WHALE RANKING

Introduction

Information technology plays a significant role in our day-to-day life and software engineering is one of the major branches. Software engineering is an interesting field, which aims at designing software to meet the consumer demands. There are plenty of challenges in considering the degree of the software product with respect to software engineering. In most of the software products, one can find different categories of the stakeholders, who aim at receiving their requested software product that prioritizes their requirements at the final delivery. Any of the following members like the users, customers, project managers, product managers, developers, and testers are the stakeholders. They hold positions in software engineering, and therefore, their requirements for the products are based on their immediate need or emergency. However, the complexity of the software system gets increased, and therefore, the practitioners face extra pressure to develop software with the demanded requirement and also, to complete the task within the predefined time. This pressure makes the software requirement prioritization, a prime importance and it is one of the best solutions to avoid the trade-offs between the conflicting requirements. In other words, one can say that the requirement prioritization plays a vital role in the decision-making process corresponding to software development. The importance of the requirement prioritization is highlighted in some research studies that pointed the problems of requirements prioritization in the software engineering domain over the years.

1.1 REQUIREMENTS PRIORITIZATION

Requirements prioritization plays a vital role in every aspect of our life mainly, in the project selection process, requirement finalization, module development and design, software testing, implementation, and in turn, in the post-implementation processes. Thus, it is very clear that the requirement prioritization is the complex decision-making process. The objective of requirement prioritization is that it identifies the most significant requirement of the software from the list and extracts the significant knowledge of the system to impact the effectiveness. The most important points to consider during the requirement prioritization include: Identifying the important requirement from the given activities, to determine the order of the requirements to implement during the software release, to achieve the settlement regarding the scope of the project with their existing limitations. The limitation list includes the schedule, budget, time, and quality. Also, the objective aims at establishing equilibrium between the cost and benefit of a specific requirement and therefore, to accept this equilibrium balance as possible equilibrium to implement. Moreover, the objective includes the reduction of the impacts caused by any one of the requirements over the architecture of the software and the final deliveries mainly, on the cost of the product. Finally, to meet the optimum satisfaction of the customers and to determine the factors that meet the customer satisfaction.

Since prioritizing the requirements is the important concern in planning the release, there are various methods to undergo prioritizing. So we concentrate on software requirement prioritization that uses an efficient prioritizing method named as WhaleRank. It uses four ranking functions based on the dictionary words, a similarity matrix, the perception of the manager, and the newly updated requirements. These ranking functions construct four different ranks, which then combines to form a linear rank function. The linear rank function is the combination of the four rank functions that gets combined using the ranking constants. These ranking constants when summed equals to one. The ranking constants are optimally determined using the WOA. WOA determines the optimal position, and it is a population-based algorithm, and it overcomes the performance of PSO, ACO, and so on. Finally, the best rank order gets retrieved using the fitness function. This method of ranking based on WhaleRank achieves customer satisfaction by efficiently prioritizing the requirements based on the customer review. It is clear that this method contributes to enhancing the quality of the product.

1.2 RELATED WORK

Prioritizing the requirements is the important concern in planning the release, there are various methods to undergo prioritizing. Some of the methods include Analytical Hierarchy Process (AHP), the weighted sum method and cost-value analysis. In situations when the requirements are not dependent on each other, scheduling the release planning is an easy task that duly depends on the priorities generated using the above techniques. These techniques establish the priority based on two criteria, namely available resources and the delivery date. Additionally, CBRank(Anna Perini et. al,

2013) provides the following solutions. It formally defines the prioritization problem to solve, and then describes the machine learning method used. Then, it characterizes the prioritization process that gets supported using CBRank, important points of the method gets assessed through a comprehensive overview of the empirical measurements and finally, CBRank gets positioned and supported the progressive of the necessity prioritization ways. However, CBRank faces a lot of problems with the scalability, and the accuracy reduces while ranking the requirements. When comparing AHP and the CBRank(Anna Perini et. Al,2009), CBRank is advantageous as it yields better accuracy and appears as a good method of handling the trade-off between the decision makers.

(Anna Perini et al., 2013) researched a technique for prioritization of the requirements which is referred to as Case-based Ranking (CBRank) that combined the venture's stakeholder's choice with requirements in order to achieve ordering approximations and measured the use of the machine learning strategies. This method improved the accuracy, and it supported the adaptive elicitation process. However, effectiveness gets affected in the real complex settings.

(Anna Perini et al., 2009) presented a study that directed at evaluation of the cutting-edge tools-supported prioritization methods of requirements, such as AHP and CBRank. It mainly focussed on three major measures, consumption time, ease of use, and preciseness. Comparative outcomes proved that the comfort of use and the time consumed for the CBRank was better than for AHP while preciseness of AHP turned sophisticated than CBRank. However, the shortcoming was that it did not consider cost factor for prioritization.

(Fei Shao et al., 2016) researched a method which is used for prioritizing requirements based on the dependencies. This method developed a attributes tree that made the prioritization work easier. It used the RankBoost for the calculation of the prioritization based on the stakeholder's preferences, and finally, an algorithm based on the weighted PageRank was used that analyzed the dependencies persisting between the requirements. However, Drank supports only the contribution dependencies and business dependencies between the requirements.

(Fei Shao et al., 2016) researched a method that considered the simultaneous effect that existed between the functional, non-functional requirements and stakeholders, which was found to have different preferences using a three-order tensor with a multi-way analysis. To evaluate the method, controlled experiment was carried out. The proposed method attained good quality compared with the state-of-the-art. But it required human interference in prioritizing the preferences.

(Mohammad Dabbagh et al., 2016) conducted two experiments that aimed at evaluating the prioritizing approaches. Comparison of the IPA approach with the HAM-based approach regarding three parameters, namely time-consumption, accuracy, and ease of use proved that IPA outperformed other approaches, but there was no hierarchical relationship between the functional and the non-functional requirements.

(Saif Ur Rehman Khan et al., 2016) researched RePizer, a framework in order to prioritize software requirements. It assisted the engineer's inaccurate decision-making through the retrieval of the historical data. It provided an overview of the entire project. It improved the decision-making process, but it lacked an automatic system for selecting the requirements.

(Rubaida Easmin et al., 2014) used a technique known as the value-based intelligent requirement prioritization technique, neural network and analytical hierarchical process for improving the scalability of the method. However, it required domain knowledge that remained a disadvantage of this method.

(Rubaida Easmin et al., 2014) researched on prioritizing the requirements to deal with the remarks problem that impacted on the real rating. This approach used the stakeholder’s feedback using the binary search technique. The method arranged the requirements based on the customer requirement and decreased the stakeholder’s feedback through the reduction of the ranking difference. However, dynamism was the major shortcoming of this method.

