International Journal of Applied Engineering Research ISSN 0973-4562 Volume 10, Number 14 (2015) pp 34652-34657

Requirement Prioritization In Software Release Planning Using Enriched Genetic Algortihm

[1]Sandhia Valsala, [2]Dr. Anil R Nair

[1]Research Scholar, Karpagam University, Coimbatore.
[2]Principal Scientist, Bangalore.

ABSTRACT
Release Planning plays a very important role in managing and maintaining releases and helps in the delivery of a high quality product to the end-users. It involves proper grouping of activities in the release of one or more versions of software to one or more customers.The most crucial decision is whether or not to select a requirement for implementation in the next software release. A number of software release planning models are available which considers a wide variety of factors in prioritizing the requirements that need to be included in a release. This paper proposes anEnrichedGenetic Algorithm (EGA) which is constructed using genetic operators and parents' selection in the improved population diversity. The specific factors taken for the prioritization process are requirement quality, requirements dependencies andstakeholders' priority. The aging factor of the enriched genetic model overcomes the premature convergence problem of traditional genetic algorithms where the solution selection gets trapped in the local minima.

Keywords: requirement prioritizing, software release, stakeholders, enriched genetic algorithm, convergence problem, local minima.

1. INTRODUCTION

A software release is an accumulation of new or changed featuresthat can be included in a new or updated version of a product. During release planning, the features to be included in the release are fixedin such a way that the resource, technical, budget and risk constraints are met [1]. The software release planning is a two step process namely requirement management in which requirement prioritization is a major process and software planning and management. In requirement management process, the documented requirements are changed; new requirements are added and so on whereas software planning deals with a way of handling the goals, satisfying the constraints, processing the risk factors, delivering the final product that guarantees user and customer satisfaction on time[2]. In case of any fault in between the development process, the design needs to be reengineered that leads to high expenditure [3].

Software requirement management is a process of supervising the requirements that needs to be included in the product software so as to fix the issues. This process should be done before adding the requirements to the software product [4]. Identifying the different degrees of priority is the most crucialtask of requirement prioritization. It needs a better understanding between delivering the requirements that are not used and not delivering the requirements that are needed.The requirement prioritization after taking into consideration the different constraints prioritizes the requirements in order (i.e.) which requirement should execute first. Various approaches are available for this prioritization, each of which is implemented in different situation and different environment. So it is necessary to choose a technique for prioritization based on the developing environment. Selecting the high priority requirements and avoiding the low priority requirements minimizes the cost and durationof the project [5].

Prioritization overcomes the issues that ariseduring complex decision making. The software quality is evaluated by its performance that satisfiesend user and product customers. Therefore obtaining the initial requirements, choosing the correct priorities and perfect planningof release schedule leads to a successful product delivery. Hence the paper proposes anEnriched Genetic Algorithm (EGA) in which the genetic operators and the initial population diversity are enhanced in a better way,such that the premature convergence problemwhere the solution selection gets trapped in the local minima is resolved. In addition, the dynamic population mechanism, EM algorithm and the aging factor is introduced through which the dynamic changes in the requirements does not affect the final software performance, minimizing the work load of genetic algorithm and finally overcoming the premature convergence problem over the diverse requirements. The implementation results are tabularized and the proposed system is assessed by comparing the proposed system methods with the existing methods.

2. RELATED LITERATURE

The important issue faced by the software industries todayis to identify the time to terminate the testing process and to fix the time of delivery of the software.The author proposes a Test Point Analysis based Module Priority approach [6] that process and calculate the time to terminate the testing and to deliver the software product. The testing phase should have proper time scheduling and this should be reduced or extended according to the faults found in the testing phase and that will influence the success of the software. Testing phase with insufficient time period may result in undetected bugs that affect the software quality. Usually the software testing phase takes much cost and repeated testing may result in higher expenditure [7]. Therefore, it is essential to prioritize the

testing components in order to reach an optimal testing phase with in the time constraints.

The software development cycle and the software release planning should possess the decision making quality that decides which requirements should be included in the next version of the release and the time schedule for the next version release. Various algorithms and methods have been suggested for requirement prioritization and scheduling so as to meet the release time. These methods may or may not take human guidance to produce the desired outcome [8]. The author [9] suggests a new SPM decision making model that solves Constraint Satisfaction Problem (CSP). The relations used in this model resembles the functionalities namely resource constraints, stakeholder preferences and feature priorities. Irrespective of the utilized algorithms and methods, the constraint solver explores the solution space without human intervention. The author also discusses the pros and cons of SPM model for solving CSP issue with respect to requirement prioritization and planning of release.

