc. requirements, it is feasible to find favorable combination of requirements which can be utilized to generate a system which executes only a subset of requirements, yet with customer satisfaction. To uncover the requirements which impart maximum value to business, some of the existing prioritization techniques can possibly be used.

The use of prioritization can be made in almost each situation, subjective to choices under consideration. For instance, in software engineering, its use has been in practice for prioritization of software process improvement related issues, stakeholders, software requirements, etc. During the software requirements prioritization in several distinct situations, it can be helpful in the form of support for decisions. For instance, it can be useful in selection and release planning, but not solely confined to [6].

1.2 Scales of Priority

Different techniques of prioritization are based on several different kinds of measurement scales. The arithmetic operations to be allowed are determined by the scale that has been utilized; and so is the type of analysis that can be performed [7]. Two most commonly utilized scales for prioritization of requirements are the ratio scale and the ordinal scale. Ratio scale is relatively more powerful. An ordinal scale maintains the order within elements and the numerical assigned to elements symbolize ranks, which indicates that no arithmetic operations like addition, subtraction, multiplication are allowed.

With ration scale, all arithmetic operations can be applied. Ordering as well as element ratios and interval sizes are relevant and a zero element can be there. Hence, it offers more information regarding the elements. It may be considered to have finer granularity in comparison with an ordinal scale [3]. With use of a prioritization technique which offers relative priorities on ratio scale, the calculation of total importance of a set of requirements can be done by adjoining their priorities as one. Along with it, the combination of different aspects and calculation of ratios within aspects can also be done. For instance, a cost-value ratio which exhibits the value that every requirement contributes in relation to its cost can be calculated. Such a way of finding most competent requirements to execute is not possible with the ordinal scale method. Another benefit of the ratio scale method is its ability to consider distances between requirements.

To exemplify, take into consideration three requirements prioritized as accounting for 75, 20 and 5 percent of the total importance. Now, considering the maximum importance of first requirement, implementation of only that particular requirement may be considered sufficient. While, conversely, as per ordinal scale prioritization, no enough information in support of such kind of decision would be there.

2. RELATED WORK

Requirement prioritization holds a momentous value in Software engineering. It has become a vital part in requirements engineering as it acts as a crucial and integral element in requirements negotiation and release planning in incremental software development. Prioritization is a multi-step process, mainly comprised of three stages-

Preparation stage: The stage of preparation is where the requirements are constituted as per the principle of technique of prioritization. In this, the team and its leader is chosen; and provided with all the essential information.

Execution stage: Secondly, the stage of execution, where actual prioritization of requirements is done by decision-makers with use of the information provided to them during the preceding stage. Prior to initiation of the execution stage, the team must agree.

Presentation stage: The lastly, the stage of presentation is the one in which the execution results are presented. Some prioritization techniques include different types of calculations which must be done prior to presentation of results [8].

Numerous techniques exist at the moment for requirement prioritization. Many of such techniques are quantitative, and they offer a methodical approach for Gathering of data and value assignment to different factors related to requirements for computing a priority.

Further techniques depend on carrying out groupings and informal generalizations prior to assigning the priorities, which is characteristically done to lessen the length of time required to compute priorities. Yet, this may forfeit a little consistency.

Rather some prioritization techniques are recognized as Analytical Hierarchy Process (AHP), Cumulative Voting (CV), Hierarchical Cumulative Voting (HCV), and Fuzzy Hierarchical Cumulative Voting (Fuzzy HCV), etc.

2.1 Analytical Hierarchy Process (AHP)

Analytical Hierarchy Process (AHP) was introduced for decision making problems in 1980 by Thomas Saaty. Thereafter, it has been adapted for the purpose of prioritization in SE (Software Engineering). AHP is ratio scale prioritization method. It yields better outcomes than rest of the approaches. It is a pair-wise comparison approach and all the possible requirement pairs are compared with each other in order to verify their priority and importance [9].

