Requirement Reprioritization: A Multilayered Dynamic Approach

Varun Gupta¹, D. S. Chauhan², Kamlesh Dutta³ and Chetna Gupta⁴

¹Research Scholar, Uttarakhund Technical University, Dehradun
²Vice Chancellor, Uttarakhund Technical University, Dehradun
³Associate Professor & Head, National Institute of Technology, Hamirpur
⁴Assistant Professor, JIIT, Noida

¹varun13_cse@yahoo.co.in, ²pdschauhan@gmail.com, ³kd@nith.ac.in,
⁴chetnagupta04@gmail.com

Abstract

The task of requirement reprioritization is an important activity initiated due to the dynamism in the developing & the working environment of the software. Reprioritization is focused on the re-allotment of priorities to both implemented as well as non implemented ones. Focusing the implemented ones will ensure quality of regression testing and stressing on non implemented ones will ensure to consider the delayed as well as new or changed requirements. The reprioritization is complex decision making and effortful activity. Thus this task will require some heuristic parameters that could lead to the minimization of reprioritization efforts without any negative fabrication of quality of the current version. This paper proposed a multilayered dynamic approach for requirement reprioritization that is well suited for both agile and non agile development methodologies employing any prioritization method. The proposal is illustrated with the help of live case study of Library Management Software System developed in the Computer Programming laboratory of National Institute of Technology, Hamirpur.

Keywords: Reprioritization, density, static and dynamic priorities

1. Introduction

The requirement prioritization is an essential and a success determining activity of incremental software delivery. Requirement prioritization involves the allotment of priorities to different stakeholder’s requirements. The decision related to selection of a subset of requirements from big set for current release, is not an easy task since the decision had to be made by making a perfect balance between different and perhaps conflicting prioritization aspects by considering the diversity of involved stakeholders, for every requirement. This may involve an iterative prioritization process involving the series of negotiations, priority value allotments, backtracking and finally final priority calculation. The final ranked set of requirements ordered in accordance with allotted priorities represents the implementation orderings as agreed by diverse stakeholders against different prioritization aspects.

This process of requirement prioritization continues in second increment, where new emerged requirements are prioritized and a decision is made between the delayed requirements of first increment and new prioritized requirements on the basis of allotted priorities. This paper terms this requirement prioritization process as “Static” because this prioritization process is not flexible enough to get adjusted in accordance with dynamism in a software environment. Dynamism could come in various forms like enhanced customer knowledge, change in business rules, competitor’s activities, misunderstandings on the part of
developers etc. For example suppose that an organization had delayed the implementation of the requirement that provides security at various levels due to limited resources available. During second increment it was decided to implement this requirement but suppose that the competitor had implemented the authentication using “Sign-in Seal” like in yahoo mail then textual password security at various levels could be removed from implementation set. This is because that “Sign-in Seal” security would be higher and more reliable security that could be implemented. Static prioritization would thus be not the need of hour in today's dynamic software environment. Also in earlier paper (Gupta et al., 2012) reported that by focusing on the requirement of highest priority one could reduce the regression testing effort. Thus during regression testing of higher increment, considering those whose priorities are still highest in a dynamic environment, would improve the quality of the delivered increment.

Dynamic requirement prioritization involves adapting the requirement prioritization process in ever changing environment. Under dynamic requirement prioritization, the priority of requirements is never fixed, In fact it varies. Thus during development of new increment, there are changes in values to be allotted against some of the decision aspects for some requirements thereby changing the assigned priorities. It is quite important that these parameters that need to be reassigned values be properly analyzed and allotted values. New requirement gets prioritized using the similar prioritization method as used in old increment but this time allotted values will be more accurate than what it might have got if prioritized in old increment. Dynamic prioritization involves reprioritization of old prioritized requirements and priorities of new ones.