The purpose of prioritizing the test cases leads to proper scheduling of test suits that minimizes the effort on testing experimentally. In order to enhance the existing schemes, the author proposes clustering process in prioritization [10] and this process minimizes the fault percentage and increases the fault detection percentage. The proposed approach aims to prioritize the test cases in order to achievean efficient and client desired test suite that proves the reliability of clustering approach.

The major problem faced during software product developmentis the endless list of requirements, coming from the stake holders. Therefore it is a very difficult task to select the most efficient requirements from this list. The Binary Priority List (BPL) technique, one of the best binary searches based techniques, does prioritization of large number of requirementsefficiently [11]. But still this technique is not explained and utilized in many situations. Hence the paper makes deep analyses of this BPL technique and utilizes this technique for prioritizingrequirements and compares its efficiency with other techniques. Many small scale software companies are using a facilitating tool which is not much cost effective to prioritize the requirements and handles low-level requirements.

Requirements prioritization is ordering the requirements according to its importance. Therefore, based on the prioritization result a decision can be made on which requirement to implementand which not.Although a wide variety of techniques are available to do requirement prioritization, it's important to understand also that each technique has its own pros and cons. The paper proposes a cutting edge method and examines its extensions based on the requirements[12]. The author also examines a number of existing methods and technologies andproposes a framework for prioritization process.

There are a number of techniques available for Requirement prioritization and choosing the most efficient technique becomes a critical issue. The paper investigates six requirement prioritization techniques , tests them experimentally and compares them based on their easiness of handling, scalability, number of comparisons each technique takes to make the final decision, total time taken and accuracy [13]. The experimental results show that Value oriented Prioritization technique is the one most suitable for prioritizing the requirements.

3. MEASURES ADDED IN THE ENRICHED GENETIC ALGORITHM

a) Dynamic Population

The list of requirements that can be added to a software release can often change even after the process of requirement selection has started. This dynamic change may affect the outcome of software release. Hence the proposed enriched genetic algorithm introduces a threshold value for the fitness value assigned to all the requirements [14]. Usually the requirements taken for the selection process are below the criterion level, but still a genetic algorithm is in need for choosing those requirements for reproduction in the next generations. So the enriched genetic algorithm fixes a threshold value, discards the requirements which are having the fitness value less than the fixed threshold value. The fitness value of the requirement is computed as follows.

Initially the effect of initially taken requirementis computed as

$$Effect = \sum_{i=1}^{n} compeer * weight \quad (1)$$

Where the effect of the requirements are computed by using the weights of the requirements and comparing the taken requirements with the original requirement set.If the both are matched, then the value of compeer is "1". If not, then the value is "0".

The delta value is computed using the calculated effect value and the priority level assigned by the stake holders(i.e.)

$$\Delta = |Effect - stakeholders\ priority\ level\ based\ on\ the\ requirement| \quad (2)$$

The requirements quality of each requirement is termed as rating. It is computed by forming a matrix where the rows and columns of the matrix are $requirements\ X\ requirements$. Consider the number of requirements initially taken from the stakeholder is"n". Then the matrix formulation is as shown in figure 1.

Requirements / Requirements	$Req\,1$	$Req\,2$	...	$Req\,n$
$Req\,1$	$(SP1+SL1)-(SP1+SL1)$	$(SP1+SL1)-(SP2+SL2)$	...	$(SP1+SL1)-(SPn+SLn)$
$Req\,2$	$(SP2+SL2)-(SP1+SL1)$	$(SP2+SL2)-(SP2+SL2)$	...	$(SP2+SL2)-(SPn+SLn)$
⋮	⋮	⋮	⋮	⋮
$Req\,n$	$(SPn+SLn)-(SP1+SL1)$	$(SPn+SLn)-(SP2+SL2)$	...	$(SPn+SLn)-(SPn+SLn)$

Figure 1 Matrix formation to calculate the requirement quality

The summation of stakeholders' preference SP_i of the requirement and the significance level SL_i of the particular requirement is calculated for each requirement individually. The difference between the requirements based on the above matrix is computed. Then the requirement quality of a particular requirement is calculated by computing the average of each requirement row wise. Since this requirement quality is termed as rating here, the penalty value is calculated as follows

$$penality = \frac{\Delta * rating}{100} \quad (3)$$

Since the summation of the penalty and the fitness value of a requirement equal "1", the fitness value can be written as

$$fitness = 1 - penalty \quad (4)$$

This process of discarding the requirements may lead to a situation where the chosen number of requirements will be greater than the number of requirements in the initial stage. If this issue occurs, then from the list of chosen requirements the one with less fitness will be discarded. Therefore the computation efficiency is increased and the computational complexity is decreased.