1.3 OBJECTIVE

The objective of requirement prioritization is that it identifies the most significant requirement of the software from the list and extracts the significant knowledge of the system to impact the effectiveness.

1.4 THE PROPOSED METHODOLOGY

This thesis concentrates on software requirement prioritization that uses an efficient prioritizing method named as WhaleRank. It uses four ranking functions based on the dictionary words, a similarity matrix, the perception of the manager, and the newly updated requirements. This thesis ranking method WhaleRank overcomes the problems faced by the existing ranking methods like AHP and CBRank. These ranking functions construct four different ranks, which then combines to form a linear rank function. The linear rank function is the combination of the four rank functions that gets combined using the ranking constants. These ranking constants when summed equals to one. The ranking constants are optimally determined using the WOA. WOA determines the optimal position, and it is a population-based algorithm, and it overcomes the performance of PSO, ACO, and so on. Finally, the best rank order gets retrieved using the fitness function. This method of ranking based on WhaleRank achieves customer satisfaction by efficiently prioritizing the requirements based on the customer review. It is clear that this method contributes to enhancing above mentioned issues the quality of the product.

This thesis depicts the overview of the proposed WhaleRank method of prioritizing the software requirements. Requirement prioritization is the process carried out in the development of software projects to determine the importance of certain requirements at the stage of the release. The prioritization method ensures customer satisfaction and the resources gets utilized in an efficient way. In other words, requirement prioritization is the order in which the requirements get arranged based on their importance. This method of prioritization orders the requirements upon the successive delivery of the products that improves the quality and adds value to the product. Prioritization takes a long way that considers the expectations of the stakeholder's, risk, cost, dependencies of the requirements among each other, and so on. The existing method for performing the prioritization using CBRank ranks the requirements of the stakeholder with good accuracy but the effectiveness of the method gets affected with the complex settings, and when new requirements get added to the list, this existing method fails to update the rank for prioritizing. Moreover, the existing methods failed to address the non-monotonic conditions availing in the elicitation process and also, the ranking methods used are less effective. To overcome the abovementioned issues, this paper uses a new method named as the WhaleRank that improves the effectiveness of ranking through the usage of four ranking functions. WhaleRank not only increases the effectiveness but also it ensures that the optimization algorithm used here determines the optimum level rank to meet the consumer satisfaction and it enables an effective trade-off between the developer and the stakeholder.

1.4.1 REQUIREMENTS

The input to the WhaleRank is the documents carrying the requirements, the perception of the manager, and a number of updates. The requirements get analyzed for determining the important requirement, to consider and implement at the time of the release to ensure the customer satisfaction and the quality of the software. Documents hold the list of requirements to prioritize that are in the textual format Let us assume there are q number of requirements represented as,

$$r = \{r_1, r_2, r_3, ..., r_q\}; r_i \qquad (1)$$

Where r represents the set of all requirements, q is the total number of requirements and 1<=i<=q. It involves the set of all requirements presented in the text format in the documents. These documents possess the manager perception that holds the low, medium and higher perception. Also, it holds the number of updates in the requirements, R_i. These inputs with the requirement list, manager perception, and the number of requirement updates undergo ranking operations. The input receives ranked based totally on the significance of the requirement. Requirement with higher precedence is realized at the time of the release in order to attain client’s satisfaction.

1.5 RANKING FUNCTIONS

This thesis depicts the four ranking functions required to prioritize the requirements. The existing method used for prioritizing the requirements of the stakeholder suffers from the issue of unable to update the priorities of the newly updated requirement and it lacked automatic ranking. To overcome the above-mentioned issues of the existing methods, this paper employs four ranking functions. The ranking functions use the dictionary words, similarity measure, the perception of the manager, and the updates to prioritize the requirements. The dictionary-based ranking determines the rank based on the similar words in the requirements. Likewise, the similarity measure depends on the pair-to-pair similarity obtained using the similarity matrix, manager perception depends on the manual perception of the manager with respect to the significance of the requirement, and the rank based on requirement updates uses the number of the updated requirements to rank the requirements. Hence, the four ranking functions generate four different rank order for the requirements. These rank orders get merged to form the single linear rank function that is the sum of the product of the rank coefficients and the rank functions. This linear rank function, which is the combination of all the rank undergoes optimization search using the WOA that determines the best rank for ranking the requirements.

1.5.1 DICTIONARY-BASED RANKING FUNCTION

This ranking method uses the dictionary for ranking the requirements. The requirement gets matched with the dictionary word, and this ranking function determines the most similar requirement in the requirement list while comparing all the documents indicating that the requirement that gets higher priority is a more important requirement and it is essential to implement at the time of the release. The ranking function determines the rank based on the common requirements that exist in all the documents. The requirement that is present in most of all the documents indicates that the particular requirement ranks good priority.

$$R_1\left(r_q\right)=\frac{D\cap r_q}{|D|} \qquad (2)$$

where q is the total number of requirements presented for prioritization, D is the dictionary word used to determine the similar words, and R_1 is the rank function that computes the priority based on the dictionary words. $|\;|$ indicates the approximate number of the total dictionary words used in defining the rank to the requirements.

1.5.2 PAIR TO PAIR SIMILARITY MEASURE

The similarity measure gets computed based on the cosine similarity function. It ranks the requirement based on the similar measure through determining the frequency of the requirement. It compares the requirements and updates the value 1 in the case of any similarity between both the requirements. If there is no similarity exists between the requirements, the similarity measure is zero. Ranking depends on the preference of the requirements. In other words, when the preference of any of the requirement is high, the value of similarity gets updated as 1 else -1. Upon no similarity, the similarity measure becomes zero. Finally, the similarity measure of a requirement gets updated by taking the average of the similarity measure of the requirement along the column. Figure 2 shows the similarity matrix.

The representation of the relations that exist between two requirements also a function is $\tau(r_j, r_k)$. The similarity matrix gets filled based on the following condition,

$$\tau(r_j, r_k) = \begin{cases} -1, & when \;\; r_k \prec r_j \\ 0, & no\, order \;\; preference\, between \;\; r_j \;\; and\, r_k \\ 1 & r_j \prec r_k \end{cases} \qquad (3)$$

From the above description, $r_k < r_j$ denotes that the preference of the requirement r_k is high than the requirement r_j. The requirement pair depends on the similarity matrix, and the ranking of the most similar pair depends on the following formula. It is the average of the similarity measure of the requirement column-wise.

$$R_2(r_q) = \frac{1}{q}\sum_{i=1}^{q} P_i^{\,p} \qquad (4)$$

where q is the number of requirements on the list and P_i^p is the pair-to-pair similarity between the requirements in the similarity matrix of size $(q \times q)$. The similarity matrix gets evaluated, and the rank of the individual requirement gets computed using the average of the entire column in the similarity matrix. Ranking depends on this average value obtained column-wise. Greater the value of the similarity measure for the requirement, higher is the rank.

