IJCSMS International Journal of Computer Science & Management Studies, Vol. 12, Issue 03, Sept 2012
ISSN (Online): 2231 –5268
www.ijcsms.com

A Comparison among Various Techniques to Prioritize the Requirements

Ritu[1] and Dr. Nasib Singh Gill[2]

[1]Department of Computer Science and Applications, M. D. University, Rohtak-124001, Haryana, India
ritudhankhar08@gmail.com

[2]Professor, Department of Computer Science and Applications, M. D. University, Rohtak-124001, Haryana, India
nasibsgill@gmail.com

Abstract

In commercial software system development, software vendors often face the many difficulties to deal with large amount of requirements that enter the company every day. It is not possible to satisfy all the requirements in given constraint like time, cost, etc. hence there is a need to select the requirements that are more important. Requirements prioritization is a way that can play important role in the selection of requirements. Selecting the right order of requirements for product release depends on how successfully the requirements are prioritized. There are different requirement prioritization methods available with different characteristics. In this paper, we take a closer look of 9 different Requirement Prioritization techniques, namedas Fuzzy Analytical hierarchy process(FAHP), Classical Analytical hierarchy process(AHP), Hierarchy Analytical hierarchy process, minimal spanning tree, bubble sort, binary search tree, priority groups, planning game, and 100 point method. The Criteria used that are used for comparison are: required completion time, ease of use, reliability, and required number of comparison.

Keywords: *Software Techniques, Requirement Specifications.*

Introduction:

Software vendors have to fulfil the requirements that enter the company through customers, R&D, sales & marketing. An industrial project have hundreds of the requirements which is bounded by time, budget and resources[1]. The project manager should know requirements priority so that they can implement most important features of the project[2] with in time and budget[3]. In market driven development does not have easily identifiable customers and the requirements need to be invented based on the needs of several customers[4].draw out the required information from customers can be difficult to achieve, especially when multiple customers with diverse expectations are involved[5].

Therefore Customer involvement is a major contributing factor to company success[6]. Not all the requirements contain equal customer satisfaction. For example, financial managers look for the requirements with low cost, project managers look for the requirements which can be implemented fast and easily, market manager look for the requirements with high market value, and end users look for the requirements which are easy to use. Hence need to satisfy all requirements but we cannot include all the requirements reasons:- Available market opportunity, risks, product strategies, and costs need to be taken into consideration when planning. Now-a-days, projects are facing the problem of low success rate. According to an annual report named 'CHAOS' summary 2011 prepared by Standish group[7], only 37% of all projects were considered as successful which are delivered on time, on budget, with the required features and functionality. Among the rest, 41% were challenged which are late, over budget, and/or with less than the required features and functionality. 22% projects failed which are cancelled prior to completion or delivered and never used. Ten major factors causing failure in projects are:- 1. Lack of change management. 2. Poor communication 3.inadiquate resources, 4. Poorly defined requirements.,5. Inaccurate estimates, 6. Poor risk management 7.pooly defined deliverables, 8.over optimism, 9.no time for project management, 10.improved PM Skillset needed. Three major are poor communication, inadequate resources, poorly defined requirements. Requirement prioritization decreases the poor communication by user involvement by letting the stakeholders decide which requirements project should contain. This helps the stakeholder to understand the current constraints like inadequate resources and poorly

defined requirements. Ngo-The and Ruhe et al[9] thinks requirements prioritization has been recognized as one of the most important decision making processes in the software development process.

This paper provides an investigation of nine basic methods for prioritizing requirements: Fuzzy Analytical hierarchy process(FAHP), Classical Analytical hierarchy process(AHP), Hierarchy Analytical hierarchy process, minimal spanning tree, bubble sort, binary search tree, priority groups, planning game, and 100 point method. To study these methods, we systematically applied all methods to prioritize the 10-well defined requirements on a library management system for BPS Women University. We categorized the methods from a user's perspective according to a number of criteria such as required completion time, ease of use, reliability, and required number of comparison.

This paper is organized as follows. Section 2 motivates this work, and the paper continues in Section 3 by outlining the nine different prioritizing methods. Section 4 describes the evaluation framework and Section 5 presents the way to find out the best technique among the techniques under consideration Section 6 represents results and conclusion.

