VALUE BASED INTELLIGENT REQUIREMENT PRIORITIZATION (VIRP): EXPERT DRIVEN FUZZY LOGIC BASED PRIORITIZATION TECHNIQUE

MUHAMMAD RAMZAN, M. ARFAN JAFFAR AND ARSHAD ALI SHAHID

Department of Computer Science
National University of Computer and Emerging Sciences (FAST-NU)
A. K. Brohi Road, H11/4 Islamabad, Pakistan
{ muhammad.ramzan; arfan.jaffar; arshad.a.shahid }@nu.edu.pk

Received July 2009; revised November 2009

ABSTRACT. Requirement Prioritization is a very critical but often neglected area of requirement engineering. Experience has shown that without proper prioritization of requirements presented by various stakeholders, the end product usually fails to meet its objectives optimally. In fact in many instances, the product is considered a failure because it fails to meet its core objectives. Several requirement prioritization techniques have been presented by various researchers over the past years. Working with these techniques has exposed several limitations when applied in software projects. In this paper, we have presented a novel multi-level value based intelligent requirement prioritization technique using fuzzy logic. We have introduced and applied the concept of requirement value to prioritize requirements. We have performed extensive experimentation using our proposed technique along with existing techniques. Results have shown that our technique has achieved superior prioritization results and consistency. The experiments have also shown that proposed technique is capable of delivering impressive prioritization under various circumstances.

Keywords: Requirement engineering, Requirements prioritization, Fuzzy systems, Intelligent requirements prioritization

1. Introduction. Software Engineering is one of the youngest engineering domains which emerged as recently as somewhere in the middle of 1980 as an accepted engineering discipline. The aim of SE is to create software products, services or their artifacts in order to meet the requirements posed by stakeholders while meeting quality constraints imposed on them. In order to meet both these objectives, any software development derives its purpose and meaning from the requirements posed by various stakeholders. In this context, better elicitation, modeling and analysis of requirements plays a very critical role towards development of a quality software. Requirement Engineering is an established domain of knowledge within software engineering which establishes practices and principles for effective requirement elicitation, modeling, specification, documentation, etc. One very important but often neglected practice of software requirement engineering is requirement prioritization. Requirement prioritization is the process of establishing worth and value of various requirements posed by multiple stakeholders based on certain established criteria of their utility for the ultimate software product. Several requirement prioritization techniques have been presented by authors. These techniques are both quantitative and qualitative in their nature. Some well known requirement prioritization techniques include Analytical Hierarchy Process (AHP), Cumulative Voting, Numerical Assignment, Ranking, Theory W, Requirement Triage and Wieger’s Method, etc. And there are several other techniques which we shall discuss in this paper.
Requirement prioritization enables us to understand the significance of requirements vis-à-vis the system to be developed and among requirements as well. With requirement prioritization, we can identify the focus areas which need most of our attention in order to develop a product which optimally meets the requirements of the stakeholders. In most of the situations, due to budget and time constraints, it becomes impossible to implement all the requirements posed by stakeholders. Also the nature of many projects is such that requirements are implemented in a staged environment. In both of these scenarios, we need requirement prioritization. We can prioritize requirement to realize which requirements can be delayed or altered so that other urgent requirements can be implemented and to what degree. We can also use requirement prioritization to determine which requirements to be implemented in earlier stages or later stages. We have been working with several funded projects during our research. These projects are faced with both of the above mentioned situations. We have found it very important to prioritize requirements in their true sense in order to develop a meaningful and successful product.

Requirement prioritization was a new practice in our specific development environment. So, our project needs required us to study further into various requirements prioritization techniques so that we can select one which can best suit our peculiar development environment. Our finding was that there is severe deficiency of any experimental results to determine which technique to prefer. Consequently, during this period of research and development, we studied various requirement prioritization techniques and tried to implement them on experimental level at various projects. We soon realized that all of these techniques worked well within certain situations but had some inherent problems attached with them which made it impossible to implement any one of these across the organization for all different kinds of projects. The main hindrances faced by us while implementing these techniques were related to cost, time and handling of evolving and creeping requirements. It was very important for us to work with a time and cost efficient prioritization technique. Since we were testing these techniques for small to medium software projects which needed a very efficient prioritization mechanism, our ultimate aim was to decide upon a technique which yielded better accuracy in requirement prioritization while at the same being efficient in terms of cost and time. We also were in search of a technique which could reduce the dependence on human factor due to certain obvious reasons which we shall elaborate in later sections. Unfortunately any of existing techniques could not satisfactorily answer all these problems. If some techniques like cumulative voting or ranking were efficient in terms of cost and time, their results were not very impressive. On the other hand, better techniques like AHP required very skilled human resource and a lot of capital to be applied in a proper environment.

In order to overcome these problems, one solution before us was to develop an artificially intelligent expert driven requirement prioritization technique. In one of our previous works, we had presented the initial sketch of a “value based requirements prioritization” technique [52]. This technique was very much similar to Theory W. In this approach, the end users and experts were asked to prioritize their requirements based upon the value that accomplishment of this requirement may have for the system. The salient feature of this technique was a combination of end users and experts in the process of requirement prioritization process. However, while implementing this technique, we encountered two major problems.

- The technique generated a lot of conflicts at the end of requirement prioritization process. Conflict resolution was a very long and time consuming process which needed to be followed at the end of every prioritization session.
The technique was completely manual. The prioritization was done through human endeavor and element of human bias was noticeable.

While applying the technique proposed in [48], we realized the need for an automated requirement prioritization technique in order to overcome both the above mentioned limitations. In this paper, we have proposed and elaborated upon a fuzzy logic based intelligent requirement prioritization technique. This technique uses fuzzy logic to prioritize requirements presented by various stakeholders. This modified scheme is basically a multilevel prioritization where end users, experts and intelligent system perform requirement prioritization at various levels. By using this technique, we achieved multiple benefits. On one hand, we achieved certain degree of automation which ultimately resulted in a more efficient technique. On the other hand, we were also able to reduce the emphasis of our approach on skilled requirement engineer solely as now intelligent component as well as end user was also contributing towards ultimate prioritization. At the same time developing this technique also offered us with the opportunity of eliminating the role of requirement engineer altogether since we are currently working on incorporating a neural network based component to replace engineer during second stage of prioritization.