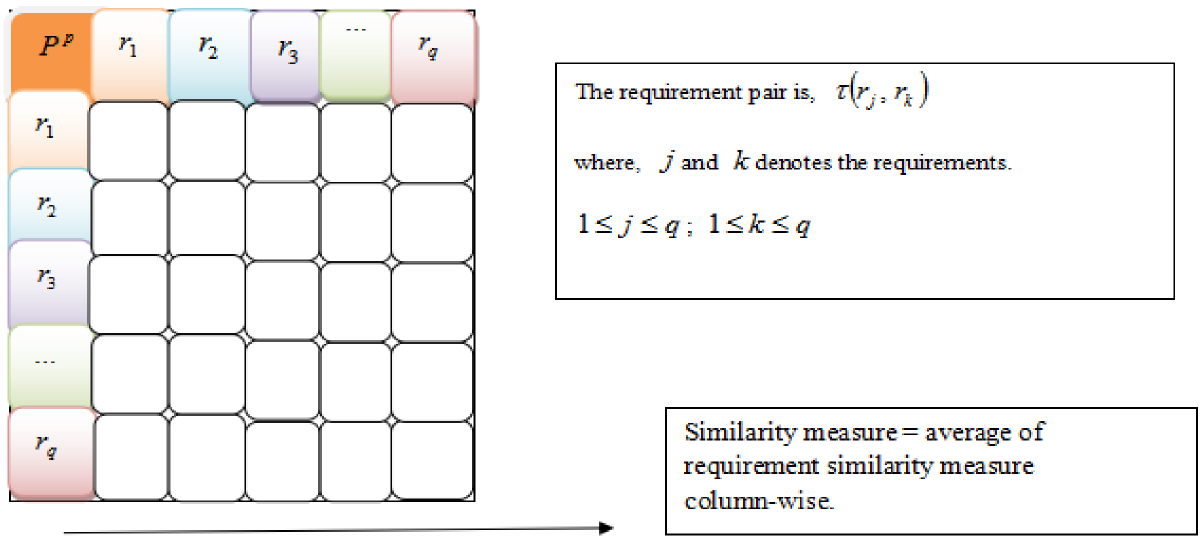

P^p
r_1
r_2
r_3
...
r_q
The requirement pair is, $\tau(r_j, r_k)$
where, j and k denotes the requirements.
$1 \le j \le q$; $1 \le k \le q$
Similarity measure = average of requirement similarity measure column-wise.

Figure 1.1 Similarity measure matrix

1.5.3 MANAGER PERCEPTION

This ranking mainly depends on the manager perception. The perception is mainly defined in high, medium, and low. It specifies the requirement based on their importance. The manager orders the requirements based on the stakeholder or customer review. The requirement with the high-level perception gets ranked first, and that requirement gains good consideration for implementation at the time of release. It considers the manager perception, which means the priority given manually based on the importance and value of the requirement.

$$R_3(r_q) = \left(\frac{M_p}{Max.\ fuzzy}\right) \qquad (5)$$

where M_p is the manager perception, r_q is the q^{th} requirement. Maximum fuzzy is nothing but the maximum value of the perception, and it retrieves the value of the high-level perception.

1.5.1 NUMBER OF UPDATES

The newly updated requirement gets considered in this framework and gets ranked using the formula through dividing the new update to the maximum updates.

$$R_4\left(r_q\right)=\frac{\left|updates\right|_q}{\max\limits_{i=1}\left(updates\right)} \qquad (6)$$

where, r_q is the q^{th} requirement. Based on these ranking functions, the requirements get ranked. Upon each update of the requirement, the importance of the newly updated requirement gets retrieved, and that newly updated requirement gets considered for implementation. The main advantage of the newly updated requirement is that it helps to enhance the features of the product and mainly, contributes in implementing new features to attain good customer satisfaction. As a result of these ranking functions, four ranking gets extracted. These four rankings altogether form a linear ranking function.

LINEAR RANK FUNCTION

The linear rank function is the combination of the four rank functions combined using the ranking constants. The ranking constants are $\alpha, \beta, \gamma, \eta$. The sum of the weights of the ranking constants is one. The linear ranking function is,

$$R = \alpha\ R_1 + \beta\ R_2 + \gamma\ R_3 + \eta\ R_4 \qquad (7)$$

where, α , β , γ , and η are the ranking constants. $R_1, R_2, R_3, and\ R_3$ are the ranking functions determined using the dictionary words, the similarity matrix, the perception of the manager and the newly updated requirements. The function based on the newly updated requirements is the major benefaction of the paper. Perception of the manager includes the preference of the customer and their idea regarding the importance of the requirement. The values of these constants get selected based on the optimization procedure. The optimization technique used to determine the best rank is the WOA that determines the value of the ranking constant $\alpha, \beta, \gamma, \eta$. The optimal weight of the ranking constants determines the best rank.

1.6 WOA

The main objective of the WOA is to compute the optimal weights of the ranking constants $\alpha, \beta, \gamma, \eta$ such that it improves the effectiveness of ranking the requirements. Effectiveness in ranking improves the quality of the product. WOA is the population-based metaheuristic algorithm that depends on the behavior of the humpback whales. WOA overcomes the issues of the existing optimization algorithms. The existing optimization algorithms like PSO, ACO provide the best solution only in their path, and some of the iterative solutions are the candidate solutions. Also, once a new population gets determined, the information of the previous iteration was lost. The advantages of WOA include: it is capable of exploring the search space globally and explores the hot areas in the search space and also, it establishes a proper balance between the exploration process and the exploitation process.

1.6.1 SOLUTION ENCODING

The main aim of the solution encoding is to determine the optimal rank for the requirements. It chooses the best values for the ranking constants for optimal rank. Here ranking constants values generated randomly. The sum of the ranking constants equals to 1. The ranking constants get determined using the search mechanism and the value that defines the rank efficiently gets considered for ranking the requirements.

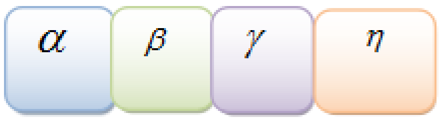


Figure 1.2 Solution encoding of the ranking coefficients

1.6.2 FLOWCHART OF WOA

This thesis presents the flowchart of the optimization method that uses the WOA optimization algorithm. WOA includes three phases like encircling, exploitation, and the exploration phases. The main objective of the WOA is about determining the weights of the ranking constants $\alpha, \beta, \gamma, \eta$ that prioritize the requirements to improve the quality of the product. This thesis discusses the steps of WOA.