Motivation:

Industrial software development has found a growing acknowledgement that requirements are of varying importance. Yet there has been little progress to date, either theoretical or practical, on the mechanisms for prioritizing software requirements [9]. In a review of the state of the practice in requirements engineering, Lubars et al. [10] found that many organizations believe that it is important to assign priorities to requirements and to make decisions about them according to rational, quantitative data. Still it appeared that no company really knew how to assign priorities or how to communicate these priorities effectively to project members [11].

A sound basis for prioritizing software requirements is the approach provided by the analytic hierarchy process, AHP [12] where decision makers compare the requirements pair-wise to determine which of the two is more important, and to what extent. In industrial projects, this approach has been experienced as being effective, accurate and also to yield informative and trustworthy results [13]. Probably even more important, after using the approach in several commercial projects, practitioners are found to be very attracted by the approach, and continue to use it in other projects [11].but AHP has only been used in few applications in the software industry because AHP has a fundamental drawback which impedes its industrial institutionalization. Since all unique pairs of requirements are to be compared, the required effort can be substantial. In small-scale development projects this growth rate may be acceptable, but in large-scale development projects the required effort is most likely to be overwhelming [11].

Since AHP may be problematic for large-scale projects, Karlsson et al [11] identified five complementary approaches to challenge AHP. since previous studies indicate that all of these methods involve pair-wise comparisons, making relative judgments tends to be faster and still produces more reliable results than making absolute judgments [13]. Again, Paetsch et al [14] claims that agile software development has become popular during the last few years and in this field, one of the most popular methods is the extreme programming, which uses a prioritization technique called Planning Game (PG). In this paper, we investigated easy and accurate method and that is the FAHP method. Next section gives a brief description of each method, both in theory and then how it works practically. We focused on methods which may reduce the required effort, but still able to produce high-quality results, considered trustworthy by its users.

Prioritizing Methods:

Prioritizing methods helps decision makers to analyse requirements to assign numbers or symbols that reflect their importance. According to Karlsson et al[11], a prioritizing session may consist of three consecutive stages:

- The first stage is preparation stage where a person structures their requirements according to the principle of the prioritizing methods to be used. A team and a team leader for the session is selected and provided all necessary information.
- The intermediate stage is execution stage where the decision makers do the actual prioritizing of the requirements using the information they were provided within the first stage. The evaluation criteria must be agreed by the team before the execution stage is started.
- The last stage is presentation stage where the results of the execution are presented for those involved. Some prioritizing methods involve different kinds of calculations that must be carried out before the results can be presented.

This section describes the prioritization techniques investigated in this paper:

a). *Classical Analytic hierarchy process(AHP)*

The AHP was introduced by saaty [15] in 1980. The AHP is a systematic decision-making method that has been adapted for the software's requirement prioritization [15,16].it is conducted by comparing all

possible and unique pairs of requirements to determine which of the two is of higher priority, and to what extent. The total number of comparisons n*(n-1)/2(where n is number of requirements) are required to perform by the decision maker. In AHP the result is exponential increase in number of comparison as the number of requirements increases. Studies have shown that the AHP is not suitable for large numbers of requirements[17,18].AHP is very trustworthy, In original form, the redundancy of the pairwise comparisons allows a consistency check where judgement errors can be identified. More advantages is that the resulting priorities are relatives and based on a ratio scale, which allows for useful assessments of requirements. Saaty[15] states that the intensity of importance should be according to table 1.

Table 1:-Fundamental scale for pairwise comparisons

Intensity of importance	Description
1	Of equal importance
3	Moderate difference in importance
5.	Essential difference in importance
7	Major difference in importance
9	Extreme difference in importance
Reciprocals	If requirement i has one of the above numbers assigned to it when compared with requirement j, then j has the reciprocal value when compared with i.

b).*Hierarchy AHP*

In large-scale development projects the requirements are often structured in a hierarchy of interrelated requirements[19]. The most essential requirements are placed at the top of the hierarchy and the more specific requirements on levels below. Hierarchies are a common structure in daily use of AHP. But, to separate this hierarchical requirements structure from the flat requirements structure outlined previously.

Hierarchy AHP possesses similar characteristics to AHP. Using a hierarchical structure reduces the required number of comparisons, and also the amount of redundancy. Thus it is more sensitive to judgmental errors than AHP .

c). *Fuzzy AHP*

The limitations of AHP are that it is failed to deal with human vagueness. Hence in order to deal with vagueness of human thought, Zadeh first introduced the fuzzy set theory[20]. Fuzzy set theory generalizes classical sets in an attempt to model and simulate human linguistic reasoning in a domain characterized by incomplete, uncertain and vague data. The classical AHP cannot reflect the human thinking style. Avoiding these risks on performance introduces the studies fuzzy AHP to address the limitation. A fuzzy linguistic label can represent by a fuzzy number which is represented by fuzzy set. A TFN (triangular fuzzy number) is applied all fuzzy theories. A TFN is represented by 3 tuple (l, m,u). There is no standardizing scale in FAHP. The most popular scale which is used generally is given in table2. Fill the pairwise comparison matrix as AHP but using TFN shown in table 2.