In order to establish the utility of intelligent requirement prioritization technique, we applied this as well as a representative group of other techniques on several projects and determined the degree of success. We have also presented these findings and observations in this paper. Based on these, we have established the case for an artificially intelligent hybridized technique on which we are currently working. We have also presented a framework for future evaluations of results acquired by application of our proposed mechanism with existing requirements prioritization techniques. We have presented preliminary results of application of this framework on various techniques.

The paper is arranged as follows: After a brief introduction in Section 1, we have described a comprehensive literature review in Section 2. In Section 3, we have presented an elaborate overview of existing requirement prioritization techniques. In Section 4, we have described the concept and implementation of fuzzy logic based intelligent requirement prioritization technique. These techniques have been applied on several projects to observe their performance. Findings and evaluations of these experiments have been presented in Section 5. Conclusion and future work is described in Section 6.

1.1. Major contributions. Following are the major contributions:

- In this paper, we have proposed and elaborated upon a fuzzy logic based intelligent requirement prioritization technique. This technique uses fuzzy logic to prioritize requirements presented by various stakeholders.
- This modified scheme is basically a multilevel prioritization where end users, experts and intelligent system perform requirement prioritization at various levels.
- In order to establish the utility of intelligent requirement prioritization technique, we applied this as well as a representative group of other techniques on several projects and determined the degree of success. We have also presented these findings and observations in this paper. Based on these, we have established the case for an artificially intelligent hybridized technique on which we are currently working.
- We have also presented a framework for future evaluations of results acquired by application of our proposed mechanism with existing requirements prioritization techniques. We have presented preliminary results of application of this framework on various techniques.

1.2. Paper organization. The paper is arranged as follows: After a brief introduction in Section 1, we have described a comprehensive literature review in Section 2. In Section
we have presented an elaborate overview of existing requirement prioritization techniques. In Section 4, we have described the concept and implementation of fuzzy logic based intelligent requirement prioritization technique. These techniques have been applied on several projects to observe their performance. Findings and evaluations of these experiments have been presented in Section 5. Conclusion and future work is described in Section 6.

2. Literature Review. Requirement Engineering (RE) is one of the earliest and very critical phases of software engineering. RE as a knowledge stream is basically aimed at acquisition, modeling and documentation of requirements for the software product. Requirement Engineering is a unique discipline in the sense that it not only incorporates the concepts of engineering but also of human and social sciences. Sometimes, referred to as requirements analysis, RE is treated as a sub discipline of system engineering and software engineering. Requirement engineering aims to define precisely the requirements that need to be met. This is not an ordinary task. According to Fred Brooks, deciding what needs to be built is the most difficult part of software development [1]. We can visualize one software requirement as one documented need that software product should accomplish. Usually requirements are classified as either as process based and product based or functional and non functional requirements. Software requirement can best be defined as the description of system functionality along with its quality concerns.

Requirement prioritization is the next logical task once requirements have been elicited and properly analyzed. In most cases, it is really difficult to meet all the requirements that have been given by various stakeholders. Most of the times, elicitated requirements are vague, conflicting or outrightly false. Over period of time, as our understanding of the system becomes more and more clear, the requirements start attaining their actual or specific shape. Similarly, in many cases, requirements are implemented in a staggered fashion. In such circumstances, it becomes important to arrange the requirements in a prioritized order to develop the system in more realistic way. This task becomes even more difficult when performed early in the lifecycle. According to Karlsson et al. [2], one of the greatest problems of software engineers is development of such a product which doesn’t satisfy the needs and expectations of stakeholders. To overcome this problem, same authors came up with the idea of prioritizing the requirements according to their value in the paper [3] titled “A Cost-Value Approach for Prioritizing Requirements”. Subsequently, many other researchers [4,5] emphasized upon the significance of requirement prioritization. According to Ed Yourdon, prioritization of requirements is an extremely important issue [6] where as Lubars et al. stated that prioritization of requirements was one major topic of discussion during the survey that they undertook[7].

There are various methods for prioritizing requirements [8]. Some major techniques are Analytical Hierarchy Process (AHP) [9-11], Binary Search Trees [12-15], 100 points method [16,17], planning game [18,19], numerical assignment technique [20,21] and theory W 20 [22], etc. The consensus of all of these studies is that the project’s success or failure is strongly determined by how effectively, we can prioritize the requirements.

Computational Intelligence and soft computing are established techniques which have helped resolve many real world problems. These methodologies include Artificial Neural Networks, Fuzzy Logic, and Evolutionary Computing etc. Fuzzy logic is a technique centered on fuzzy set theory. Thus it is considered as an extension of classical set theory [23,24]. The concept of fuzzy sets as introduced by Lotfi Zadeh [25] can be considered as generalization of the classical sets which are crisp in their nature. The purpose of fuzzy logic is to both reduce the complexity of existing solutions as well as increasing the accessibility of control theory [26]. Computational intelligence based techniques including
fuzzy logic have been widely used recently to tackle many real world problems. Some of the recent applications of computational intelligence can be found in [27,28]. Fuzzy logic has also found its way in software engineering where it has most recently been used in effort estimation [29], software project similarity [30], software development [31], project evaluation [32], software evolution [33] etc. Fuzzy logic has been used in requirement engineering as well for various tasks [34,35]. Chengdong Li et al. [36] presented a novel approach for using prior knowledge and sampled data in fuzzy systems. Some other recent developments in the domain of fuzzy logic [37-39] have also presented new vistas of research in software engineering. What is evident after studying extensively is the fact that researchers in software engineering need to apply artificial intelligence in various domains of knowledge of SE to propose such techniques which are evolvable and can intelligently generate efficient results. In this paper, we have proposed another application of fuzzy logic in software engineering. We have proposed to use fuzzy logic in the domain of requirement engineering. We suggest introducing fuzzy logic to determine the priority of requirements. The next section is devoted to elaboration of this proposed technique.

