# Software Requirement Prioritization based on Non-Functional Requirements

Umang Garg
Department of CSE, ASET
Amity University, Uttar Pradesh
Noida, India
umanggarg76@gmail.com

Abhishek Singhal
Department of CSE, ASET
Amity University, Uttar Pradesh
Noida, India
asinghal1@amity.edu

***Abstract*— Requirements can be characterized as needs or demands. In software, requirements are a portrayal of what a system is ought to do. Software requirements describe what must be produced and delivered [1]. System may have few to many requirements. In case of shortage of time or budget prioritizing requirements become a necessity. It not only helps the developer comprehend, which requirements are essential, important or urgent but also proves to be useful in decision making in every phase of software development lifecycle. A lot of techniques have emerged over time to prioritize requirements, each with a set of strengths and weaknesses. In this paper, we present an approach by combining a few of existing techniques that will help the software engineers prioritize requirements.**

***Keywords- Requirements Prioritization; Functional Requirement; Non-functional Requirement; AHP; Algorithm***

## I. INTRODUCTION

Requirements prioritization is a way towards dealing with the relative significance and urgency of different requirements to adapt to the limited resources of projects. Adequate prioritization guarantees that the most critical requirements are tended to immediately in case time or budget runs out [1]. In case the development team is running short of budget or time a decision needs to be made as to which subset of requirements should be supplied. This decision needs to be taken by the stakeholders.

Prioritization is a method of dealing with allocating limited resources to maximize profit. When the expectations of customer runs higher than what can be achieved in timeline and resource availability, one needs to make sure that they are able to maximize profit in that limited time and resources by including the most important functions in the system. When it comes to prioritizing requirements or choosing a subset of requirements we needs to make sure developers and customers work together as developers don't generally know what is most vital for the customers, and at the same time customers do not know cost and time required to achieve a particular requirement.

Before the process of requirement prioritization, users must be made aware of the reason for software prioritization and the repercussions of not going through with the process. We need to do this because it gets difficult and confusing for customers to rate their requirements and it gets even more difficult when we have multiple stakeholders and a common consensus needs to be drawn. Everybody naturally have their own advantages at heart, so they don't feel like continually bargaining their requirements for someone else. Nonetheless, making this decision is customer's responsibility.

The objective of prioritization is to help in making decisions. Prioritization is utilized in almost every phase of software development lifecycle i.e., requirements elicitation, design and development of system, testing, implementation, and post implementation activities [1]. Prioritization of requirement not only helps in selecting a subset of them in situations of limited time and resources, but also helps developers understand their sensitivity. Most important requirement not only needs to be achieved but also needs a lot of attention while development and testing. It will evaluate as a high impact risk due to its importance for customer.

In its most straightforward form, there would be no need to prioritize objects if there was only one of them. But in any case with just a few options, it gets hard to take a decision in view of more than one choosing criterion. If there should be an occurrence of having tens, hundreds or even thousands of options, making a decision becomes more troublesome and difficult. Many prioritization techniques are present to help with this situation.

An approach for prioritizing requirement takes into account a few criterion which are likewise called key elements. Some of these key elements are penalty, importance, cost, risk, time, volatility and many other elements along with a combinations of these have been discussed in [2].

This paper is divided into 5 sections. The upcoming section gives a small review of some of the prioritization

978-1-5090-3519-9/17/$31.00 ©2017 IEEE

techniques. We propose an approach based on the existing techniques and explain its methodology in section III. We also provide an example to show the working of the algorithm proposed in the next section and finally, we present our results in the 4th section and conclude our work in the last section.

## II. LITERATURE REVIEW

There are various techniques available for prioritization of software requirements. Next, we have provided a small review of some of these techniques.

### A. ANALYTICAL HIERARCHY PROCESS

The Analytical Hierarchy Process (AHP), a technique put forward by Thomas Saaty [3] is a statistical procedure for complex multi criterion decision making issues. It is utilized for the prioritization of requirements based on various criteria [2], like cost, penalty, importance, risk, and time. In this technique, we compare all the candidate requirements with each other in pairwise manner so as to determine the degree to which a requirement is more vital than the other. This degree of importance should be allotted according to Table 1. [4]. At each hierarchy level this process makes (n*(n-1)) / 2 comparisons for n number of requirements. The calculation of these comparisons becomes more and more difficult as the candidate requirement's number increases.