Table 2:-Example of Fundamental scale for pairwise comparisons in FAHP

Intensity of importance	Description
(1,1,1)	Of equal importance
(5/2,3,7/2)	Moderate difference in importance
(9/2,5.11/2)	Essential difference in importance
(13/2,7,15/2)	Major difference in importance
(17/2,9,19/2)	Extreme difference in importance
Reciprocals (1/u,1/m,1/l)	If requirement i has one of the above numbers assigned to it when compared with requirement j, then j has the reciprocal value when compared with i.

The optimal solution is to find the normalized triangular fuzzy weights to obtain the local fuzzy weights.

d). *Minimal Spanning Tree*

The pairwise comparisons in AHP provide interesting relationships to each other. For example, if requirement A is determined to be of higher priority than requirement B, and requirement B is determined to be of higher priority than requirement C, then requirement A should be of higher priority when compared to requirement C. Despite this, AHP lets the decision maker perform the last comparison. Because of this redundancy AHP can indicate inconsistent judgments (such as claiming A to be of higher priority than C in this example).

The redundancy of the comparisons would be unnecessary if the decision makers being perfectly consistent. In such a case only (n–1) comparisons would be needed to calculate the relative intensity of the remaining comparisons. This implies that the least effort required by a decision maker is to create a minimal spanning tree in a directed graph (i.e. the graph is at least minimally connected).In the directed graph which can be constructed by the comparison provided, there is at least one path between the requirements not pair-wise compared [11].

The minimal spanning tree approach is supposed to be very fast as it dramatically reduces the number of pairwise comparisons. On the other hand, it is more sensitive to judgmental errors since all redundancy has been removed.

e). *Bubble Sort*

Bubble sort is one of the simplest and most basic methods for sorting elements with respect to a criterion [22]. It is also a candidate method for prioritizing software requirements, since the actual prioritizing process can be viewed as sorting requirements (i.e. the elements) according to their priorities (i.e. the criterion) [11].

Interestingly, bubble sort is closely related to AHP. As with AHP, the required number of pairwise comparisons in bubble sort is. But, the decision maker only has to determine which of the two requirements is of higher priority, not to what extent [11].

f). *Binary Search Tree*

A binary tree is a tree in which each node has at most two children. A special case of a binary tree is a binary search tree where the nodes are labelled with elements of a set[21]. Consider the elements of the set as the candidate requirements. This is of interest for prioritizing purposes since an important property of a binary search tree is that all requirements stored in the left sub tree of the node x are all of lower priority than the requirement stored at x, and all requirements stored in the right sub tree of x are of higher priority than the requirement stored in x. If the nodes in a binary search tree are traversed using in order traversing method, then the requirements are listed in sorted order. Consequently creating a binary search tree with requirements representing the elements of a set becomes a method for prioritizing software requirements [11].

Prioritizing n software requirements using the binary search tree approach involves constructing a binary search tree consisting of n nodes. The first thing to be done is to create a single node holding one requirement. Then the next requirement is compared to the top node in the binary search tree. If it is of lower priority than the node, it is compared to the node's left child, and so forth. If it is of higher priority than the node, it is compared to the node's right child, and so forth. Finally the requirements are inserted into the proper place and the process continues until all requirements have been inserted into the binary search tree [11].

Since the average path length from the root to a leaf in a BST is O(log n), inserting a requirement into a BST takes on the average O(logn) time. Consequently, inserting all n requirements into a BST takes on the average O(nlog n)time. In this case, too, the requirements are ranked on an ordinal scale.

g). *Priority Groups*

In some software development projects, one set of requirements can clearly be of a different kind of importance than another set. One way to reduce the required effort is therefore not to compare the requirements in these distinct sets. Thus another candidate method is to initiate the prioritizing process by dividing the requirements into separate groups based on a rough prioritization. Subsequently, the groups can be internally ranked either by using a suitable approach for ordering the requirement, for example, using AHP or to continue with another grouping of even finer granularity [11].