In this paper, we propose another application of fuzzy logic in the domain of requirement engineering. We suggest introducing fuzzy logic to determine the priority of requirements. The next section is devoted to elaboration of this proposed technique.

3. Requirement Prioritization Techniques: An Overview. As mentioned in the literature review, there are various requirement prioritization techniques. However, no evaluation of these techniques has been made so far so that their utility and relevance can be determined. In this section, we give a comprehensive overview of various requirement prioritization techniques.

3.1. Analytical hierarchy process (AHP). AHP is a relative assessment based statistical technique to prioritize requirements for software products. If we have $n$ number of requirements, AHP makes $n \times (n-1)/2$ comparisons at each hierarchy level. In real life, we are usually working with requirements which have multiple objectives. AHP works as an efficient technique in these kinds of situations by making pair wise comparison to calculate relative value and cost of each requirement against the other one. This significantly large number of comparisons makes the technique less effective as increase in number of comparisons always takes place at the rate of $O(n^2)$. AHP is regarded as a five step method.

1. Establish completeness of requirements.
2. Apply the pair-wise comparison method to assess the relative value.
3. Apply the pair-wise comparison method to assess the relative cost.
4. Calculate each candidate requirement’s relative value and implementations cost, and plot each on a cost-value diagram.
5. Use the cost-value diagram as a map for analyzing the candidate requirement.

A sample cost value diagram is shown in Figure 1. A large number of studies have been made in recent past to determine the effectiveness of AHP for requirements prioritization. Karlsson [3,40] has made a number of studies which have shown the effectiveness of this technique in industrial settings. At the same time, some other studies [19,41] have talked about AHP as being difficult, less efficient and time consuming. AHP can be considered as a highly sophisticated and complex technique which can establish prioritization at the level of individual requirements.

Efforts have been made to reduce the number of comparisons. However, this has always enhanced the margin of error. In our opinion, this tradeoff is necessary since some comparisons may actually never be needed.
3.2. **Cumulative voting (CV).** Also referred to as 100 $ test or 100 point method sometimes, resembles very much to voting mechanism of brainstorming sessions. Each stakeholder is given 100 points that he or she can distribute among the requirements as they seem fit. It seems like very straightforward mechanism but it becomes complex as the number of requirements increases or the stakeholders involved become too many. This scheme also has several drawbacks associated with it. Firstly this scheme treats all the requirements as equal opportunity candidates. Secondly, the element of bias can never be over ruled. It has been observed that in second or subsequent voting, stakeholder assign more votes to their favorite requirements in order to move them up. Many researchers [42-44] have pointed out shortcomings in cumulative voting mechanism. Cumulative voting technique can also be considered as one which is complex in its nature but attempts to prioritize requirements at their individual level.

3.3. **Numerical assignment (NA).** It is probably the most common prioritization technique which is also very easy to use. In the first step, requirements are classified into different groups. These requirements are given to each stakeholder. Each requirement within these groups is assigned a number on a scale of 1 to 5 by individual stakeholders. The final ranking is determined by calculating average of all the ranking given to each requirement by every stakeholder. This technique because of its ease of use has also been suggested by IEEE Std. 830-1998. Since the requirements are first classified into groups and then prioritized so we can say that this technique does not prioritize requirements at the level of individuality. Instead one level of abstraction is introduced.

Despite its wide applicability, this technique also poses several problems. Clear definition of the groups is one major drawback. Second problem is that even with clear definitions, stakeholders will tend to put most of their requirements into critical groups because of their bias (which can’t be overruled). Another fact that we have to be mindful about is that within each group, all the requirements are initially at the same priority level. Most of these drawbacks in numerical assignment technique have been well documented in [42,45,46].

3.4. **Ranking.** This technique is more suitable in the environment where a single stakeholder is involved. If there are $n$ number of requirements, these requirements are ranked from 1 (most significant) to $n$ (least significant). This ranking is exclusive in its nature because requirements are not ranked relative to other requirements as is the case of AHP.
or cumulative voting. Various techniques like bubble sort, quick sort or binary search techniques can be used to achieve this ranking.

There are two major drawbacks associated with this technique. First major problem is that it can cause more conflicts than agreements when applied in an environment of multiple stakeholders. The second drawback is that requirements are viewed and ranked in isolation. The impact of one requirement over the other doesn’t play any role in overall prioritization. Since requirements can have multiple dimensions to them so researchers have devised a mechanism of combining all of these dimensions and calculating a mean priority for each requirement [47]. This modification has its own limitations as well as has been shown in [47].

3.5. **Top-ten requirements.** This technique prioritizes only the most important requirements into a set of top-ten from a larger set of requirements. Selection of the most important requirements is subjective to the project environment and so it can be erroneous if based on human judgment. Since we create only a set of top-ten requirements, no prioritization within this set takes place. This can be termed as a shortcoming in many situations. The technique can be applied in conjunction with other techniques to achieve better results. According to lausen [48], it is mostly helpful in situations where there are multiple stakeholders with uniform or similar significance.

3.6. **Theory W.** The main proponent of this theory is Dr. Barry Boehm who introduced this concept [49] in 1989. Popularly known as Win-Win model, this technique relies heavily on negotiation to resolve any differences of opinion among various stakeholders. The negotiations are conducted in such a way that each stakeholder is in a “Win” situation. The principles of this technique are progress based on predefined plan, risk assessment and risk handling. In this technique, users are asked to rank their requirements before actual negotiations start. Users are asked to carefully consider which requirements they are willing to negotiate and which they are not. Theory W has been an active area of research among scholars which has been applied in not only requirement engineering but also in other domains of software engineering. Theory W is a major constituent of Value Based Software Engineering (VBSE) agenda and principle as well.