Dynamism in an environment where software is developed, employed and used, leads to changes in decision aspects priorities and may change the stakeholder’s view points about some requirements. Considering such dynamism would make the task of requirement prioritization a true reflection of the viewpoints of its intended stakeholders since the requirements will be prioritized against newly prioritized decision aspects. Already implemented requirements priorities needs to be modified since requirement priorities plays an important role during regression testing. This would make regression testing a dynamic activity. Dynamism leads to change in rankings of decision aspects and change in viewpoints of some stakeholder groups.

Since requirement priorities are dynamic rather than static, this paper proposed a novel method for incorporating dynamism in the requirement prioritization process. Heuristics makes it possible for lower the efforts involved in reprioritization of already prioritized requirements. This technique is well suited for requirement prioritization techniques based on any measurement scale like nominal, ordinal, interval and ratio scales.

2. Related Work

(Z. Racheva et al., 2008) presented various prioritization approaches as employed in agile methodologies. The authors classified these approaches into two categories i.e., those comparing requirements pairwise and those grouping requirements on the basis of importance. A conceptual model from the client’s perspective is then presented followed by issues and solution strategies related to requirement re-prioritization. The paper was focused on two main research goals i.e., to highlight the main factors and identify the problems in the process of reprioritization of requirements.

(Z. Racheva et al., 2010) presented the findings acquired after the review of requirement prioritization by considering the different information sources like journals, books etc, in the form of two conceptual models. First model (Say model A) describes the requirement prioritization process in agile from the client's perspective while the second model (Say model B) is the detailed description of the above model. It
basically describes the various factors that are considered by the clients while making prioritization decisions at an inner iteration time i.e., during requirement re-prioritization. The outcome of the analysis of the collected information was the conceptual reprioritization model (Model A & B) that was based on the perspective of the clients rather than that of developers. This had provided the future research path concerning re-prioritizations.

(Z. Racheva et. al., 2010) presented the findings acquired by conducting the interviews with 11 practitioners belonging to 8 agile methodologies based software developing firms in the form of another conceptual model. The practitioners include Project Managers, Developers, Product Owner, Client and Scrum Master. The results were analyzed for total 10 projects. The conceptual model illustrates that at the time of reprioritization, client’s perspectives included the five main factors i.e. Business Value, Effort Estimation/ Size Measurement, Learning Experience, Input from the developers and External Change. It was also found that only three decision aspects are considered by the clients during reprioritization namely business values, negative value and risks. The Project constraints like delivery time, costs etc and requirement dependencies are given due consideration during the process of reprioritization. The authors reported the results in the form of conceptual model illustrating the reprioritization in inter iteration time for agile projects only.

(Z. Bakalova, et. al., 2011) presents the gaps as exist between the current reprioritization practices as revealed by survey of literature focussing on prioritization techniques and those as described by practitioners from agile methodologies bases software developing firms. The authors analyzed total 22 prioritization techniques employed in agile software developments and the concepts presented in the conceptual model, a model developed through interviews with total 11 practitioners belonging to 8 agile software firms drawing experience of 10 projects. The practitioners include Project Managers, Developers, Product Owner, Client and Scrum Master. In order to capture the gaps, the authors analyzed 22 prioritization techniques to capture the use of the concepts as presented in the model. This was termed as mapping in the paper by the authors. The results indicate that there are the huge gaps between the current practices as described in the literature and those practiced in industries. Existing requirements prioritization techniques are at coarse grained descriptions since they only present the upper levels of the prioritization techniques without focussing on all aspects that drives the prioritization process. Very few are in fact based on client’s perspectives.

The main finding acquired after the study of these existing work in the area of requirement reprioritization, is that it simply describes the model of the reprioritization by considering the various factors that come to light during reprioritization. Thus the methods stressing on reprioritization are missing from the literature. This means following things are missing from the existing works:

- Reprioritization refers to the task of re-allotment of prioritization values of only those requirements that are still to be implemented. Thus priorities of the requirements that are already implemented are assumed static. Thus the impact of requirement prioritization on regression testing as identified by (Gupta et al., 2012a) is ignored.