b) Estimation-Maximization (EM) algorithm

Since the requirements specified by the stakeholders may be huge in number, it is necessary to determine an efficient way to reduce the number of requirements. Thus, the Estimation-Maximization (EM) algorithm is used for this purpose. This algorithm is spitted into two traditional steps namely E-step (Expectation) and M-step (Maximization). The expectation process estimates the likelihood of the requirements while comparing them with all the other requirements. Then the maximization process finds the requirements with high likelihood (similarity).The different stakeholders present their requirements in different way and some of the requirements may have similar outcome. So among those requirements with similar outcome, one sample requirement is taken.This process reduces thenumber of requirements in an efficient way and supports the proposed Enriched Genetic Algorithm (EGA) to be performed with reduced number of requirements.

c) Aging Factor

The next parameter that is added to the traditional genetic algorithm is aging factor. The purpose of introducing this parameter is to overcome the premature convergence problem. Usually the requirements that are evaluated using the above threshold process will have been generated in the previous or before generations. So, including these requirements generated in the previous generations may become the reason for premature convergence problem where the diversity of population is decreased that leads to minimizing the searching process in a local minima. Hence the aging factor parameter is introduced in the Enriched Genetic Algorithm to overcome this premature convergence problem. The aging factor discards the older requirements even though they have good fitness value and allows the newer requirements to be included even though they have bad fitness value.

4. ENRICHED GENETIC REQUIREMENT PRIORITIZATION PROCESS

Requirement Prioritization has a most significant place in software development lifecycle and these requirements should be chosen taking into consideration the time constraints, technical constraints, available budget, and most importantly stakeholders' expectations. Genetic algorithm is mostly suggested to find the optimal features from among the extracted features. The requirements are referred to as features here. But the requirements in a generation are processed under three operatorsand the comparison is made among the requirements in the same generation. In addition, the traditional genetic algorithm has premature convergence problem. Hence the enriched genetic algorithm is introduced here in order to overcome this premature convergence problem which arises when the highly elite requirements are identified in a closed space, restricting it to the local minima [15]. This is possible by incorporating the aging factor process which restricts the requirements which are not satisfying the age limit to take part in the reproduction of next generation. In addition, the static size of the requirements is also another problem where the requirements may be included after the process has started. Therefore the enriched genetic algorithm has adopted the dynamic population mechanism to overcome the static size population problem.

As depicted in the figure 2, the requirements for the next version of the software or upcoming software release are initially taken from the stakeholders of the software. It is essential to order the initiated requirements according to their priority. So the Enriched Genetic Algorithm (EGA) is proposed in order to prioritize the requirements and some additional parameters are added for the purpose of making the prioritization process more efficient in terms of computation, robustness and reliability.In the existing system of prioritizing

the requirements [16], 4 requirements were taken for processing from which the significant requirements were produced as output for the software development process.