3.7. **Planning game (PG).** This specific requirement prioritization technique is very suitable to extreme programming. In this specific technique, requirements are prioritized in consultation with customers. This is a variation of numerical assignment technique as discussed in Section 3.3. However, it offers more flexibility than numerical assignment where users are asked to essentially divide the requirements into three groups.

Some other new and innovative techniques to emerge recently include Requirement Triage (RT) [50] and Wieger’s Method (WG) [51]. In requirement Triage, each requirement is prioritized relative to the resources that are necessary to meet that specific requirement. In this way, a subset of requirements is selected which can optimize the probability of success of product while using the allocated resources efficiently. In Wieger’s method, the priority of each requirement is set by determining the utility of that requirement to the customer as well as penalty incurred by the customer if that requirement remains unfulfilled.

In this section, we have presented a brief overview of existing requirements prioritization techniques. In this next section, we shall present a theoretical evaluation of these techniques as well as present the brief idea of our proposed and implemented approach.
4. **Value Based Intelligent Requirement Prioritization (VIRP): The Proposed Technique.** Value based intelligent requirement prioritization (VIRP) as shown in Figure 2 is basically a multilevel prioritization and classification technique. This technique involves the use of stakeholder’s, experts and automated fuzzy logic based system at various stages to iteratively prioritize the requirements. This iterative prioritization ensures that requirements are evaluated again and again by different actors and a more meaningful and realistic result is achieved.

**Figure 2.** Flow graph of fuzzy based intelligent requirement prioritization process

4.1. **Requirement elicitation and stakeholder level prioritization.** Requirement elicitation process starts at the early stages of requirement engineering. This is a standard process that is followed in any requirement engineering technique. In order for our technique to gather its first level prioritized requirements, we introduce two new concepts in our requirement elicitation.

- The requirement elicitation is done electronically. All stakeholders submit their requirements on specially designed web scripts. Stakeholders submit their requirements according to their own priority.

A new concept of stakeholder profiling is introduced. While submitting their requirements, stakeholders also give a brief description regarding themselves, their expectations about the system, system functionalities of their choice etc. Requirements engineers assigned to gather requirements also write down their observations regarding those stakeholders independently. Later both of these are combined to form a stakeholder profile. Stakeholder profile template is given in Figure 3. These elicitated requirements as well as stakeholder’s profile are stored in specially designed requirements database.

4.2. **Expert level prioritization.** Once all the requirements from stakeholders along with their profiles have been elicited, these are submitted to the experts for second level prioritization. In this stage, experts perform three tasks.
(a) Success in prioritization achieved for different process models

(b) Success ratio achieved for two projects with varying degree of requirement clarity

(c) Success ratio achieved for two projects with different level of requirement change

(d) Success ratio for two projects with varying number of stakeholders and their commitment
(e) Success ratio achieved for two projects with different level of constraints i.e. budget, time etc.

**Figure 3.** Experimental evaluation results of various prioritization techniques

In the first task experts quickly review all the requirements given by stakeholders independent to the stakeholder’s profile and the requirements given by other stakeholders and modify the prioritizations given by stakeholders in a quick manner if some glaring problem emerges.

In the second task, experts review each stakeholder profile and allot it a score in the range of 1-10 based on his significance and the overall impact that requirements posed by him may have on the success of the project.

In the third and most significant task, experts assign value to requirements in order to prioritize them. This valuation is based on certain requirements classification factors (RCFs) which can enhance the overall value of the requirement with their presence. $RCF_i$ can be defined as “one factor whose degree of presence or absence in a requirement can have a direct bearing on its value. We have identified ten requirement classification factors whose names and brief descriptions are given in Tables 1 and 2. These RCF are divided into two groups.

4.2.1. *Project specific RCFs.* These RCF illustrate the value of requirement in relation to the project for which it has been elicitated. The importance of requirement for the project is determined through these RCFs. These include feasibility, modifiability, urgency, traceability and testability. Better value for these RCFs means that this requirement can better enhance the quality of end product. These are represented as $pRCF$. Their detailed description is given in Table 1.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Feasibility</td>
<td>The requirement is capable of being implemented within the constraints and resources</td>
</tr>
<tr>
<td>2 Modifiability</td>
<td>Requirement can undergo change to optimize the system without affecting the system adversely</td>
</tr>
<tr>
<td>3 Urgency</td>
<td>Degree of necessity of the requirement for system to be considered successful</td>
</tr>
<tr>
<td>4 Traceability</td>
<td>Requirement is such that subsequent function of the system can be traced to it. Requirements are less compound</td>
</tr>
<tr>
<td>5 Testability</td>
<td>Requirement can be tested and validated during testing phase. Independent test cases for the requirement can be generated</td>
</tr>
</tbody>
</table>
4.2.2. Requirement specific RCFs. These RCFs relate to those attributes of requirements which if implemented increase the overall value of requirement from the perspective of its description quality. These include completeness, consistency, understandability, within scope and non-redundancy. Collectively, these are represented as \( rRCF \). Their detailed description is given in Table 2.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Completeness</td>
<td>The requirement statement has enough information to proceed to the next development phase</td>
</tr>
<tr>
<td>2 Consistency</td>
<td>Requirement specifications use standard terminology and there are minimum conflicts due to statement and specifications</td>
</tr>
<tr>
<td>3 Understandability</td>
<td>Requirements are easy to describe and review. Requirements are grammatically correct with single and clear meaning</td>
</tr>
<tr>
<td>4 Within Scope</td>
<td>Requirement does not envisage something which is not described in original statement of scope</td>
</tr>
<tr>
<td>5 Non-Redundant</td>
<td>Requirement is not duplicated in complete or partially</td>
</tr>
</tbody>
</table>

Each factor is given a certain score in the range of 0 – 5 depending upon the effect of that factor on the value of given requirement. 0 denotes the total absence of that specific RCF in any requirement whereas a value of 5 denotes the perfect presence of RCF in that requirement. All RCFs are given scores between 0 and 5. The scores of these RCFs are used to determine the final value of the given requirement. Using the equation given below, the requirement value (RV) is calculated.

\[
RV = 0.35 + 0.02 \left( \sum_{i=1}^{5} pRCF_i + \sum_{i=1}^{5} rRCF_i \right) \tag{4.1}
\]

Requirements are prioritized using their RV which can be a number between 0.35 and 1.35. This exercise gives the prioritized requirements with respect to each factor. There are two exceptions where we have introduced additional measures.