- Reprioritization, an inner iteration time activity, is discussed in terms of process model from client prospective for agile methodologies. On the basis of case study, it was concluded that the decision aspects like business values, negative values and risks would be employed during prioritization/reprioritization. This
leads to ignorance to reprioritization of decision aspects. Reprioritization of decision aspects is identified as one of the important activities in (Gupta et al., 2012b, 2012c). Also, reprioritization case studies neglect the iterative developments of non-agile software’s.

- Reprioritization is considered from the perspective of clients only thereby ignoring the developers. Various non-functional developers’ requirements might have frequent changes in their priorities.

- Since reprioritization would subject to a bigger set of requirements to prioritization, the amount of efforts involved would be greater. The reprioritizations methods that stresses on new methods of reprioritization that would reduce the amount of effort involved without sacrificing the criteria for high quality.

Not all the software development firms have made the transition from traditional software development practices to agile ones. Also requirement prioritization effectiveness is dependent on effective prioritization and selection of decision aspects. This means that the model of requirement reprioritization should not address the issue of prioritization efforts but also focus on both agile & non-agile methodologies and larger set of decision aspects. Keeping all above points in mind, this paper proposes a new method of requirement re-prioritization that reprioritize all the requirements i.e., requirements that are already implemented and requirements that are yet to be implemented. Since this method is dealing with a larger set of requirements, minimization of prioritization efforts is of prime interest.

3. Dynamic Reprioritization Model

This involves analyzing the changed parameters, re-allotment of values, and recalculation of priority values. Thus the new requirement would be allotted values by utilizing the changed decision parameters while problems persists with the reprioritization of already implemented and delayed ones since it involves huge prioritization efforts. One way of reducing the efforts involved with reprioritization is to take into consideration some of the heuristic parameters that are to be allotted values. These parameters are then used to alter the priorities of already prioritized requirements. The multilayered model of requirement reprioritization is given in Figure 1:

**Figure 1: Requirement Reprioritization Multilayered Model**

```
<table>
<thead>
<tr>
<th>Analyze Environment Dynamism</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameters Values Re-Alotted</td>
</tr>
<tr>
<td>Priority Alteration</td>
</tr>
<tr>
<td>Implementation Set Selection</td>
</tr>
</tbody>
</table>
```
Each of the phases of requirement reprioritization is discussed below:

1. **Analyze Environment Dynamism:** In this step, whenever new increment development is initiated, efforts are made to analyze the changes that are made right from the delivery of old increment along with its type, nature and impact. For example with the passage of time customer satisfaction parameter will definitely change along with changes in few others like competitor advantage, dependencies etc.

2. **Parameters values re-allotted:** In this step the parameters that newly added or founded to be changed like customer satisfaction would undergo through re-allotment of values by diverse stakeholders. Some heuristics would be required in this step since involving of diverse stakeholders to prioritize already prioritized requirements would be effortful activity.

3. **Priority Alteration:** In this step the final priorities are computed using the new assignment values and taking into consideration changes in decision aspects or parameters.

4. **Implementation set selection:** In this step decision is made regarding the implementation order of requirements by selecting the delayed and newly emerged ones in accordance with new assigned priorities. If already implemented ones had got less priority this time, then it will not make any difference since new features will be built on existing features. The only difference will be during the regression testing of new increment.

This model highlights in brief the main stages involved during the reprioritization of requirements.

### 4. Proposed Reprioritization Approach: Mathematical Formulation

The proposed approach for reprioritization of requirements involves considering the three cases as given below:

a) **[For New Requirement]:** The new requirements are prioritized against the similar decision aspects as used in earlier increments. However it might be possible that some decision aspect be dropped as a process of reprioritization, then simply the old ones minus the dropped one is used. For example, some technological limitations might exist during old increments but for current one there might not be any such limitation. Hence requirements that were ignored only due to technological limitations might be preferred this time.

b) **[For Already Implemented Requirements]:** Since a change in priorities of already implemented requirements would affect regression testing phase, thus careful analysis and allotment of values to should be carried out in a controlled manner. As an experience with the project on “Library Management” implemented in a Computer programming laboratory of National institute of Technology, Hamirpur, it could be easily reported that the parameter that involves re-allotment of values is “Customer Satisfaction”. This happened because the stakeholders of this project gets well equipped with the necessary knowledge of the software after they starts using its first increment and hence leads to exploration of new requirements and changes in existing ones. It might be possible that some of the requirements might score very poor this time.