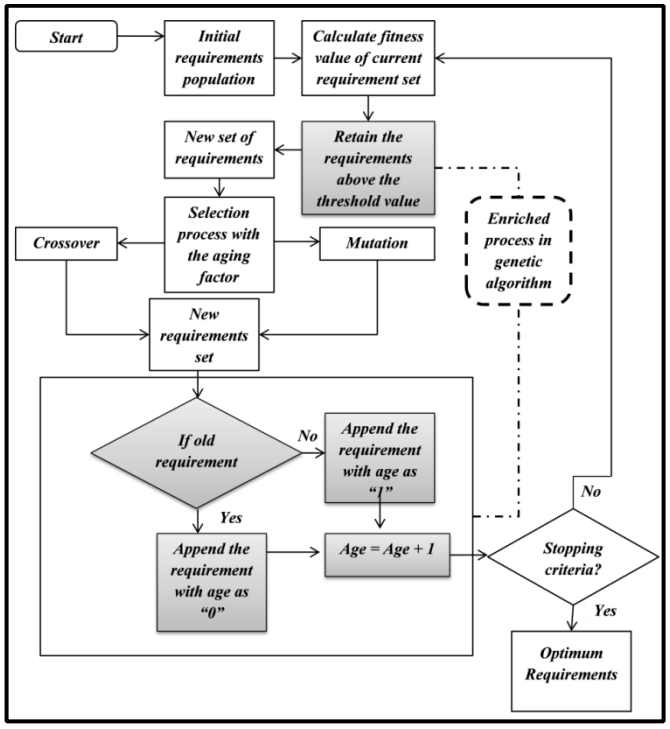

Figure 2. Enriched genetic requirement selection process

Hence, in the proposed system of requirement prioritization the requirements initially are taken from the stakeholders, and then the fitness value of each requirementis calculated separately. Since the requirements are large in number, the EM algorithm selectively suggest the optimized requirements to the genetic algorithm and the process is carried out using the Enriched Genetic Algorithm (EGA). The process of EnrichedGenetic Algorithmincluding the aging factor, EM algorithm and dynamic population as depicted in the figure 2 is briefly explained below.

Step 1: Initiallythe requirements from the stakeholders are takenand the process is initiated.

Step 2: The fitness value of each requirements are computed by using the equation (4).

Step 3: The EM algorithm is applied to customize the similar and likelihood requirements into single requirement.

Step 4: The threshold value is fixed through which all the requirements from the EM algorithm compares their fitness value with the threshold value. This fitness value is calculated based on the selection factors. Here the selection factors are stakeholders' priority, requirements dependencies and requirement quality. Through these selection factors, the revenue is increased; the development cost and the development risk is minimized.

The threshold value is calculated by using the formula

$$threshold\ value = fitness\ average(1 - fitness\ average) \quad (5)$$

First, find the average fitness value among all the requirements and apply in the formula of the above threshold equation (5).

The following pseudo code is for the selecting the requirements which are good in its fitness value.

$Find\ threshold\ value$
$For\ each\ requirement\ (i)$
$If\ threshold\ value\ >\ fitness\ value(i)$
$then\ discard\ the\ requirement(i)$
$else$
$allow\ the\ requirement(i) to\ proceed\ to\ next\ step$
$end\ for$

The threshold value using the fitness value is found. Then the each requirement compares its fitness value with the threshold value; remove the lowest fitness value requirements and preserve the highest fitness value requirements for the next process.

Step 5: The preserved requirements are considered as new requirements set and is processed under the usual genetic operators such as selection, crossover and mutation.

The requirements having higher fitness value are prompted to reproduce the next generation. The next generations of requirements are generated using the process of crossover and the mutation. First the crossover is performed in order to mate two requirements to create the new offspring requirement. Mutation attempts to improve the requirement with dynamic elitism that gives the efficient requirement.

Step 5: A new set of requirements are generated as next generation by using the crossover and mutation operations of the genetic algorithm after the selection process.

Step 6: The continuous process of crossover and mutation restrict the system to local minima and this cause the premature converge problem. So the aging factor concept in inserted after the crossover and mutation process. This process keeps the initial generation requirements in its memory and checks whether the requirements from the newly generated set is same as the initially generated one.

$If\ requirement\ (i)\ =\ requirement\ (i)\ in\ memory$
$then\ requirement\ age\ = 0$
$else$
$requirement\ age = 1$

Step 7: Checks whether the stopping criteria is met (i.e.) limit of number of requirements or fitness value of the requirements attained. If the criteria aremet, then those requirements are the final optimal requirements. The proposed system identifies eight optimal requirements.

Step 8: Then these eight optimal requirements are prioritized based on the fitness values computed for each requirement and the high risk priorities.

5. RESULTS AND DISCUSSION

A successful software release planning has the initial stage of requirements prioritization (optimal requirement selection) and then scheduling these requirements to be released on time. This paper analyses the process of requirement prioritization and proposes a genetic algorithm along with the integration of aging factor, EM algorithm and the dynamic population scheme and hence named as enriched genetic algorithm. The

stake holders' value for each requirement are calculated and shown in the table 1.

Table 1 Stakeholder Values for Requirement

Requirements	***Stakeholder Values***
Authorization on order cancellation and removal (AO-C/R)	0.43625
Authorization on archiving service orders (A-SOA)	1.3075
Performance improvements on order processing (PIOP)	2.18
Inclusion graphical plan board (GPB)	3.05125
Link with Acrobat reader for .pdf files (LPDF)	3.92375
Adaptations in rental systems (ARS)	2.18
Symbol import (SI)	0.43625
Comparison of services per department (SCPD)	0.87125

The prioritized requirements along with its value are shown in the table 2.