1. Population initialization:

The first ten vectors(solutions) generated based on equation (7). Each vector has one $\alpha, \beta, \gamma, \eta$ (Ranking constants). Let W_n be the population of vectors and the let q be the number of vectors in the search with 1<=n<=q. Calculate the fitness of each vector and find the best vector P*(T) based on equation (17).

2. Random selection of the current best solution:

In this phase, we randomly update the value of each vector, which remains as the current best solution and the optimal solution. Once the current vector gets assigned, the other vectors update their optimal values on the optimal best solution based on equation(8).

$$\overrightarrow{O_p} = \left| \overrightarrow{A} \bullet \overrightarrow{P^*(T)} - \overrightarrow{P(T)} \right| \qquad (8)$$

O_P is the optimal vector which is the current best solution assigned. A is the search coefficient vector, P*(T) is the best current best solution of the vectors. P(T) is the vector that defines the solution.

The vector updated based on equation (9)

$$\overrightarrow{P(T+1)} = \overrightarrow{P^*(T)} - \vec{G} \bullet \vec{A} \qquad (9)$$

Where, A and G is the search coefficient vector, P*(T) is the updated solution of the vector, and

$$\vec{G} = 2\vec{s} \bullet \vec{r} - \vec{s} \qquad (10)$$

$$\overrightarrow{A} = 2 \bullet \vec{r} \qquad (11)$$

$\vec{s}$ is the coefficient vector whose value reduces linearly with increase in the number of iterations, and it takes the value 2 initially and r value between 0 and 1.

3. Vectors update based on probability:

The vector update depends upon the probability of iterations. Initially, the location of the vector gets localized. When p<0.5 and within the search space(based on equation (10)) G>1, the value of the search vectors gets updated based on equation(16). When p>0.5 the vectors are updated based on equation (12)

$$\overrightarrow{P(T+1)} = \overrightarrow{O_p^*} \bullet e^{sr} \bullet \cos(2\pi l) + \overrightarrow{P^*(T)} \qquad (12)$$

$$\overrightarrow{O_P^*} = \left| \overrightarrow{P^*(T)} - \overrightarrow{P(T)} \right| \qquad (13)$$

$$\overrightarrow{P(T+1)} = \begin{cases} \overrightarrow{P^*(T)} - \vec{G} \bullet \overrightarrow{O_p} & p < 50\% \\ \overrightarrow{O_p^*} \bullet e^{sr} \bullet \cos(2\pi l) + \overrightarrow{P^*(T)} & p > 50\% \end{cases} \qquad (14)$$

$$\overrightarrow{O_p} = \left| \vec{A} \bullet \vec{P_r} - \vec{P} \right| \qquad (15)$$

$$\overrightarrow{P(T+1)} = \vec{P_r} - \vec{G} \bullet \overrightarrow{O_P} \qquad (16)$$

Where, p is the probability value and the value of the probability is **0<=p<=1**

P-value will change based on a number of iterations.

4. Fitness evaluation:

The fitness of the rank gets determined and the best rank gets selected based on the fitness formula. For evaluating the fitness, the original training dataset is necessary. The fitness uses the original training dataset with the rank for the requirements, to determine the mean square error between the original training set and the optimized rank that holds the ordered requirement. The value of the fitness function varies between the values 0 and 1 respectively.

$$F = MSE\left(O_r, P^*(T+1)\right) \qquad (17)$$

where, F is the fitness function, MSE is the Mean Square Error, O_r is the original rank, and $P^*(T+1)$ is the determined rank using the optimization algorithm. The fitness of the derived rank gets analyzed, and the best rank gets retrieved. The rank obtained for the requirements possess efficient ranking. The requirement with the higher precedence receives carried out at the time of the discharge to attain the client satisfaction and to enhance the satisfactory of the product.