Let us consider that $R_{new}$ is a new requirement priority, $R'_{p}$ is the new requirement priority obtained after re-allotment of values, $R_{p}$ is old priority of
requirement and \( N \) be the adjustment factor that considers the dynamism that is not related to decision aspects. This adjustment factor is obtained by combining the values allotted for (a) Number of times used by stakeholders (\( U \)) (b) Number of change requests (\( CR \)) (c) Competitive edge (\( C \)) (d) Dependency (\( D \)). So a requirement will be of highest priority if it is used maximum number of times, there are large change request associated with it, there is a huge competitive edge and had large dependencies with other requirements. Although last two are decision aspects but still keeping in view their importance, it’s better to keep them mandatory and separate.

This adjustment factor is given by Equation 1:

\[
N = U + CR + C + D. \tag{1}
\]

New priority is given by Equation 2:

\[
R_{\text{new}} = R'_{p} + R_{p} + N. \tag{2}
\]

c) [Delayed Requirements]: The problem that persists with reprioritization of delayed ones is related to employing diverse stakeholders that could be a very effortful activity. Thus one may be interested in accurate reprioritization by the involvement of few stakeholders. Only those aspects will be considered that are analyzed to have been changed with the passage of time. In order to achieve this, the measurement scale as under in analytical Hierarchical Process (AHP) by (Satty, 1980) could be categorized into three categories i.e., Minimal Effect (1 to 3), Moderate Effect (4 to 5) and High Effect (Above 5 to 9). The changed aspects and values allotted during the earlier increment are analyzed for reprioritization purpose. This paper proposed a term called average density that is defined as the number of values given for each category divided by the number of stakeholders. Thus the average density is defined for each category of scale used. The idea is that if average density is higher in minimal effect then chances are too low for change of values to high effect. This means that there may be minute changes to priority, and hence could be ignored. Similarly high density in high effect also would not have any effect since low values if translated into high ones would change priority to small amount. Only in the situation where the average density is almost similar for each case, would change the requirement priority by big amount. In this case, values are re-allotted and priority is recomputed.

5. Case Study

Consider the case study of “Library Management Software System” developed in the Computer Programming laboratory of National Institute of Technology, Hamirpur, as an illustrative example. This software was delivered one increment and hence this software can be considered only as an illustrative example rather than validation and evaluation means. The various requirements for this software are as follows:

R1: Software must be able to do Book management.
   
   R1.1: Software must be able to enter new book information.
   
   R1.2: Software must be able to delete existing book information.
   
   R1.3: Software must be able to modify existing book information.
   
   R1.4: Software must be able to display book information.
R2: Software must be able to do Member management.
   R2.1: Software must be able to add new members.
   R2.2: Software must be able to delete existing member.
   R2.3: Software must be able to view existing member details.
R3: Software must be able to Issue book.
R4: Software must be able to return books.
R5: Software must be able to search book in the book information database.
R6: Software must be able to provide password security.
R7: Software must be able to provide voice password security.
R8: Software must be able to work using touch screen facility.
R9: Software must be able to store log information for security diagnosis.
R10: Software must be able to inform the administrator about consecutive three unsuccessful failure logins.
R11: The software must provide GUI.
R12: Software must be able to store passwords in encrypted forms.
R13: Software must be able to generate responses very quickly.

Students implemented all the requirements except those from R7 to R13. The reason for the neglect of the other requirements is attributable to less familiarity with development technology, huge resources to be invested, lesser familiarity with these advanced features. The features i.e., from R1 to R7 were successfully implemented and thus software was delivered in first increment only.