First Scenario
In case of situations where any two or more requirements have similar RV, we use \( pRCF \) as deciding measure to assign priority using following rule:

If we have two requirements I and j with value \( RV_i \) and \( RV_j \) such that \( RV_i = RV_j \), then

\[
\text{if } \sum pRCF_i > \sum pRCF_j \text{ then } \Rightarrow \text{ high priority } RV_i \]

\[
\text{else } \Rightarrow \text{ high priority } RV_j
\]

Second Scenario
In case of requirements where \( RV_i = RV_j \) and \( \sum pRCF_i = \sum pRCF_j \) as well, we use prioritized stakeholder’s profile to determine which stakeholder is more important among the two competing ones.

The requirement posed by more significant stakeholder is prioritized higher in that case. In case of both competing requirements coming from the same stakeholder, we leave it to the judgment of experts to assign higher priority to any requirement which he deems fit.
At the completion of these three tasks, we have a complete unified list of requirements in their prioritized order and a prioritized list of stakeholders. Both of these are submitted to Fuzzy logic based component to perform third level prioritization.

4.3. **Fuzzy logic based requirement prioritization.** Following steps are executed in this third and final level of prioritization:

Fuzzy c-means (FCM) is a method of clustering which allows one piece of data to be in the right position to two or more clusters. This method (developed by Dunn in 1973 and improved by Bezdek in 1981) is frequently used in pattern recognition. However, the same model can be applied after suitable modifications for requirement prioritization which we have accomplished. FCM starts with an initial guess for the cluster centers, which are proposed to mark the mean location of each cluster. The initial guess for these cluster centers is most likely incorrect. Additionally, FCM assigns every data point a membership rank for every cluster. By iteratively updating the cluster centers and the membership grades for each data point, FCM iteratively moves the cluster centers to the correct place within a dataset. This iteration is based on minimizing an objective function that symbolizes the distance from any given data point to a cluster center weighted by that data point’s membership rank.

The FCM algorithm assigns requirements to each group by using fuzzy memberships. Let $XZ(x_1, x_2, \ldots, x_N)$ indicates a requirement with $N$ variables to be partitioned into $c$ clusters, where $x_i$ represents multispectral (features) data. The algorithm is an iterative optimization that minimizes the cost function defined as follows:

$$J = \sum_{j=1}^{N} \sum_{i=1}^{C} U_{ij}^m \| x_j - V_i \|^2$$  \hspace{1cm} (4.2)

where $u_{ij}$ represents the membership of feature $x_j$ in the $i$th cluster, $v_i$ is the $i^{th}$ cluster center, $\| \ldots \|$ is a norm metric, and $m$ is a constant. The parameter $m$ controls the fuzziness of the resulting partition, and $m = 2$ is used in this study. The cost function is minimized when requirements close to the centroid of their clusters and are assigned high membership values, and low membership values are assigned to requirements with data far from the centroid. The membership function represents the probability that a requirement belongs to a specific cluster. In the FCM algorithm, the probability is dependent solely on the distance between the requirement and each individual cluster center in the feature domain. The membership functions and cluster centers are updated by the following:

$$U_{ij} = \frac{1}{\sum_{k=1}^{c} \left( \frac{x_j - v_i}{x_j - v_k} \right)^{\frac{2}{m-1}}}$$  \hspace{1cm} (4.3)

and

$$v_i = \frac{\sum_{j=1}^{N} U_{ij}^m x_j}{\sum_{j=1}^{N} U_{ij}^m}$$  \hspace{1cm} (4.4)

Starting with an initial guess for each cluster center, the FCM converges to a solution for $v_i$ representing the local minimum or a saddle point of the cost function. Convergence can be detected by comparing the changes in the membership function or the cluster center at two successive iteration steps.
5. Theoretical and Experimental Evaluation of Requirement Prioritization Techniques. In Section 3, we have given a brief overview of existing requirement prioritization techniques. In Section 4, we have briefly described the implementation of intelligent requirements prioritization technique. Several of these techniques are being applied in software industry for a while now. Some research studies and surveys to determine the usability of some of these techniques have also been conducted (as mentioned in previous section). So a significant mass of literature on these requirement prioritization techniques exists.

However, no significant comparative study has so far appeared where all or most of the above mentioned techniques might have been applied to the same set of projects. This can be a very valuable study as it can determine that in what kind of development environment which specific requirement prioritization technique can yield best results. As we have already mentioned in the introduction section, we faced a severe problem of selecting suitable requirement prioritization technique for our projects. This search ultimately culminated in proposing and implementing our own intelligent requirement prioritization technique. We feel that it is need of the hour to catalogue the pros and cons of all existing as well as proposed technique at one place so that it becomes easier for software engineering community to evaluate and select one technique which better meets its needs.

This section is dedicated to theoretical evaluation of requirement prioritization techniques. It is further subdivided into three subsections. In the first section, we have presented a theoretical evaluation of existing requirement prioritization techniques. In the second part, we have given a brief introduction of our intelligent requirement prioritization technique. In the last subsection, we have presented experimental results for our analysis of this technique with all existing ones.

5.1. Analysis of existing requirement prioritization techniques. Literature offers us many insights into the intricacies and working of various requirement prioritization techniques. We have gathered certain valuable findings while working with these techniques as well. In this section, we shall briefly elaborate upon those techniques.