Analytical Hierarchy Process was initially developed with an objective of hierarchically prioritizing various objects considering various diverse aspects such as costs, benefit, etc. Yet AHP can possibly be used with hierarchically categorized objects like quality attributes as well.

However, in software domain, utilization of AHP as such is neither with consideration of hierarchical feature nor with several criteria or entities on different levels. Use of hierarchies for prioritization of objects on different levels in AHP implies the reduction in comparisons; and thereby the number of unnecessary comparisons decreases making the method extra sensitive to judgmental errors. During non-utilization of hierarchical feature, the flat variant is recognized as a pair-wise comparison technique in general. Nevertheless, it is noteworthy that pair-wise comparison is not an exclusive feature of AHP, but other techniques also do utilize it. During AHP utilization, the resulting priorities come under a ratio scale irrespective of the utilization of hierarchies in it.

Number of comparisons increases with number of requirements. It has been observed from studies that AHP is unsuitable for outnumbered requirements [10].

2.2 Cumulative Voting (CV)

Cumulative Voting (CV) was proposed in 2003 by Lefingwell and Widrig. It is synonymously known as a 100-point method or 100-Dollar test.

Ratio scale results can also be obtained with Cumulative Voting. Stake holders are allotted with 100 fictitious units like hours, money, etc. for assigning requirements. Every one of them is allotted 100 points of equal value. Distribution of points or numbers is made among requirements. Highest score requirement is considered as the most important one. Priority of any of them may also be ‘0’. It can be used easily. The crisis with CV is that a bias may occur with evaluations of stake holders. There is a chance that only a single requirement may be assigned all of the points by stake holders, in order to make that particular requirement the most important one. Cumulative voting is quicker and superior to AHP when utilized without hierarchical features of it. Drawback of CV technique is that the stake holders may be compelled for prioritization lacking accordance with their actual importance [11].

2.3 Hierarchical Cumulative Voting (HCV)

With the combination of AHP and CV, a novel technique called Hierarchical Cumulative Voting (HCV) has been launched by Patrik Berander and Per Jonsson in 2006. In HCV, benefits of both of these techniques are utilized.

The prioritization is carried out with distribution of points among requirements; like it is done in CV. Prioritization of requirements is carried out at different levels of hierarchy [12].

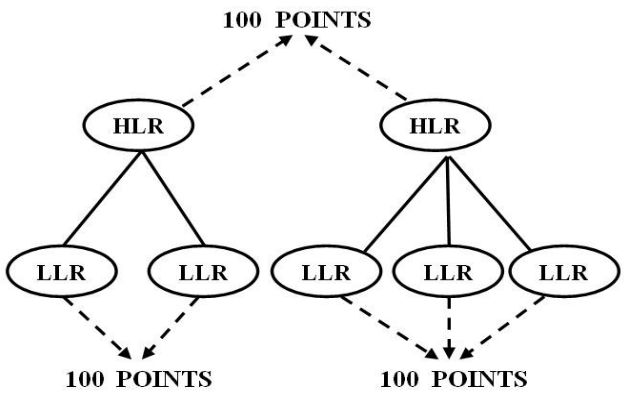

Fig -1: Hierarchical Cumulative Voting

Hierarchy-wise two different levels are created into which requirements are divided. The levels are known as High Level Requirements (HLR) and Low Level Requirements (LLR). Requirements are not prioritized all at once. Both HLR and LLR levels are prioritized by distribution of 100 points within them as shown in Fig -1. Multiple stake holders may be included in this method [13].

2.4 Fuzzy Systems for Requirement Prioritization

Stakeholder’s decisions about requirements to be prioritize may vary to a great extent. Importance of a requirement may vary from person to person. A crucial requirement for one person may be least important for another one. Requirements are seldom quantified accurately; generally they are in imprecise and vague manner. In order to cope up with such vagueness and impression fuzzy logic is used with requirement prioritization techniques [14].