Now let’s suppose that second increment has to be delivered by implementing the remaining functionality. Now consider the following scenarios:

- No new change request has emerged in the meantime.
- No new requirement has emerged.
- Values of heuristic parameters i.e. parameters of adjustment factors are collected.
- The average density is computed for each category.

There were 6 stakeholders of this group and for requirements numbered R1 to R4 had got maximum values in high effect category, R5 had got maximum values in the moderate category while R7 in the highest category. Requirement R12 had got maximum values in the minimal effect category due to less knowledge about encryption. Now it might be possible that environment encounter changes in the following forms: (1) Competitor implementing biometric security using “Sign-in Seal”. (2) Security breaches in working environment of stakeholders (3) enhanced working knowledge of the software.

Now the priorities allotted to each requirement will be differed by a large amount and low priority requirements would get highest priority after above cited changes. R7, R9 and R12 will now get highest priority after changes mentioned at sr. no 1 and 2.
Thus the delayed requirements will be reprioritized by calculation of average density. As mentioned earlier that due to less familiarity and resource constraints, maximum density lies in minimal effect and moderate effect category. Thus it’s better to reprioritize these requirements and regenerate the requirement rankings.

The priority of already implemented requirements will be altered using Equation 1 and 2. In case change requests and emerging of new requirements for implementation, these are prioritized by using the existing prioritization method employing the changes set of decision aspects.

6. Conclusion and Future Work

This paper proposes the multilayered dynamic prioritization method that could perform the task of reprioritization and prioritization of software requirements in accordance with a change in environment. This means that the static prioritization method is being adapted for use in the actual dynamic environment. This method is not evaluated on live systems and thus its validation and evaluation on live case study is kept as a future work. Also this method of dynamic prioritization is best suited for non pairwise comparison prioritization methods, hence it is further expected that this technique will be extended so as to work with pairwise comparison technique. This technique is valid for both agile and non agile process models.

References

Authors

Varun Gupta is a research scholar at the Uttarakhand Technical University, Dehradun. He is teaching at Jawaharlal Nehru Govt. Engineering College, Sundernagar and earlier worked with National Institute of Technology, Hamirpur and Indian Institute of Technology, Mandi. He holds a Masters of Technology and Bachelor of Technology in Computer Science & Engineering. His area of interest is Software Engineering (Requirement Engineering, Requirement prioritization, Global Software Development, Software Testing etc). The author has authored numerous research papers in peer reviewed, referred International Journals and various International and National Conferences that were held in India and Abroad.

Durg Singh Chauhan received a Ph.D. degree from Indian Institute of Technology, Delhi in1986, and did his postdoctoral work at Goddard space Flight Centre, Greenbelf Maryland, USA (1988-91). He is currently working as Vice Chancellor, Uttarakhand Technical University, Dehradun. Dr. Chauhan also served as Vice Chancellor of three other Indian universities. He had been nominated chairman/member of various committees of national importance. He got the best Engineer honor of the institution of Engineer in 2001 at Lucknow. He has authored two books and published and presented 150 research papers in international journals and international conferences. He is Member of prestigious organizations like IEEE, ACM, SIAM and Fellow of Institution of Engineers.

Kamlesh Dutta is Associate Professor and Head in the Department of Computer Science & Engineering at the National Institute of Technology, Hamirpur (Himachal Pradesh) INDIA. She obtained her doctorate from Guru Gobind Singh Indraprastha University, Delhi, INDIA and her master’s degree in Computer Science & Engineering from Indian Institute of Technology, Delhi, INDIA. Her major research interests include Software Engineering, Artificial Intelligence and Natural Language Processing. She has published more than 80 research papers in national and International Journals and Conferences.

Chetna Gupta is Assistant Professor at Jaypee Institute of Information Technology, India. She obtained her doctorate in the area of Software Testing. She also holds a Masters of Technology and a Bachelor of Engineering degree in Computer Science and Engineering. Her areas of interest are Software Engineering, Requirement Engineering, Software Testing, Software Project Management, Data Structures and Web Applications. She has many publications in international journals and conferences to her credit.