The primary gain is that, it is not necessary to compare high priority requirements with requirements of low priority, since they are placed in different groups. The actual choice of the number of groups depends on the situation as well as the knowledge of the people performing the prioritization. A simple strategy suggests using three distinct groups: low, medium and high priority. It may even be the case that the high priority requirements must be implemented, and hence there is no need to prioritize between them. In the same way the low-priority requirements may perhaps postponed to a later release [11].

h). *Planning Game (PG)*

In PG, numerical assignment and ranking is combined by first dividing the different requirements into priority groups and then ranking requirements with each other[24]. In extreme programming the requirements are written down by the customer on a story card which is then divided into three different piles. According to Beck [22], the piles should have the names; "those without which the system will not function", "those that are less essential but provide significant business value" and "those that would be nice to have". At the same time, the programmer estimates how long time each requirement would take to implement and then begin to sort the requirements into three different piles, i.e. sort by risk, with the names; "those that can be estimated precisely", "those that can be estimated reasonably well" and "those that cannot be estimated at all". Final result of this sorting is a sorted list of requirements on an ordinal scale[23].

Since PG takes one requirement and then decides which pile the requirement belongs to and each requirement is not being compared to any other requirement, the time to prioritize n requirements is n comparisons. This means that PG is very flexible and can scale up to rather high numbers of requirements, without taking too long time to prioritize them all[23].

i). *100 Points Method*

The 100 point is very straight forward technique where the stakeholders are given 100 imaginary units to distribute between the requirements. In this method each stakeholder gets hundred points of some value. With these points they should "purchase ideas". Each person writes down on a paper how much he/she thinks that one requirement is worth. When all the participants have written down their points, one person calculates, by taking the paper and summing up the points that each requirement has got, and presents the cumulative

voting results. The requirement that has got the highest score is the most important requirement [23].

Theoretically 100P is equally flexible as PG when it comes to the number of comparisons, i.e. n requirements takes n comparisons. Hence, it should be a really fast and scalable method, also in comparison to both AHP and BST. However, even though it has the same amount of comparisons as PG, i.e. n, it probably would take longer time to do the actual comparisons. The reason for this is that, while in PG the decision should be in which pile to put a requirement, i.e. ordinal scale, which is the same scale as BST, in BST the decision should be if requirement A is more or less important than requirement B. For 100P the scale is ratio, which is the same scale as for AHP. So the person that does the prioritization has to consider to which extent one requirement is more or less important than the other. At the same time he/she has only a small amount of points to distribute, which probably also takes some time to take into account in the distribution of points to the different requirements [23].

Evaluation:

The objective is to evaluate the prioritizing methods presented in the previous section. This section evaluates which has been carried out in the form of an experiment. We performed a single project with the aim of evaluating the nine prioritizing technique from the perspective of PMs and users. We took 10 basic requirements like **user-interface, functionality, size, cost, performance, OS-required, reliability, correctness, flexibility, maintainability.** This experiment was carried out on 7 different libraries' staff members i.e. on chief librarian, and other library members. The requirements was prioritize according to their need.

Techniques/ Parameters	AHP	Hierarchy AHP	FAHP	Minimum Spanning Tree	Bubble Sort	Binary Search Tree	Priority Group	Planning Game	100 Points
Consistency Index	Yes	Yes	Yes	No	No	No	No	No	No
Scale of Measurement	Ratio	Ratio	Ratio	Ratio	Ordinal	Ordinal	Ordinal	Ordinal	Ordinal
Year	1980	1995	2002	1983	1983	1997	2000	2004	2003
Author's Name	T.L. Saaty	Forenca	CK Kwong	Ullman, Hopcroft	A.V. Aho	Ryan, Karlsson	Beck	Beck	Leffingwell
Sophistication	Complex	Complex	Very Complex			Easy	Very Easy	Easy	Complex
Average Time Complexity	$O(N^2)$	$O(N^2)$	$O(N^2)$	O(Nlogn)	$O(N^2)$	O(NlogN)	O(NlogN)	O(N)	$O(N^2logN)$
Strengths	1.Rank choices in the order of their effectiveness 2. Easy to operate	1.Reduces the number of comparisons 2. Arrange the requirement in structure	1.Changes the crisp to TFN. 2.More reliable and accurate	1.Weight scale-free network 2.Greedy in nature	1.Compare adjacent requirements 2. Effective for less number of requirements	1.Reduces the number of comparisons 2. Easy to implement	1.Stakeholder have to choose one group. 2. Easy to operate	1.Very easy to operate	No mathematical calculations 2. give results in percentages.
Weaknesses	1.A fix rating scale.	1.Fix rating scale 2.increase in mathematical calculations	1.Increase in calculations 2. no any fix scale method to solve	1. Difficult to operate 2. Less accurate.	1. Weight of requirement should be known.	1. Weight of requirement should be known.	1.Recursive in nature	1.Give less accurate results	If requirements are high, then need to increase the points.