FLOWCHART OF WOA

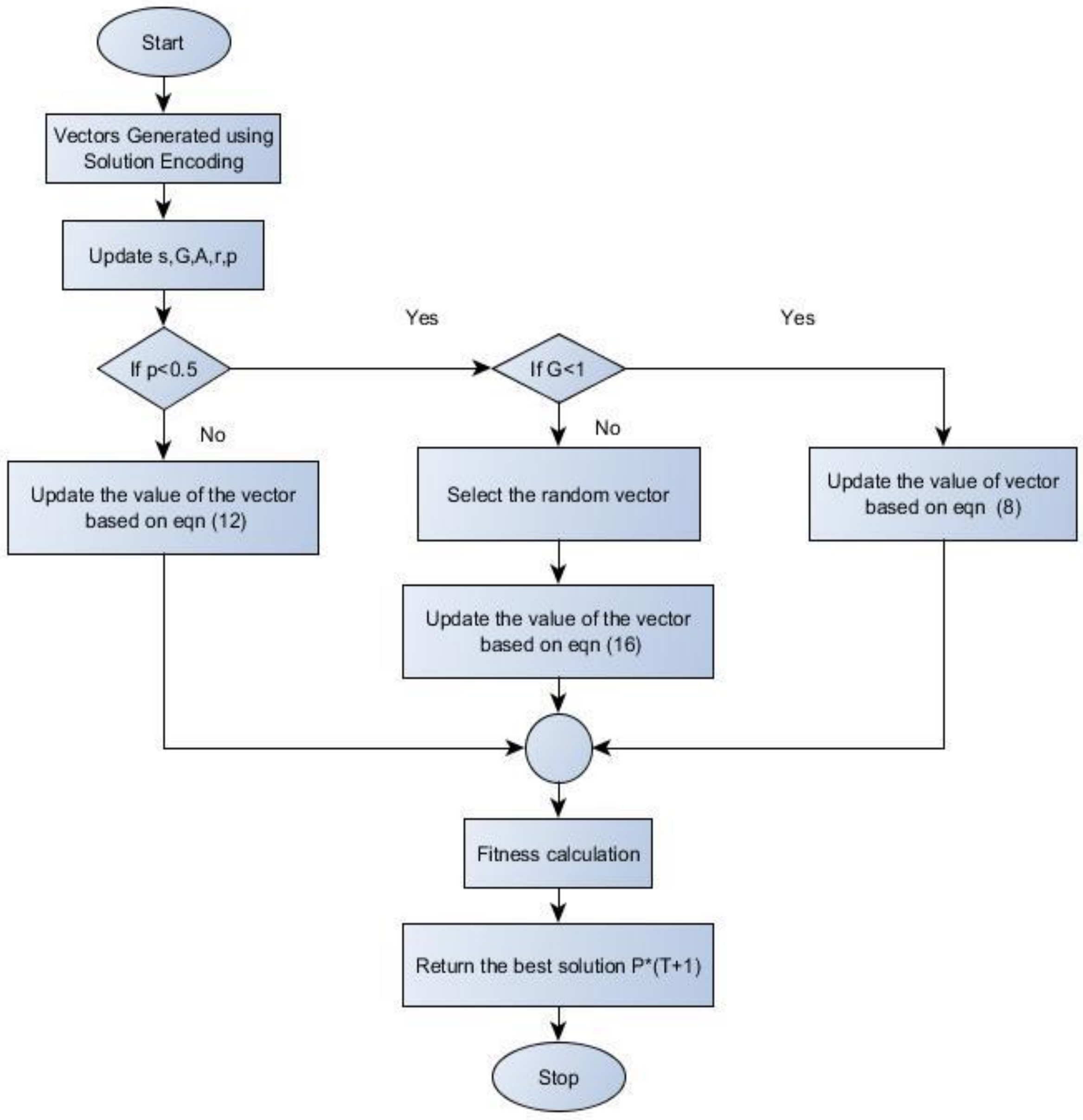

Start
Vectors Generated using Solution Encoding
Update s,G,A,r,p
If p<0.5
Yes
No
Update the value of the vector based on eqn (12)
If G<1
Yes
No
Select the random vector
Update the value of vector based on eqn (8)
Update the value of the vector based on eqn (16)
Fitness calculation
Return the best solution P*(T+1)
Stop

Figure 1.3 Flowchart of WOA

PSEUDOCODE OF WOA:

```
Initialize the 10 vectors randomly.
Calculate the fitness of each vector.
X*=the best vector
    While (t < maximum number of iterations)
        for each search vector
                Update s, G, A, r, and p
                if1 (p<0.5)
                    if2 (|A| < 1)
                        Update the value of the current vector by the Eq. (8)
                    else if2 (|A|
                        Select a random vector
                        Update the value of the current vector by the Eq. (16)
                    end if2
                else if1 (p>=0.5)
                    Update the value of the current vector by the Eq. (12)
                end if1
        end for
    Check if any vector goes beyond the another vector
    Calculate the fitness of each search agent
    Update X* if there is a better solution
    t=t+1
    end while
return X*
```

1.7 ARCHITECTURE

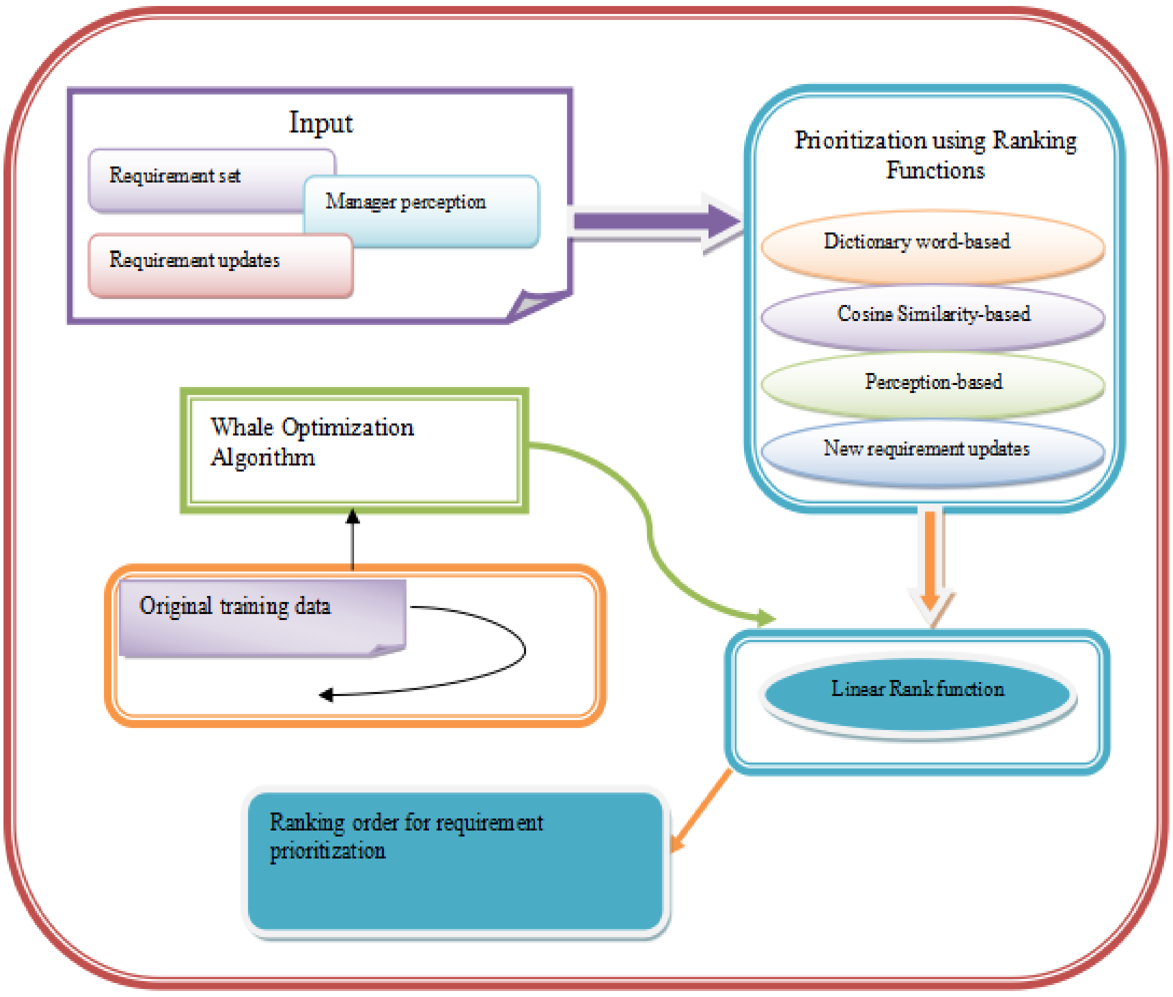


Figure 1.4 Architecture of WOA

Fig 1.4 shows the architecture of the proposed method that uses the WhaleRank method. The input to the WhaleRank is the set of documents that carries the requirements of the stakeholder, perception of the manager, and the number of updates performed. The input to the WhaleRank formulates the priority based on four ranking functions that extract the features based on the frequency of the words in the document, pair-to-pair similarity measure, M_p, and the number of updates. All these four rank functions provide four different ranking priorities, which gets merged to form a linear rank function using the ranking theory. The WOA determines the best-ranked priority using the iterative search mechanism. It includes a fitness function that gets computed using the mean square error of the original training set and the optimized requirement priority obtained using WOA.