Fuzzy logic aims to formalize appropriate reasoning which is based on multi-valued logic. Some of the characteristics of fuzzy logic are fuzzy sets, linguistic variables, fuzzy rules. Fuzzy set is a collection of objects characterized by a membership function. A linguistic variable uses words to represent its values instead of numbers. Fuzzy rule represents human knowledge in the fuzzy systems.

In fuzzification crisp inputs are converted to linguistic variables. The values of the variables are then calculated using fuzzy rules. At last, the defuzzification method takes these values to obtain crisp output [2].

2.5 Fuzzy Hierarchical Cumulative Voting (Fuzzy HCV)

A novel technique called as Fuzzy Hierarchical Cumulative Voting (Fuzzy HCV) is designed for the prioritization of requirements by combining Fuzzy system with HCV. HCV is the base of Fuzzy HCV. In order to get a single crisp value, fuzzification is done at the beginning of HCV and their defuzzification is done when the HCV process completes. In Fuzzy HCV, instead of crisp points/ numbers, Fuzzy Triangular numbers are utilized. For determining the intermediate priority, compensated calculation is utilized. Final priorities are calculated with use of normalization formula [1].

The comparison of characteristics of different requirement prioritization methods is given in Table -1.

Table -1: Comparison of Requirement Prioritization Techniques

Parameters	Name of Technique			
	AHP	CV	HCV	Fuzzy HCV
Author	T. L. Saaty	Lefing well and Widrig	Patrik Berander and Per Jonsson	Naila Sharif
Year	1980	2003	2006	2014
Result Scale	Ratio	Ratio	Ratio	Ratio
Speed	Slow	Fast	Fast	Fast
Concept	Pair-wise	Point Distrib ution	Point Distributio-n	Point Distributi-on
Best suited for no of requiremen ts	Small	Small, Mediu m	Large	Large
Hierarchy	Not Supp-orted	Not Supp-orted	Supported	Supported
Fuzziness	Not Supp-orted	Not Supp-orted	Not Supported	Supported

3. PROPOSED WORK

Nowadays, it has become a growing concern that requirements are of diverse importance. Besides this, theoretically or practically, there has been a modest progress on the mechanisms for software requirement prioritization till date.

Review of the state of practice in requirements engineering indicated that many organizations consider it crucial to prioritize requirements; and to fix their decisions according to rational or quantitative data. Yet, hardly any organization actually knew about assigning of priorities and communicating them to project members efficiently.

So, in order to make the prioritization of requirements more efficient and its use in many different organizations much simplified, we have attempted to present a new approach towards prioritization of requirements. It can deal with complexity, ambiguity, uncertainty and easily target the situations where complex service behavior can be deviated from customer expectations. This technique is a combination of fuzzy logic and adaptive mechanism; and can be recognized as 'Adaptive Fuzzy Hierarchical Cumulative Voting'. It may lessen the requisite efforts and may enable to generate high-quality outcomes which are considered reliable by its users. This new technique will certainly prove helpful for many different organizations in their process of prioritization of requirements.

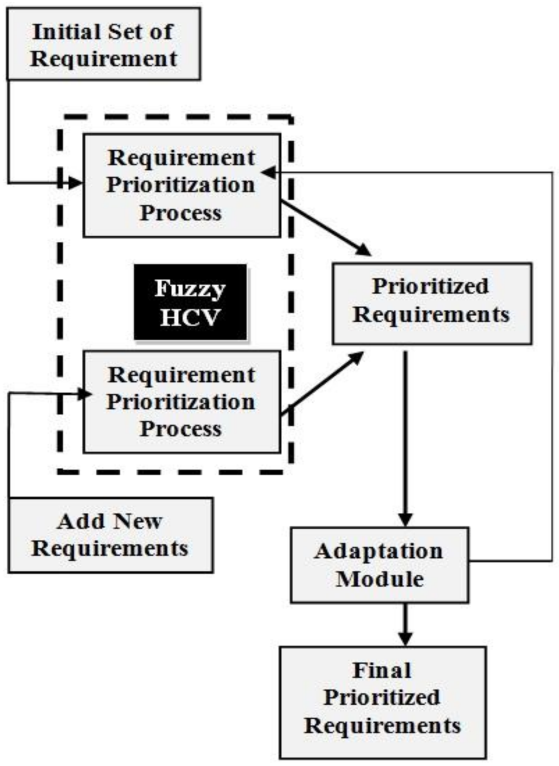
Initial Set of Requirement
Requirement Prioritization Process
Fuzzy HCV
Requirement Prioritization Process
Prioritized Requirements
Add New Requirements
Adaptation Module
Final Prioritized Requirements