Our experience has shown that for large projects with multi-objective requirements, AHP is a more preferred approach among the professionals. This technique yields statistically very reliable results. The experience has also shown that the cost of conducting AHP is also marginal as compared to various other techniques. This technique however is not suitable in the situation where requirements are fast evolving and new requirements are being introduced at a much higher pace. The purely statistical nature of AHP (and it is true for other techniques described above as well) makes it difficult to generate a prioritization which accommodates these changes taking place. The technique due to it’s highly time consuming nature also becomes an unfit solution for development models where several iterations take place (unless a sufficient time box is available for prioritization process at each iteration). Cumulative voting is a human intensive exercise. In our experience, while working with iterative process models, we were able to plan our projects in such a way that cumulative voting could be conducted at every iteration. Since cumulative voting involves human insight apart from statistical techniques, we also experienced a more flexible and accommodating prioritization when handling changing and creeping requirements. The cost factor for cumulative voting becomes an inhibiting factor when we deal a project of several hundred requirements and a very tight budget. The element of bias was also visible in some of our experiments as experts inadvertently prioritized those requirements which they thought were more important from their perspective. We believe that AHP is more suitable for projects with medium number of requirements and
waterfall or prototyping model. Cumulative counting on the other hand, can manage iterative development quite efficiently provided enough budget and expertise is provided. We have also observed that some kind of automation is required for both of these techniques. This automation can reduce the time requirement of AHP and make it more suitable for iterative development while it can also reduce the element of bias for cumulative counting and lend it more credibility.

Numerical assignment technique was one of the most difficult to work with in our experience. Despite it being the most commonly used technique, we face almost insurmountable problems while working with numerical assignment. This technique was rendered useless when working in iterative environment. It was difficult to identify and gather all the stakeholders in each iteration, determining the exact status of their requirements (including all changes, creeps and incomplete) and then performing classification based prioritization. Our experience has shown us that numerical assignment is very unreliable when software is to be developed in iterations, has several stakeholders and fluidity of the requirements is very high. Some degree of success was achieved where the stakeholders are very few and highly oriented. Second problem while working with numerical assignment was the much greater degree of bias that was exhibited by stakeholder’s when prioritizing than the bias we experienced in the case of cumulative voting. Third problem that we experienced was that classification posed problems instead of solutions. It was because a large majority of requirements posed by various stakeholders were actually placed in the highest classification by their owners while requirements posed by other stakeholders were put in lower classifications as those were considered less important.

Ranking was another prioritization mechanism which had very low potential in modern day development in its true sense. Exact problems as mentioned in Section 3.4 were faced while working with ranking technique. Top ten techniques is good at establishing a set of the most critical requirements. Our experience has shown that all these three techniques are very difficult to work with and can’t meet the objectives of requirement prioritization in an optimal way.

Theory W is a very valuable requirement prioritization technique. It has a two tier prioritization system which works within predefined limits. Stakeholders are given the opportunity to prioritize their own requirements which are then further studied and adjusted by experts before those are presented to all for negotiations. These negotiations last until we have asset of requirements in such a prioritized order that every stakeholder is a winner. We were able to get much better results by applying theory W than any other technique. The major problem that we faced while working with theory W was when requirements were fluid beyond certain degree. It was impossible to perform negotiations at each iteration. So the utility of this technique was somewhat diminished in iterative development with highly evolving and changing requirements. Planning game is also a better variant of numerical but the same problems persist (with somewhat less intensity).

Wieger’s method and requirement triage are relatively new entrants in the field of requirement prioritization. These techniques offer solutions to the problem of requirement prioritization which are more realistic and are more in sync with ground realities. These techniques are good in both linear and iterative process models. Our experimentations and observations have shown that AHP and cumulative voting are best existing techniques for linear and iterative models respectively. However, AHP being a time consuming and purely statistical technique needs certain improvements. On the other hand, cumulative voting is very costly and has the margin of bias and error due to total dependence on human experience. Apart from these shortcomings, the aspect of automation also demands certain attention. All these techniques are either statistical or human based. In order to be more reliability and flexibility in our requirement prioritization mechanism, we need
to develop a computer based approach which can prioritize requirements to a large extent on its own. Keeping all of these observations in mind as we applied all the above mentioned techniques, we conceptualized and implemented a fuzzy logic based intelligent requirement prioritization approach. In the following sub-section, we briefly describe our requirement prioritization technique. After that, we shall present our experimental results comparing the performance of all existing as well as new technique.

5.2. Experimental results for requirement prioritization techniques. The theoretical analysis that we have presented in Section 4.1 is based upon the application of these techniques as well as our proposed technique. The experiments were conducted on ten different projects. These projects were ongoing university level projects. Some of these projects were sub-contracted from industry while some others were government sponsored projects. These projects were carefully selected to represent the vast spectrum of different natures of requirements. The brief description of all of these selected projects is given in Table 1. All nine requirement prioritization techniques presented in Section 3 as well as proposed technique presented in Section 4.2 were applied on all of these techniques. The utility of each technique was determined by the degree of success in accurate requirement prioritization determined by a panel of experts at the completion of the project. Selection criterion description is as follows:

**Process Model**
As we have described earlier, the process model has significant bearing on the success or failure of selected process model. The major factor to consider in process model for requirement prioritization is whether the process model is linear or iterative in its nature. After careful analysis we selected Mobile Business Intelligent Tool as the project which was developed in iterative fashion while ALIF was selected for linear development model.

**Level of Ambiguity**
A couple of projects were selected on the basis of degree of ambiguity in their requirements posed by various stakeholders. This was aimed at determining the efficiency of various techniques in the presence of high ambiguity requirements as well as when the requirements are very clear and formal. Sign Language Recognition and Interpretation was selected as a project which had a high degree of ambiguous and faulty requirements whereas Semantic Web Based Clinical Decision Support System had high degree of precision and clarity in its requirements.