1.8 PSEUDOCODE

```
Input: $r_i$; $1 \le i \le q$, $M_p$, $R_u$
Output: Best solution $P^*(T+1)$
Parameters: $r_q$, $R$.
Procedure
        Start
        Read the requirements, $r_i$.
            1. Determine dictionary-based rank, $R_1(r_q)$
            2. Compute the pair-to-pair similarity-based rank, $R_2(r_q)$
            3. Compute the perception-based rank, $R_3(r_q)$
            4. Compute the rank for newly updated requirements, $R_4(r_q)$
Determine the linear Rank $R$
Apply WOA to determine the values of $\alpha, \beta, \gamma, \eta$.
Evaluate the fitness, $F$; $0 \le F \le 1$.
Return the best position $P^*(T+1)$
End
```

1.9 RESULTS AND DISCUSSION

This section depicts the results and discussion of the proposed method with the comparative analysis of the existing methods. The comparison is also accuracy and Disagreement Measure (NDA).

1.9.1 EXPERIMENTAL SETUP

The ranking performance of the proposed WhaleRank gets evaluated using the experiment. Experimentation is developed in the personal computer with Intel Core i-3 processor 4GB RAM and Windows 8 operating system in the JAVA environment. The dataset used is the CM1 data.

1.9.2 EVALUATION METRICS

This section presents the metrics taken for analysis. The analysis metrics include the NDA and the accuracy.

NDA*:*

It is the measure of the disagreement measure existing between the order relations of the same requirement sets. It compares the calculated rank with the original rank order. The formula is,

$$NDA = \frac{1}{U_d} \times TDA(P,Q)$$

where NDA is the disagreement measure; $0 \le NDA \le 1$ U_d is the original rank-set. TDA is the total disagreement measure.

$$TDA(P,Q) = \sum_{j=1}^{q} \sum_{k=1}^{q} D_{P,Q}(r_j, r_k)$$

$$D_{P,Q}(r_j, r_k) = \begin{cases} 1 & if \;\; P(r_j) < P(r_k) \,\&\, Q(r_j) > Q(r_k) \\ & or \\ & P(r_j) > P(r_k) \,\&\, P(r_j) < P(r_k) \\ 0 & \\ & otherwise \end{cases}$$

where $P \; and \; Q$ are the order relations and $D_{P,Q}(r_j, r_k)$ is the disagreement between the requirement pairs. j, k are the q number requirements in the input. The disagreement measure determines the accuracy of the computed rank when compared to the target rank.

Accuracy:

It is the measure of the accuracy of the computed rank. If the order of the computed rank matches the original order, it returns the value 1, or it returns zero. Formula for accuracy is, $Accuracy = \frac{1}{q} \sum_{i=1}^{q} \sigma_i$

$$\sigma_i = \begin{cases} 1\,; & R(P_i) = R(Q_i) \\ 0\,; & otherwise \end{cases}$$

where q is the total number of requirements, $R(P_i)$ and $R(Q_i)$ are the rank of the requirement relations P and Q respectively.

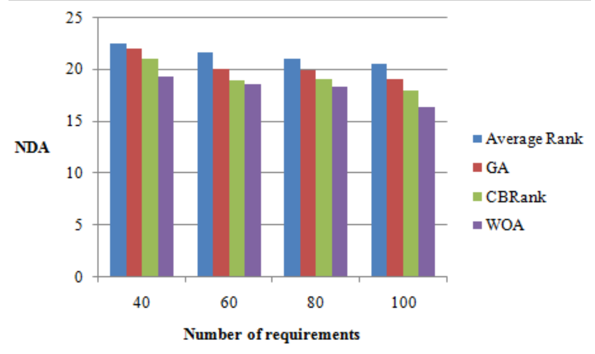
1.10 PERFORMANCE ANALYSIS

This section depicts the performance evaluation of the proposed WhaleRank also the performance metrics, and the proposed method gets compared with the existing methods to evaluate the effectiveness of the proposed method. The results of the proposed WhaleRank get compared with the existing methods like the CBRank, average rank, and the Genetic Algorithm (GA). The performance comparison is also the accuracy and the NDA to prove the effectiveness of the proposed method.

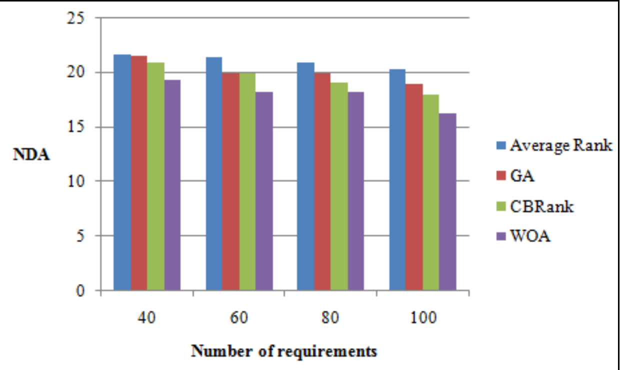
1.10.1 PERFORMANCE ANALYSIS BASED ON NDA

Figure 1.4 shows the comparative analysis of the ranking methods that depends on the varying size of the whale population. When the population size is 4, the values of NDA attained using the methods like WOA, CBRank, GA, and average rank is 20.64%, 21.06%, 21.69%, and 22.61% respectively for 40 number of requirements. For 80 requirements, the values of NDA for the abovementioned methods are 18.06%, 21.06%, 21.69%, and 22.61% respectively. It makes clear that the value of NDA is the minimum for the proposed WOA when compared with the other existing methods.

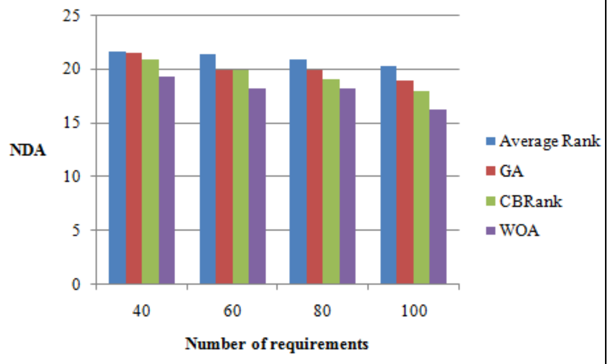
When the population size is 8, the values of NDA attained by the methods WOA, CBRank, GA, and average rank with the 60 requirements is 19.11, 20.03, 20.06, and 22.06 respectively. Similarly, when there are 100 requirements, the values of NDA for the methods WOA, CBRank, GA, and average rank are 16.39%, 18.38%, 18.63%, and 19.36% respectively. When the population size gets increased further to 16, the values of NDA attained using the methods WOA, CBRank, GA, and average rank with 60 requirements attained 19.02%, 20.01%, 19.96%, and 21.53% respectively. The performance comparison reveals that the proposed WOA attained a minimum NDA when compared with the other existing methods, which proved the effectiveness of the proposed method. Also, it is very clear that with the increase in the population size the NDA value decreases. Hence, the proposed method is the best method compared with the other methods.