Fig -2: Adaptive Requirement Prioritization Process

Fig -2 gives rough idea about how the proposed approach should work. Requirements from the customers or stakeholders are given as initial input and if the customer wants to add new requirements in between the execution of prioritization process, it can be added as new requirement. Requirement prioritization is done on set of requirements using Fuzzy Hierarchical Cumulative Voting technique and prioritized requirements are monitored and analysed by adaptation module and accordingly again Fuzzy HCV is applied on the requirements if they are incorrectly prioritized and final result of requirement prioritization will be gained.

A requirement is generally a singular documented physical and functional need that a particular product, design or process must be able to perform which is most commonly used in a formal sense in software engineering, systems engineering or enterprise engineering. In software engineering customers or stake-holders set the requirements and provide them as input to requirement prioritization technique (Fuzzy HCV).

In case of Adaptive Fuzzy HCV, firstly, items that are being prioritized are split into different levels of abstraction. Later, every item at the topmost level of abstraction is decomposed into more detailed level. Every level with greater detail is recognized as a separate abstraction level.

The item might be functions, characteristics, requirement levels or similar breakdowns, yet they should feature high levels of cohesion with their lower-level children and low levels of cohesion with their siblings

At the highest level of abstraction, voter allocates 100 points among all HLRs. Again voter allocates another 100 points to middle level requirements at the middle level of abstraction. The lower levels are representative of the decomposition of just one each of the mid-level items and that you would repeat the process with the next decomposition abstraction of level of all of the mid-level items.

Using Fuzzy HCV, first interval is selected in triangular fuzzy number for assigning priorities to all the requirements. Priorities are assigned in the form of triangular fuzzy number where primarily priorities are assigned to high level requirements and then to the low level requirements.

Intermediate priorities are calculated using compensated calculation or straight calculation using a formula.

Assuming Low Level Requirement (LLR_u) has parent High Level Requirement (HLR_v), the straight calculations are computed as shown in equation (1).

$$LLR_{u}(Pi) = LLR_{u}(Pa) \times HLR_{v}(Pa) \qquad (1)$$

The compensated calculations are computed as shown in equation (2).

$$LLR_{u}(Pi) = CHLR_{v} \times LLR_{u}(Pa) \times HLR_{v}(Pa) \qquad (2)$$

Where,
C = Compensation factor
Pi = Intermediate priority
Pa = Assigned priority
HLR = High level requirements
LLR = Low level requirements

Final priorities are calculated using process of normalization. Then Low Level Requirement (LLR_u) at LLR level for final normalized priority are calculated as shown in equation (3).

$$LLR_u(P_f) = \frac{LLR_u(Pi)}{\sum_k LLR_k(Pi)} \qquad (3)$$

Where,
P_f = Final Priority
Pi = Intermediate priority
LLR = Low level requirements

At last priorities are defuzzified by using defuzzification process and its result contains final priorities which are in the form of single crisp value [1]. Prioritized requirements block holds this result from Fuzzy HCV. In between the process, customer or stack-holders are able to add new requirements which are again prioritized using Fuzzy HCV in requirement prioritization process and get combined with previous result. Adaptation module monitors and analyses the result of prioritized requirements and according to analysis, it determines whether requirements are correct or not. If the requirements are not correctly prioritized then Fuzzy HCV is applied on that again.

4. CONCLUSION

Requirement Prioritization is a very important step towards making good decisions about product plan and it is used when it become necessary to deploy most critical and important features as early as possible. It may seem that existing requirement prioritization techniques do not deal with complexity, ambiguity, uncertainty and they are not able to easily target the situations where complex service behavior can be deviated from customer expectations.