***Table 1. Basic scale for pairwise comparisons in AHP***

| SIGNIFICANCE | DESCRIPTION |
|---|---|
| 1 | Equal significance |
| 3 | A little difference in significance |
| 5 | Crucial difference in significance |
| 7 | Extensive difference in significance |
| 9 | Enormous difference in significance |
| 2,4,6,8 | Intermediate values between |
| Reciprocals | When comparing a requirement j with requirement i, it is assigned the reciprocal of the value assigned to requirement i when it was compared with requirement j |

The areas where AHP has been effectively implemented [5] include: resource allocation; selection of one option from many, forecasting, quality function deployment, business process re-engineering, balanced scorecard and total quality management.

AHP incorporates a consistency check [6]. It is a fault tolerant technique. The priorities obtained in the end are relative and in view of a proportion scale, which permits useful evaluation of requirements [7]. This technique provides a flexible, effective and powerful means in the regard of achieving both qualitative and quantitative aspect of a decision.

### B. COST-VALUE APPROACH

Joachim Karlsson et al [8] created this technique to aide people responsible for decision making to prioritize the requirements in the light of the ratio of cost of implementation to value [9]. Stakeholders are curious about the 'value' aspect which represents the profit achieved from receiving the candidate requirements. Whereas 'cost' on the other hand refers to the cost of implementing that software requirement in terms of money and time. This approach utilizes the AHP technique in order to prioritize requirement by computing a value-cost ratio of all requirements in pairwise manner and represent the result in a graph [10].

With a specific end goal to examine competitor prerequisites, this approach utilizes AHP method to figure an esteem cost proportion and exhibits the outcome in a chart as contribution to discharge choice [10]. In straightforward terms, it can basically be said that this approach depends on AHP and its two dimensional qualities give a simple intend to plot them on cost-esteem diagram however fails to assess interdependencies between prerequisite

### C. NUMERICAL ASSIGNMENTS (GROUPING)

This strategy is also proposed [11] in RFC 2119 and IEEE Std. 830-1998. In this approach we group the requirements in different groups depending upon the ordinal scale to which the stakeholders can correlate to for confident classification. Mostly 3 groups are classified namely standard, optional, and critical [7].

We begin by classifying requirements into different groups and handing them over to each stakeholder. The interested party then, assigns each requirement in all groups with a number on a scale of 1 to 5 based on their evaluation. Then we calculate the final priorities based on the average of all the rankings given by the stakeholders to all the requirements.

We combined the above techniques to build a new approach in order to exploit all the advantages offered by each technique.

## III. PROPOSED APPROACH

In this paper we put forward an approach consolidating the models explained above and prioritizing functional requirements based upon non-functional requirements. We first assign importance value to each non-functional requirement using pair-wise comparison method in order to obtain a relative importance of all non-functional requirements based on user's perspective. Then assign an importance value to each requirement in respect to these non-functional requirement much like numerical assignment

according to table 2. He we don't want a relative result, we are only concerned with getting an ordinal scale values for each functional requirement based on non-functional requirement.

**Table 2. Scale for Importance of functional requirement based on non-functional requirement**

| IMPORTANCE | DESCRIPTION |
|---|---|
| 1 | Identical significance |
| 2 | A little difference in significance |
| 3 | Crucial difference in significance |
| 4 | Extensive difference in significance |
| 5 | Enormous difference in significance |

We then calculate priority value of functional requirements by multiplying their importance value with the importance value attached to the respective non-functional requirement. Based on these priority value along with the cost and time associated with each of the requirement we can select the requirements to be implemented by presenting them on a graph and marking the deadlines.