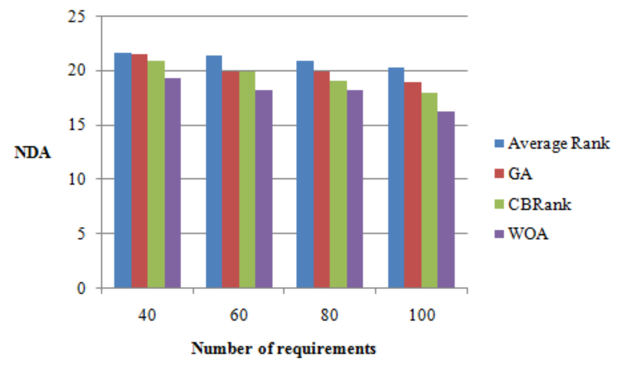
a) NDA with whale population size = 4



b) NDA with whale population size = 8



c) NDA with whale population size = 12



d) NDA with whale population size = 16

Figure 1.5: Comparative graph of NDA for varying population size

1.10.2 PERFORMANCE ANALYSIS BASED ON ACCURACY

This section depicts the performance analysis of the proposed method with the other existing methods also the accuracy. Figure 7 shows the comparative analysis of accuracy for varying population size. For analysis, the population size considered is 4, 8, 12, and 16 respectively. When the population size is 4, the accuracy attained for the methods, WOA, CBRank, GA, and average rank with the number of requirements 40 is 79.65%, 78.29%, 78.16%, and 77.34% respectively. For 100 requirements, the accuracy obtained using the abovementioned methods are 82.94%, 81.37%, 80.93%, and 80.07% respectively. This numerical interpretation makes it clear that the accuracy of the proposed method is high and with the higher number of requirements, the proposed method attains higher accuracy.

When the population size is 16, the accuracy attained using the methods WOA, CBRank, GA, and average rank is 80.03%, 78.32%, 78.35%, and 77.63% respectively for 40 requirements. For 80 requirements, the accuracy achieved using the comparative methods is 81.37%, 81.03%, 80.19%, and 80.02% respectively. It is clearly evident that the proposed method attained good accuracy compared with the other existing methods. Moreover, with the increasing number of requirements, the accuracy is found to increase. Hence, the proposed WOA possess good accuracy in ranking the requirements.

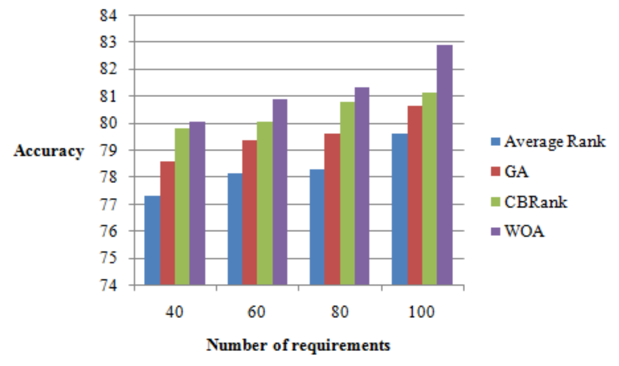
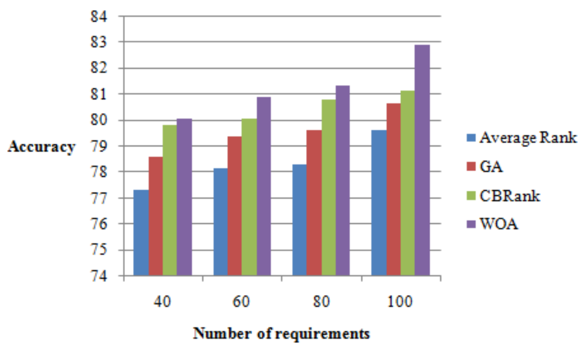
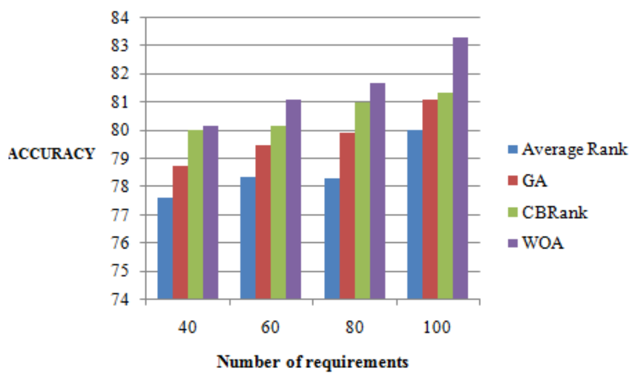
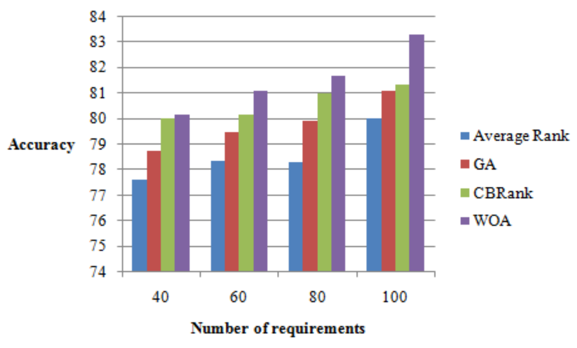
	
a) Accuracy with whale population size = 4	b) Accuracy with whale population size = 8
	
c) Accuracy with whale population size = 12	d) Accuracy with whale population size=16

Figure 1.6 Comparative graph of accuracy for varying population size

The below table 1.1 shows the comparative analysis of the ranking methods also NDA and accuracy. This method of ranking the requirements is effective in improving the quality of the product and hence, brings customer satisfaction. The proposed method outperforms the existing methods, with a minimum NDA and greater accuracy and the value of accuracy and NDA is 83.33% and 16.24% respectively.

Methods	NDA %				Accuracy%			
	n=4	n=8	n=12	n=16	n=4	n=8	n=12	n=16
WOA	**16.39**	**16.39**	**16.24**	**16.24**	**82.94**	**82.94**	**83.33**	**83.33**
CBRank (Anna Perini et al., 2013)	18.38	18.38	18.22	18.22	81.37	81.37	81.69	81.69
GA	18.63	18.63	18.31	18.31	80.93	80.93	81.11	81.11
Average Rank	19.36	19.36	19.32	19.32	80.07	80.07	80.16	80.16

Table 1.1 Comparative analysis of the ranking methods also NDA and accuracy.