Evaluation Criteria:

Required completion Time:- This is average time required to complete all the stages of the method. Shown in Figure 1.
*Ease of use:-*this measure is used to describe how easy it is to use the prioritizing methods shown in figure 2.
*Reliability:-*this measure is used to describe how reliable the results are judged to be. Shown in figure 3.
*Required number of comparisons:-*for first five methods the number of decisions are pre-defined, but last four methods the number is based on the specific session was carried out. Shown in figure 4.

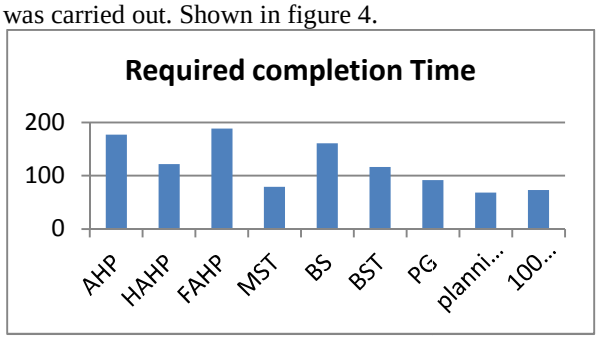

Figure 1 \

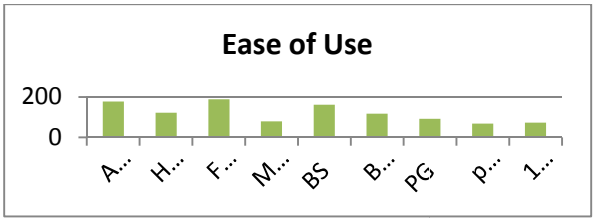

Figure 2

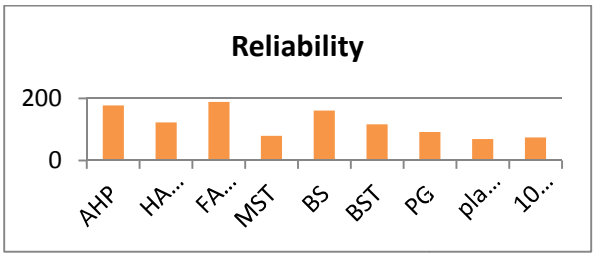

Figure 3

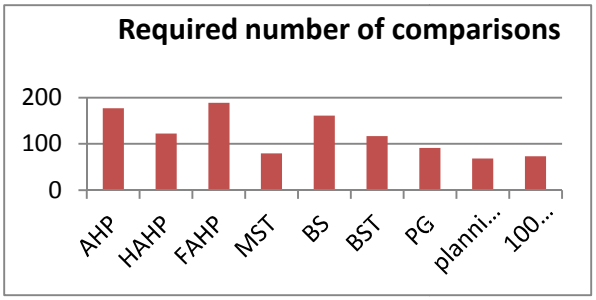

Figure 4

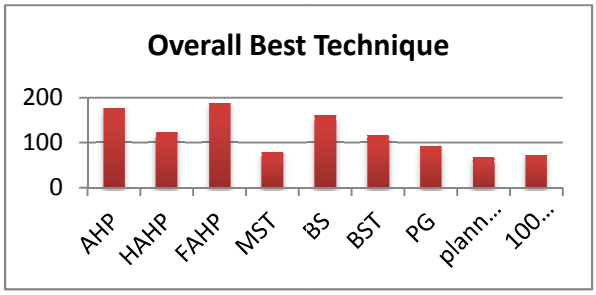

Figure 5

Calculating the Best Technique:-

After calculating data based on criteria, we assigned weight for each criteria to find the overall best technique. Each criteria was assigned weight according to table 3.
Table 3:-

Name of the Criteria	Weight
Required Completion Time	6
Ease of Use	9
Reliability	8.5
Required number of comparison	7.5

$\sum_{j=1}^{n}$ $SC_j=\sum_{i=1}^{nc}$ W_i*C_{ij} by using this formula the Score of criteria is calculated.
$OS_i=SC_i/n$
Where OS=overall score
nc= number of criteria
n= number of technique
SC_j= score of technique j
W_i=weight of criteria i
C_{ij}=score of technique j under criteria i.