The problem can be overcome and better output can be achieved by Adaptive Fuzzy Hierarchical Cumulative Voting technique which uses fuzzy logic with adaptive mechanism.

REFERENCES

[1]. Sharif N., K. Zafar, W. Zyad, and A. Afzal, 2014. Optimization of requirement prioritization using computational intelligence technique. Proceedings of the International Conference Robotics and Emerging Allied Technologies in Engineering (iCREATE), April 22-24, IEEE Xplore Press, Islamabad, pp:228-234. DOI:10.1109/iCREATE.2014.6828370
[2]. de Gyves Avila S. and K. Djemame, 2013. Fuzzy logic based QoS optimization mechanism for service composition. Proceedings of the 7th International Symposium on Service Oriented System Engineering (SOSE), March 25-28, IEEE Xplore Press, Redwood City, pp. 182-191.DOI:10.1109/SOSE.2013.28.
[3]. Berander P., and A. Andrews, 2005. Requirements prioritization. In: Engineering and Managing Software Requirements, A. Aurum and C. Wohlin (Eds.) Springer, Verlag, Berlin, pp: 69-94. ISBN: 978-3-540-25043-2, 978-3-540-28244-0.
[4]. Bhole, Ashish T., and Satpalsing D. Rajput, 2013 Ensuring Accountability for Application Sharing in the Cloud. Proceedings of 2013 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC), Dec. 26-28 , IEEE Xplore Press, Madurai, India, pp: 1-5. DOI: 10.1109/ICCIC.2013.6724146.
[5]. Ruhe G,, A. Eberlein, and D. Pfahl, 2002. Quantitative WinWin: A new method for decision support in requirements negotiation. Proceedings of the 14th International Conference on Software Engineering and Knowledge Engineering (SEKE'02), ACM New York, NY, USA, pp:159-166. DOI:10.1145/568760.568789
[6]. Ruhe G., and M. O. Saliu, 2005. The art and science of software release planning. Software IEEE, 22(6):47-53. DOI: 10.1109/MS.2005.164
[7]. Fenton NE., and S. L. Peeger, 1997. Software Metrics: A Rigorous and Practical Approach. 2nd Ed. PWS Publishing Co., Boston. ISBN: 9781439838228.
[8]. Joachim Karlsson, Claes Wohlin and Bjorn Regnell, 1998. Information and Software Technology. 39(14): 939-947. DOI: 10.1016/S0950-5849(97)00053-0
[9]. Saaty TL.,, 1980. The Analytic Hierarchy Process. McGraw-Hill, Inc.

[10]. Saaty RW. 1987. The analytic hierarchy process-what it is and how it is used. Mathematical Modelling, 9(3): 161-176. DOI:10.1016/0270-0255(87)90473-8
[11]. Lefingwell D. and D. Widrig, 2003. Managing software requirements: A use case approach. 2nd Ed. Addison-Wesley, 2003.ISBN:032112247X, 9780321122476
[12]. Berander P., and Mikael Svahnberg.2009. Evaluating two ways of calculating priorities in requirements hierarchies – An experiment on hierarchical cumulative voting. Journal of Systems and Software, 82(5):836-850. DOI:10.1016/j.jss.2008.11.841
[13]. Berander P., and P. Jonsson. 2006. Hierarchical cumulative voting (HCV)– Prioritization of requirements in hierarchies. International Journal of Software Engineering and Knowledge Engineering, 16(6): 819-850. DOI: 10.1142/S0218194006003026
[14]. Abdel Ejnioui, Carlos E. Otero and Luis D. Otero. 2012. A Simulation-based Fuzzy Multi-attribute Decision Making for Prioritizing Software Requirements. Proceedings of the 1st Annual conference on Research in information technolog, RIIT'12 , October, ACM New York, NY, USA, pp. DOI:10.1145/2380790.2380800.