1.11 SUMMARY

These results demonstrate that the accuracy can be increased using WOA algorithm. The overall comparative performance of the ranking methods WOA, CBRank, GA, and average rank. The parameters namely accuracy and NDA defines the effectiveness of the proposed method. The proposed method outperforms all other existing methods also accuracy. Analysis of the ranking methods for varying population size proves that the proposed methods possess a higher accuracy and minimum NDA compared with the other existing methods. The higher accuracy of 83.33% is attained when the population size is 16 and with 100 requirements. It is evident that with for a higher number of requirements, the proposed method achieved good accuracy. Moreover, the NDA of the proposed method is minimum, and the value of NDA is 16.24% when the population size is 16. This NDA value for the proposed method is low compared with the other existing methods.

REFERENCES

Abdel Ejnioui, Carlos E. Otero, and Abrar A. Qureshi, "Software requirement prioritization using fuzzy multi-attribute decision making," In Proceedings of the IEEE Conference on Open Systems, pp.1-6, 24 January 2013.

Anna Perini, Angelo Susi, and Paolo Avesani, "A Machine Learning Approach to Software Requirements Prioritization," IEEE Transactions On Software Engineering, vol. 39, no. 4, APRIL 2013.

Anna Perini, Filippo Ricca, and Angelo Susi, "Tool-supported requirements prioritization: Comparing the AHP and CBRank methods," Information and Software Technology, vol.51, no.6, pp.1021-1032, June 2009.

CM-1 data from http://promise.site.uottawa.ca/SERepository/datasets-page.html

Fei Shao, Rong Peng, Han Lai, and Bangchao Wang, "DRank: A semi-automated requirements prioritization method based on preferences and dependencies," Journal of Systems and Software, In Press, Corrected Proof, 26 September 2016.

Heena Ahuja; Sujata; G. N. Purohit, "Understanding requirement prioritization techniques," In Proceedings of the International Conference on Computing, Communication and Automation (ICCCA), pp.257 - 262, 16 January 2017.

João Pimentel; Maria Lencastre; Jaelson Castro, "Implicit Priorities in Adaptation Requirements," In Proceedings of the 10th International Conference on the Quality of Information and Communications Technology (QUATIC), pp.83 - 86, 16 January 2017.

Jorge Rômulo Frota Dos Santos; Adriano Bessa Albuquerque; Plácido Rogério Pinheiro, "Requirements Prioritization in Market-Driven Software: A Survey Based on Large Numbers of Stakeholders and Requirements", In Proceedings of the 10th International Conference on the Quality of Information and Communications Technology (QUATIC), pp.67 - 72, 16 January 2017.

Kavita Srivastava, Sanjay Kumar Malik, Poonam Yadav, "Discovering Semantic Web Services with Desired Quality Parameters,"National Conference on Trends in Computing, 2007.

Laura Lehtola, Marjo Kauppinen, Sari Kujala, "Requirements Prioritization Challenges in Practice," Lecture Notes in Computer Science, vol.3009, pp 497-508, 2004.

Mari Inoki, Takayuki Kitagawa, and Shinichi Honiden, "Application of requirements prioritization decision rules in software product line evolution,"In Proceedings of the 2014 IEEE 5th International Workshop on Requirements Prioritization and Communication (RePriCo), pp.1-10, 11 September 2014.

Masooma Yousuf; Mohammad Ubaidullah Bokhari; Md Zeyauddin, "An analysis of software requirements prioritization techniques: A detailed survey," In Proceedings of the 3rd International Conference on Computing for Sustainable Global Development (INDIACom), pp.3966 - 3970, 31 October 2016.

Md. Saeed Siddik and Kazi Sakib, "RDCC: An effective test case prioritization framework using software requirements, design and source code collaboration," In Proceedings of the 17th International Conference on Computer and Information Technology (ICCIT), pp. 75-80, 02 April 2015.

Mohammad Dabbagh, Sai Peck Lee, and Reza Meimandi Parizi, "Functional and non-functional requirements prioritization: empirical evaluation of IPA, AHP-based, and HAM-based approaches," Soft Computing, vol.20, no.11, pp. 4497–4520, November 2016.

Mohd. Sadiq, Jawed Ahmed, Mohammad Asim, Aslam Qureshi, and R. Suman, "More on Elicitation of Software Requirements and Prioritization Using AHP," In Proceedings of the 2010 International Conference on Data Storage and Data Engineering, pp. 230 - 234, 22 April 2010.

Muhammad Aasem, Muhammad Ramzan, and Arfan Jaffar, "Analysis and optimization of software requirements prioritization techniques," In Proceedings of the 2010 International Conference on Information and Emerging Technologies, pp. 1-6, 09 November 2010.

Muhammad Imran Babar, Masitah Ghazali, Dayang N.A. Jawawi, Siti Maryam Shamsuddin, and Noraini Ibrahim, "PHandler: An expert system for a scalable software requirements prioritization process," Knowledge-Based Systems, vol.84,pp.179–202, August 2015.

Muhammad Imran Babar, Muhammad Ramzan, and Shahbaz A. K. Ghayyur, "Challenges and future trends in software requirements prioritization," In Proceedings of the International Conference on Computer Networks and Information Technology, pp. 319 - 324, 15 September 2011.

Negin Misaghian and Homayun Motameni, "An approach for requirements prioritization based on tensor decomposition,"Requirements Engineering, pp.1-20, 07 Dec 2016.

Poonam Yadav, "Annotation of web pages using semantic tagging and ranking model to effective information retrieval," International Journal of Computer Science & Engineering Technology, vol.5, no.12, pp.1094-1098, 2014.

Rao Muzamal Liaqat, Mudassar Adeel Ahmed, Farooque Azam, Bilal Mehboob, "A Majority Voting Goal Based technique for Requirement Prioritization," In Proceedings of the 22nd International Conference on Automation and Computing (ICAC), pp.435 - 439, 24 October 2016.

Rubaida Easmin, Alim Ul Gias and Shah Mostafa Khaled, "A partial order assimilation approach for software requirements prioritization," In Proceedings of the International Conference on Informatics, Electronics & Vision (ICIEV), pp. 1-5, 10 July 2014.

Saif Ur Rehman Khan, Sai Peck Lee, Mohammad Dabbagh, Muhammad Tahir, Muzafar Khan, and Muhammad Arif, "RePizer: a framework for prioritization of software requirements," Frontiers of Information Technology & Electronic Engineering, vol.17, no.8, pp. 750–765, August 2016.

Seyedali Mirjalili, Andrew Lewis, "The Whale Optimization Algorithm," Advances in Engineering Software, vol. 95, pp. 51–67, 2016.

Soroush Forouzani, Rodina Ahmad, Sepehr Forouzani, and Nasim Gazerani, "Design of a teaching framework for software requirement prioritization," In Proceedings of the 8th International Conference on Computing Technology and Information Management (NCM and ICNIT), vol.2, pp.787 - 793, 2012.