Table 2 Requirement quality values for requirement

Requirements	***Quality Values***
Authorization on order cancellation and removal (AO-C/R)	0.37
Symbol import (SI)	0.37
Comparison of services per department (SCPD)	0.74
Authorization on archiving service orders (A-SOA)	1.11
Adaptations in rental systems (ARS)	1.85
Performance improvements on order processing (PIOP)	1.85
Inclusion graphical plan board (GPB)	2.59
Link with Acrobat reader for .pdf files (LPDF)	3.33

The proposed enriched genetic algorithm is compared with the existing traditional genetic algorithm and the particle swarm optimization. The figure 3 depicts that the enriched genetic algorithm processes the given eight requirements in a minimized time when compared with the existing traditional genetic algorithm [17] and the particle swarm optimization [18].

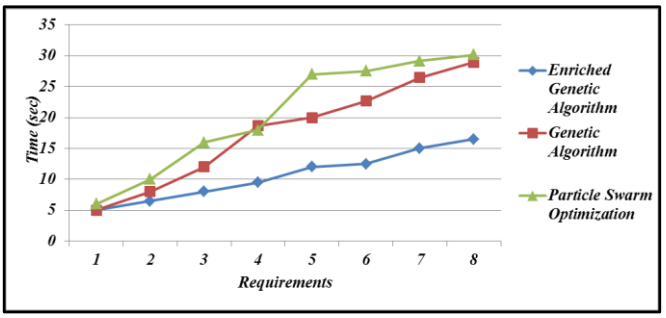

Figure 3 Comparison of proposed and existing algorithms in terms of time

The proposed enriched genetic algorithm is compared with the existing traditional genetic algorithm and the particle swarm optimization in terms of efficiency. The figure 4 shows that the proposed system proves better efficiency than the existing algorithm in the feature selection process.

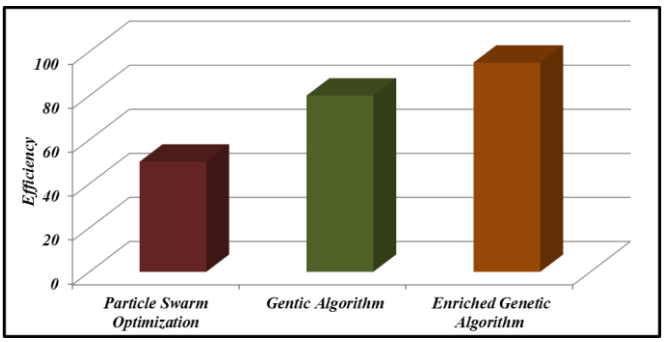

Figure 4 Comparison of proposed and existing algorithms in terms of efficiency

Even though many approaches including the proposed system are used for the purpose of selecting the optimal requirements, these optimal requirements are not always accurate all the time.The requirement prioritization may produce some average delay in the final software release. But the proposed enriched genetic algorithm used for the prioritization purpose minimizes the average delay considerably and is shown in the figure 5. All features attain less than 0.75 sec time delay.

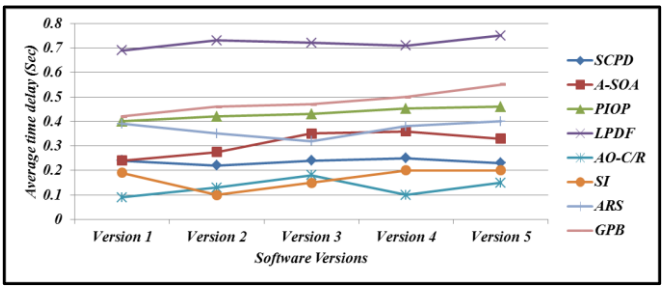

Figure 5 Average delay measures of some of the optimal requirements

6. CONCLUSION

In this paper, the requirement prioritization processin software release planning is explained and also proposed is an enriched genetic algorithm that adds somefunctionalityto the traditional genetic algorithm in order to overcome the drawbacks of the genetic algorithm. The aging factor removes the issue of premature convergence problem and the dynamic population eliminates the computational complexity and enhances the computational efficiency. Finally the optimal requirements with better accuracy are generated. From the implementation results, the final optimal requirements are tabulated with the corresponding stakeholders' value and the quality values.The proposed enriched genetic algorithm is compared with the existing algorithms for the same purpose and proves that it shows better result than the other algorithms.

REFERENCES

[1] SandhiaValsala and Anil R, "Review and Analysis of Software Release Planning Models", International Journal of Engineering and Advanced Technology, Volume 3, Issue 5, pp. 1-7, ISSN: 2249 – 8958, June 2014.