STEPS IN OUR APPROACH

1. Assign importance value to each non-functional requirement based on pairwise method.
2. Assign importance value to each functional requirement based on the non-functional requirement.
3. Calculate priorities by matrix multiplication.

```
ALGORITHM
i,j k: int
Q(1..a,2): cell //having two attributes with Q{x}(1) //giving quality metric x's name and Q{x}(2) giving //importance of x quality metric
R(1..d,1..a+6): cell //having a+6 attributes with
//R{x}(1) giving requirement x's name
//R{x}(2),R{x}(3),..R{x}(a+1) giving importance value //of requirement x on basis of quality metric Q{1}, Q{2} //and so on till Q{a} where this value
//can vary in btw 1 to 5.
// R{x}(a+2) priority value
//R{x}(a+3) contains time needed to implement //requirement x.
//R{x}(a+4) contains cost to implement //requirement x.
// R{x}(a+5) cumulative time
// R{x}(a+6) cumulative cost
Cost: int //cost cutoff
Time: int //time cutoff
Begin
// arrange Quality metric cell array in decreasing order of //their importance
for i=1 to a
for j=i+1 to a
        if(Q{i}(2)<Q{j}(2))
        interchange their values
end for
//calculate priority value of each requirement

for i=1 to d
for j=1 to a
        R{i}(a+2)=R{i}(a+2) + Q{j}(2)*R{i}(j+1)
end for
end for
//arranging requirements in decreasing order of their //priority
k=1
for i=1 to d
for j=i+1 to d
        if(R{i}(a+2)<R{j}(a+2))
                interchange their values
        else if(R{i}(a+2)==R{j}(a+2)){
        while((R{i}(k+1)==R{j}(k+1))&&(k!=a))
                k=k+1
        if((R{i}(k+1)<R{j}(k+1))&&(k!=a+1))
        interchange values
        }
end for
end for
//calculating cumulative cost and time
R{1}(a+5)= R{1}(a+3)
R{1}(a+6)= R{1}(a+4)
for i=2 to d
        R{i}(a+5)= R{i}(a+3)+ R{i-1}(a+5)
        R{i}(a+6)= R{i}(a+4)+ R{i-1}(a+6);
end for
for i=1 to d
        if((R{i}(a+5)>Time)||(R{i}(a+6)>Cost)
                display i
        exit
End
```

## IV. APPLICATION EXAMPLE

Let us take a few requirements of an article publishing software. Out of the many non-functional requirement we consider here the following:-

1. Security
2. Data integrity
3. Usability

Now, assign t heir importance value using pair-wise comparison.

**Table 3. Pair-wise comparisons of non-functional requirements**

| | SECURITY | DATA INTEGRITY | USABILITY |
|---|---|---|---|
| SECURITY | 1 | 3 | 9 |
| DATA INTEGRITY | 1/3 | 1 | 7 |

| | | | |
|---|---|---|---|
| **USABILITY** | 1/9 | 1/7 | 1 |
| **TOTAL** | 13/9 | 23/7 | 17 |

**Table 4. Q(1..a,2) CELL**

| | SECURITY | DATA INTEGRITY | USABILITY | PRIORITY | TIME | COST | CUMULATIVE TIME | CUMULATIVE COST |
|---|---|---|---|---|---|---|---|---|
| **REQUIREMENT 1** | 5 | 3 | 1 | 13.73 | 1 | 10 | 1 | 10 |
| **REQUIREMENT 2** | 1 | 5 | 2 | 7.43 | 2 | 5 | 3 | 15 |
| **REQUIREMENT 3** | 1 | 1 | 5 | 3.99 | 5 | 20 | 8 | 35 |
| **REQUIREMENT 4** | 1 | 1 | 3 | 3.63 | 3 | 30 | 11 | 62 |

**Table 5. Priority value of non-functional requirements**

| | SECURITY | DATA INTEGRITY | USABILITY | PRIORITY |
|---|---|---|---|---|
| **SECURITY** | 9/13 | 21/23 | 9/17 | 2.14 |
| **DATA INTEGRITY** | 3/13 | 7/23 | 7/17 | 0.95 |
| **USABILITY** | 1/13 | 1/23 | 1/17 | 0.18 |