**Requirements Fluidity**
Most of the times we are faced with situations where requirements are highly fluid. Requirements change too often. Many creeping requirements emerge during the development process. Sometimes existing requirements lose their relevance or simply cease to exist. This affects the overall quality of requirement prioritization. We selected a representative project titled “Content Based Image Retrieval System” which had highly fluid requirements while another project “beyond RADIX” was selected due to low degree of fluidity in its requirements.

**Stakeholders**
Number of stakeholders and their degree of involvement in the project also impacts the overall requirement prioritization effort. We selected “site management system” as a project which had few stakeholders but their involvement in the project was high. On the other hand “i Pakistan Sign Language” was selected as a representative project where several stakeholders are involved and their involvement in the project is significantly low.

**Project Constraints**
Project constraints (time, budget, human resource etc.) have a high impact on determining the utility of any requirement prioritization technique. We selected two project
Table 3. Selected projects and selection criteria

<table>
<thead>
<tr>
<th>Project ID</th>
<th>Project Name</th>
<th>Project Acronym</th>
<th>Selection Criteria</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Mobile Business Intelligent tool</td>
<td>MBIT</td>
<td>Iterative development</td>
</tr>
<tr>
<td>2</td>
<td>Academic Learning and Information</td>
<td>ALIF</td>
<td>Linear Development</td>
</tr>
<tr>
<td></td>
<td>Framework</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Sign Language Recognition and</td>
<td>SLR</td>
<td>Ambiguous Requirements</td>
</tr>
<tr>
<td></td>
<td>Interpretation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Semantic Web Based Clinical Decision</td>
<td>semclin DSS</td>
<td>Clear and Formal requirements</td>
</tr>
<tr>
<td></td>
<td>Support System</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Beyond RADIX</td>
<td>BRADIX</td>
<td>Low fluidity requirements</td>
</tr>
<tr>
<td>6</td>
<td>Content Based Image Retrieval System</td>
<td>CIRS</td>
<td>High fluidity requirements</td>
</tr>
<tr>
<td>7</td>
<td>Intelligent Site Management System</td>
<td>ISM</td>
<td>Few stakeholders, High Commitments</td>
</tr>
<tr>
<td>8</td>
<td>Pakistan Sign Language</td>
<td>iPSL</td>
<td>Several Stakeholders, Low to</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Medium Commitment</td>
</tr>
<tr>
<td>9</td>
<td>Semantic Bees</td>
<td>SeBe</td>
<td>High Project Constraints</td>
</tr>
<tr>
<td>10</td>
<td>IMS Intrusion Detection System</td>
<td>IIDS</td>
<td>Relaxed Project Constraints</td>
</tr>
</tbody>
</table>

Figure 4. Performance comparison of leading prioritization techniques

Constraints for our experimentation. These were budget and schedule. Based on this criteria, semantic bees was selected as project which had high constraints (low budget and tight schedule) whereas “IMS Intrusion Detection System” was selected as project with low constraints (ample budget and sufficient time for scheduling).

The experiments were conducted on two different levels. On first level, the techniques were applied on different projects to determine their degree of success in precise prioritization under different environments. The results have been shown in Figures 3, 4 and 5 respectively. Major findings as highlighted in Figure 3 were as follows:

- Projects 1 and 2 were selected to determine the performance of various techniques on projects that follow iterative and linear process models respectively. We have observed that results of prioritization often get faulty when we adopt iterative process models since requirements change quite often. This is true for almost all requirement prioritization techniques presented in literature. However, the results have shown that intelligent requirement prioritization was able to best prioritize requirements.
Figure 5. Empirical evaluation of prioritization techniques
when iterative model was adopted and it was followed by cumulative voting. In case of linear model, intelligent requirement prioritization, AHP and theory W yielded best results with intelligent requirement prioritization able to accurately prioritize more than 90 percent of the requirements. The reason for theory W not performing better in case of iterative development was that with every iteration, the degree of involvement of stakeholders reduced so many ambiguities were introduced in later stages. On the whole, it was observed that quality of all techniques deteriorated in iterative development model. Overall intelligent requirement prioritization techniques performed better and more consistently in both kinds of process models.

- Projects 3 and 4 were put to test with special emphasis on degree of clarity of requirements. It is difficult to understand the utility and ultimate value of requirements if requirements are not very well elaborated. In that case it becomes very important to use such mechanism which can support the application of domain knowledge to clarify the requirements to a greater extent. It was observed that in case of project 3, where requirements were highly ambiguous, all the techniques delivered relatively poor prioritization results. Many of the well established techniques showed around 50% of error. The better performing techniques in this case were cumulative voting, wieger method and intelligent requirement prioritization techniques. However, theory W performed the best in this case with 68% accurate prioritization. The success of Theory W, cumulative voting as well as intelligent requirement prioritization can be attributed to the wise and meaningful utilization of end users and customers in an environment where perhaps customers themselves can better understand their given requirements. Similar techniques exhibit leading results in case of project 4 where requirements are much clear and formally written. However, it is interesting to note that precision and formality in requirement specifications can significantly increase the accuracy of requirement prioritization (90% for project 4 against 60% for project three in the case of intelligent requirement prioritization). Overall, intelligent requirement prioritization showed much better results than almost all techniques for projects 3 and 4.

- Projects 5 and 6 dealt with degree of change of requirements during the course of project execution. It was observed that in case of project 5, where degree of change in requirements was very low, almost all the leading techniques showed very good results with theory W achieving 90% accurate prioritization. Many other leading techniques like AHP, cumulative voting, wieger method and intelligent requirement prioritization were also able to achieve 85% or more accurate prioritization. Project 6 was a project with higher degree of change in project requirements. The results degraded for all of these leading techniques except for intelligent requirement prioritization. In fact intelligent requirement prioritization was able to prioritize requirements to a degree of 88 percent accuracy which is quite impressive. This was due to multilayer nature of intelligent requirement prioritization technique which involved stakeholders, experts and fuzzy logic at the same time. The resultant prioritization exhibited more farsightedness and degree of anticipation of change than all other techniques. This factor is important in modern day software development as requirements change and evolve exceptionally quickly now.