Conclusion:-

Figure 5.is the combined graphical representation of the above comparison graph. From the figure 5 we can conclude the technique name Fuzzy Analytical Hieratical process has the highest points and this technique is best among all the other technique. This technique takes a range of the values and gives the more reliable results. This paper will be very useful in future when there will be need of the requirement prioritization. The user can easily see the comparison of the various technique on different criteria and their overall results.

References:-

[1] P.E. Drucker, The effective executive, HarperCollins, Australia, 1966.
[2] N.W. Kassel and B.A, Malloy, "An approach to automate requirements elicitation and specification", Inproc. Of the 7th IASTED international conference on SOFTWARE ENGINEERING AND APPLICATIONS, Marina Del Rey, CA USA, November 2003.
[3] Mohammad javeed Ali," Metrics for requirements Engineering", Master's thesis, Umea university, Sweden, june 2006.
[4] Hatton, S. (2007). Early prioritization of goals. In Advances in conceptual modeling- Foundations and application(pp. 235-244).
[5] Wiegers, K.: First Things First: Prioritizing Requirements. Software Development 7(9),(1999).
[6] Kabbedijk, j., Brinkkemper, S., Jansen, S.: Customer Involvement in Requirements Management: Lessons from Mass Market Software Development. In: 17th IEEE international Requirements Engineering Conference, pp.281-286. Atlanta(2009).
[7] www.staterlibrary.state.pa.us
[8] www.gtislig.org/Documents/10_Major_Causes_of_Project_Failure.pdf
[9] J.Siddiqi, M.C. Shekaran, Requirements engineering: the emerging wisdom, IEEE software 13(2) (1996) 15-19
[10] M.Lubars, C. Potts, C. Richter, A review of the state of the practice in requirementsmodeling, in: Proc. of the IEEE Int. Symp.on Requirements Eng. (1993) pp. 2–14.
[11] Karlsson, J., Wohlin, C., &Regnell, B. (1998). An evaluation of methods for prioritizing software requirements. Information and Software Technology, 39(14-15), 939-947.
[12] T.L. Saaty, The Analytic Hierarchy Process, McGraw-Hill, Inc. (1980).
[13] J. Karlsson, Software requirements prioritizing, in: Proc. of 2nd IEEE International Conference on Requirements Engineering (1996) pp. 110–116.
[14] Paetsch F., Eberlein A., Maurer F., "Requirements Engineering and Agile Software Development", Proceedings of the 12 IEEE International workshop, IEEE Computer society, 2003, pp 1-6.
[15] T.L. Saaty, The Analytic Hierarchy Process, McGraw-Hill, Inc. (1980).
[16] Regnell B, Host M, Nattoch Dag J, Beremark P, HjelmT(2001),"An industrial case study on distributed prioritization in market driven requirements engineering for packaged software. Requirements Engineering 6(1):51-62.
[17] Lehtola :, Kauppinen M(2004) Empirical Evaluation of Two Requirements Prioritization Methods in Product Development projects, Proceedings of the European Software Process Improvement Conference(EuroSPI 2004), Springer-Verlag, Berlin Heidelberg, pp. 161-170.
[18] Maiden NAM, NcubeC(1998) Acquiring COTS Software Selection Requirements. IEEE Software 15(2):46-56.
[19] A. Davis, Software Requirements: Objects, Functions and States. Prentice-Hall International, Englewood Cliffs, New Jersey, 1993.
[20] L.A. Zadeh. Fuzzy Sets information and control, volume 8: pp.338-353, 1965
[21] A.V. Aho, J.E. Hopcroft, J.D. Ullman, Data Structures and Algorithms. Addison-Wesley, Reading, MA, 1983.
[22] Beck K., "Extreme programming: explained", Addison-Wesley, USA, December 2001, 7th edition
[23] Mohammad ShabbirHasan," An Evaluation of Software Requirement Prioritization Techniques", (IJCSIS) Vol. 8, No. 9, December 2010
[24] Karlsson l, Berander P, Regnell B, WohlinC(2004) Requirements Prioritization: An experiment on exhaustive pairwise comparisons versus planning game partitioning. Proceedings of the 8th international conference on empirical assessment in software engineering(EASE 2004). IEE, Stevenage, pp. 145-154.