**Table 6. Q(1..a,2) CELL**

| | |
|---|---|
| **SECURITY** | 2.14 |
| **DATA INTEGRITY** | 0.95 |
| **USABILITY** | 0.18 |

Now we list the functional requirement we consider here

1. Authentication or implementing login id and password
2. Read mode, Read and Write mode
3. Simple interface
4. Managing cookies i.e. to provide suggestions

Now, we assign their importance value with respect to the non- functional requirements and also mention the time and cost to implement each requirement along with cost and time cut off

Let us suppose the cut off time is 9 days and cut off cost is 65 units

**Table 7. Requirement Table**

| | Security | Data Integrity | Usability | Time | Cost |
|---|---|---|---|---|---|
| **Requirement 1** | 5 | 3 1 | | 1 | 10 |
| **Requirement 2** | 1 | 5 2 | | 2 | 5 |
| **Requirement 3** | 1 | 1 | 5 | 5 | 20 |
| **Requirement 4** | 1 | 1 | 3 | 3 | 30 |

Now, we calculate their priority value as well as cumulative time and cost. We observe from table 4 that based on the cut off cost all the requirements can be executed but based on cut off time only first 3 requirement can be implemented.

## V. RESULTS AND DISCUSSION

We also timed our algorithm, it took 39.757 seconds to run including self time. On excluding the self time the code 0.111 seconds to run as measured by the matlab profiler. One makes (m*(m-1))/2 comparisons in order to find out the priority of 'm' non-functional requirments much like AHP and in order to find out priority of 'n' functional requirements we make n*m calculations The total no. of calculations necessary to achieve priority values of 'n' functional requirements based on 'm' non-functional requirements is equal to (m(m-1))/2 + m*n calculations. This value is considerably small when compared with AHP in case where no. of functional requirements is high and comparatively less than non-functional requirements

## VI. CONCLUSION

We designed an approach to relate functional and non-functional requirement and prioritize functional requirements on the basis of non-functional requirements.Hence, based on this the developer can take the decision as to what to implement first and what to implement second and so on based upon their priority and also pay more attention on requirements with high importance value in respect to security and other non-functional requirements.

## REFERENCES


[1] M. Aasem, M. Ramzan, and A. Jaffar, "Analysis and Optimization of Software Requirements Prioritization Techniques," International Conference on Information and Emerging Technologies (ICIET), June 2010, pp. 1-6.

[2] J. Karlsson, "Software Requirements Prioritizing," IEEE Proceedings of the Second International Conference on Requirements Engineering, April 1996, pp. 110-116.

[3] R. Beg, Q. Abbas, and R. P. Verma. "An Approach for Requirement Prioritization using B -tree," First International Conference on Emerging Trends in Engineering and Technology, IEEE, July 2008, pp. 1216-1221.

[4] V. Ahl, "An Experimental Comparison of Five Prioritization Methods: Investigating Ease of Use, Accuracy and Scalability," 2005.

[5] E. H. Forman, and S. I. Gass, "The Analytic Hierarchy Process-An Exposition," Operations Research 49.4, 2001, pp. 469-486.

[6] J. Karlsson, C. Wohlin, and B. Regnell. "An Evaluation of Methods for Prioritizing Software Requirements," Information and Software Technology 39.14 (1998), pp. 939-947.

[7] K. A. Khan, "A Systematic Review of Software Requirements Prioritization," Diss. Blekinge Institute of Technology, 2006.

[8] J. Karlsson, and K. Ryan. "A Cost-Value Approach for Prioritizing Requirements," IEEE Software 14.5, 1997, pp. 67-74.

[9] L. Karlsson, "Requirements Prioritisation and Retrospective Analysis for Release Planning Process Improvement," Diss. Lund University, 2006.

[10] P. Berander, "Evolving Prioritization for Soft ware Product Management," 2007.

[11] A.Aurum, C. Wohlin, "Engineering and Managing Software Requirements," Springer, July 2005.

[12] www.cse.msu.edu/~chengb/RE-491/Papers/**SRS**Example-webapp.**doc.**