- Projects 7 and 8 experimented with degree of involvement of stakeholders. Intelligent requirement prioritization achieved above 90% accuracy for project 7 which was much better than all other techniques. High commitment of stakeholders was instrumental in achieving these results. Theory W also exhibited impressive results by achieving 80% accuracy. In case of project 8 where level of commitment of stakeholders was low, the quality of prioritization achieved by all techniques degraded. Intelligent
requirement prioritization achieved 73% accuracy followed by AHP which achieved 69%. Better performance on intelligent requirement prioritization was due to the fact that its prioritization mechanism did not rely too heavily on stakeholders. It used experts and fuzzy logic as well to achieve prioritization. AHP also showed better results than other techniques because it was a statistical method with very low reliance on stakeholder’s opinion. The experimental results substantiated the fact that in several cases lack of proper involvement of stakeholders can actually degrade the quality of end product.

- The last two projects were used for experimentation to determine the impact of project constraints on the accuracy of prioritization. These included constraints such as time, human resource, capital etc. The results have shown that intelligent requirement prioritization was able to prioritize requirements accurately to a degree of more than 80% in both the cases. Prioritization techniques yielded better results on the whole in the case of relaxed project constraints. In case of high constraints, all techniques showed relatively poor performance as compared to intelligent requirement prioritization. However, when the project teams were allowed to work in a relaxed environment, the performance and accuracy of all prioritization techniques improved considerably.

The comparative analysis of leading techniques is shown in Figure 4. As the results show, in most of the environment, intelligent requirement prioritization was able to perform better than any other requirement prioritization. We have created a profiling of various requirement prioritization approaches based on our experimentation. The brief profile of these techniques is given in Table 4. This profile shows that when all the factors are combined together, value based intelligent requirement prioritization technique performs better than any existing approach.

Another evaluation of all techniques was conducted from the perspective of time consumed, average cost, precision and consistency. The results have been graphically represented in Figure 5. The results clearly demonstrate that intelligent requirement prioritization techniques is less time consuming with more precision and consistency than any other requirement prioritization technique. However it is costlier than some other techniques owing to the fact that stakeholders and experts are involved in first and second level prioritization. When the increase in cost is viewed in conjunction with other factors, the overall cost increase is quite negligible.

As was mentioned earlier, this evaluation exercise was conducted on a limited number of projects and was on academic basis. The need is to apply all these techniques on industrial level in future and to gather data accordingly. This can further improve visibility into the working and performance of these techniques. At the same time, a more comprehensive evaluation and profiling can be generated.

6. Conclusion and Future Work. Requirement prioritization is one important activity of requirement engineering phase in software development. There are various requirement prioritization techniques in literature and practice. However, no significant comparative evaluation of these techniques has been made so far. In this paper, a new intelligent requirement prioritization technique has been proposed and described. This new technique for requirement prioritization is based on fuzzy logic and is a multilevel approach. In this technique, stakeholders, experts and fuzzy logic based system perform separate prioritizations at three different levels. A comparative analysis based on experimental results conducted on several projects has also been presented. This analysis shows that in almost all different environments, intelligent requirement prioritization is able to exhibit better and impressive results.
### Table 4. Requirement prioritization techniques profile

<table>
<thead>
<tr>
<th>Technique name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AHP</td>
<td>Better results when used in linear development model, low fluidity in requirements and less constrained projects. Not suitable iterative model and ambiguous requirements.</td>
</tr>
<tr>
<td>Cumulative Voting</td>
<td>Works good in all the environments where end users and stakeholders involvement is vital. Better than other techniques when requirements are ambiguous.</td>
</tr>
<tr>
<td>Numerical Assignment</td>
<td>This technique is not recommended except for the situation when there is a low degree of change in the requirements. However, many techniques outperform numerical assignment in this situation as well.</td>
</tr>
<tr>
<td>Ranking</td>
<td>Only marginally effective when using linear development model. In all other situations, the performance of this technique is much lower than most other approached.</td>
</tr>
<tr>
<td>Top Ten Requirements</td>
<td>Partially effective when there is low degree of change in requirements. Many other techniques perform much better than this approach in most situations.</td>
</tr>
<tr>
<td>Theory W</td>
<td>The best among currently established approaches. Work better than any other technique currently in practice. Performance of the prioritization slightly deteriorates in iterative development model and when project constraints are too high.</td>
</tr>
<tr>
<td>Planning Game</td>
<td>Not an affective prioritization approach. Yields relatively better results when requirement specifications are very clear.</td>
</tr>
<tr>
<td>Requirement Triage</td>
<td>A new technique with encouraging results in various situations. Best suited when requirement specifications are very clear and there is low degree of change in these requirements. However, some other techniques like Theory W and Intelligent Requirement Prioritization perform better.</td>
</tr>
<tr>
<td>Wieger Method</td>
<td>Another such technique which yields better prioritization when requirements are clearly specified and there is low degree of change. Not much effective in other situations.</td>
</tr>
<tr>
<td>Intelligent Requirement Prioritization</td>
<td>The best performing technique in almost all situations. Only other technique to generate comparative results is theory W. The only situation when quality of prioritization is less than satisfactory is when requirements specifications are not very clear.</td>
</tr>
</tbody>
</table>

There is an urgent need to extend this work towards classification of the prioritized requirements. Work is already underway towards extending this application in such a way that it can automatically classify requirements as critical, essential, peripheral etc. Work can also be done in such a way that prioritized requirements can be classified as non-negotiable and negotiable requirements. We are also in the process of developing neural networks to perform expert level prioritization. Using neural networks can make this system highly evolvable and self optimizing.

**Acknowledgments.** The authors, Mr. Muhammad Ramzan and Mr. Arfan Jaffar would like to acknowledge the Higher Education Commission (HEC), Govt. of Pakistan and NU-FAST for providing funding and required resources to complete this work. It would have been impossible to complete this effort without their continuous support.
REFERENCES