[2] Muhammad Naeem Ahmed Khan and Muhammad Khalid, "Review of Requirements Management Issues inSoftware Development", I.J.Modern Education and Computer Science, Volume 1, pp. 21-27, 2013.

[3] Meenakahi.T, "Reengineering a Software Process by using Unified Foundation", International Journal of Inventions in Computer Science and Engineering, Volume 1 Issue 3, ISSN (Print): 2348 – 3431, 2014.

[4] DhirendraPandey and VandanaPandey, "Importance of Requirement Management: A Requirement Engineering Concern", International Journal of Research and Development - A Management Review, Volume 1, Issue 1, ISSN: 2319–5479, 2012.

[5] TamjidRahman and M. Rokonuzzaman, "A Noble Methodology for Users' Work Process Driven Software Requirements for Smart Handheld Devices", International Journal of Software Engineering & Applications, Volume 5, Issue 4, pp. 21-38, July 2014.

[6] Praveen RanjanSrivastava, SubrahmanyanSankaran and PushkarPandey, "Optimal Software Release Policy Approach Using Test Point Analysis and Module Prioritization", MIS Review, an International Journal, Volume 18, Issue 2, ISSN: 1018-1393, pp. 19-50, March 2013.

[7] BjörnRegnell and Krzysztof Kuchcinski, "Exploring Software Product Management Decision Problems with Constraint Solving – Opportunities for Prioritization and Release Planning", International Workshop on Software Product Management, IEEE, ISBN: 4577-1146, pp. 47-56, 2011.

[8] Srividhya. J and K. Alagarsamy, "A Novel Method for Reduction of Regression Testing Cost", International Journal of Inventions in Computer Science and Engineering, Volume 1 Issue 3, ISSN (Print): 2348 – 3431, 2014.

[9] Sunil Yadav, "Efficient operating system scheduling for symmetric multi-core architectures in CPU scheduling", International Journal of Innovative Computer Science & Engineering, Volume 1 Issue 2, ISSN: 2393-8528, pp. 24-27, 2014.

[10] Arvind Kumar Upadhyay and A. K. Misra, "Prioritizing Test Suites Using Clustering Approach in Software Testing", International Journal of Soft Computing and Engineering, Volume 2, Issue 4, ISSN: 2231-2307, September 2012.

[11] Thomas Bebensee, Inge van de Weerd and SjaakBrinkkemper, "Binary Priority List for Prioritizing Software Requirements", International Working Conference on REFSQ, Springer Berlin Heidelberg, Volume 6182, pp. 67-78, July 2010.

[12] Aasem.M, Ramzan.M and Jaffar.A, "Analysis and optimization of software requirements prioritization techniques"International Conference on Information and Emerging Technologies, IEEE, ISBN: 4244-8001, pp. 1-6, June 2010.

[13] ManjuKhari and Nikunj Kumar, "Comparison of Six Prioritization Techniques for Software Requirements", Journal of Global Research in Computer Science, Volume 4, No. 1, ISSN: 2229-371X, pp. 38-43, January 2013.

[14] Zhanshan (Sam) Ma and Axel Krings, "Dynamic Populations in Genetic Algorithms", 23rd Annual ACM Symposium on Applied Computing, March 16-20, 2008.

[15] BalajiSridharan, "Modifications in Genetic Algorithm using additionalparameters to make them computationally efficient", IEEE International Advanced Computing Conference, pp. 55-59, ISBN: 4244-4791, 2010.

[16] Chen Li, Marjan van den Akker, Sjaak Brinkkemper and Guido Diepen, "An integrated approach for requirement selection and scheduling in software release planning", Journal of requirements engineering,Springer-Verlag, Volume 15, Issue 4, pp. 375-369, ISSN: 0947-3602, 2010.

[17] Ruchika Malhotra and Abhishek Bharadwaj, "Test Case Prioritization Using Genetic Algorithm", International Journal of Computer Science and Informatics, Volume 2, Issue 3, pp. 63-66, ISSN: 2231 –5292, 2012.

[18] Luciano S. de Souza, Pericles B. C. de Miranda, Ricardo B. C. Prudencio and Flavia de A. Barros, "A Multi-Objective Particle Swarm Optimization for Test Case Selection Based on Functional Requirements Coverage and Execution Effort", IEEE international conference on tools with Artificial intelligence, IEEE, pp. 245-252,ISSN : 1082-3409, 2011.