Perspectives in Science (2016) 8, 79—82

Available online at www.sciencedirect.com

ScienceDirect

journal homepage: www.elsevier.com/pisc

Experimentation on accuracy of non functional requirement prioritization approaches for different complexity projects☆

Raj Kumar Chopra[a], Varun Gupta[b,*], Durg Singh Chauhan[c]

[a] Uttarakhand Technical University, Dehradun, Uttarakhand, India
[b] Amity University, Noida, Uttar Pradesh, India
[c] GLA University, Mathura, Uttar Pradesh, India

Received 19 February 2016; accepted 6 April 2016
Available online 20 April 2016

KEYWORDS
Non functional requirements;
Software project complexity;
Requirement prioritization

Summary Non functional requirements must be selected for implementation together with functional requirements to enhance the success of software projects. Three approaches exist for performing the prioritization of non functional requirements using the suitable prioritization technique. This paper performs experimentation on three different complexity versions of the industrial software project using cost-value prioritization technique employing three approaches. Experimentation is conducted to analyze the accuracy of individual approaches and the variation of accuracy with the complexity of the software project. The results indicate that selecting non functional requirements separately, but in accordance with functionality has higher accuracy amongst the other two approaches. Further, likewise other approaches, it witnesses the decrease in accuracy with increase in software complexity but the decrease is minimal.

Introduction

Requirement prioritization is an activity to perform the selection of requirements, the task that is challenging due to the involvement of many stakeholders with potentially conflicting view points, multiple requirements to be handled and large effort to be invested in this activity. The wrong requirement selection not only results in wasteful effort and potentially increased effort of the next release, but also possesses the risk of project failures.

The software comprises functional and non functional requirements that together determine the acceptability of it within the market. The users never demand the non functional requirements, but appreciate if they are

☆ This article belongs to the special issue on Engineering and Material Sciences.

* Corresponding author.
E-mail address: varun13_cse@yahoo.co.in (V. Gupta).

http://dx.doi.org/10.1016/j.pisc.2016.04.001

implemented. The potential reason could be that non functional requirements determines the success of functional aspects of the system and are usually unheard amongst its users. User invests few of non functional requirements after the software is put to use and other requirements as he interacts with competitor products. For example, a mobile app with a good interface, but with slow speed will not feel appealing to the users and they may ask for fast applications. The software could have many non functional requirements that determine the success of the developed application.

Resource constraints with potentially being time and cost limitation put an end to the idea of implementation of all non functional requirements and hence accuracy and effort optimized prioritization is undertaken. However, the prioritization of non functional requirements is challenging due to several reasons:

- Non functional requirements are prioritized by developers and not by users. It is important that the selection of such requirements must be aligned to the selection of functional requirements.
- Non Functional requirements are always considered as the overhead as they do not provide any functional aspect to the system. Hence, investing huge effort in their selection and implementation is considered as only overhead effort for overall development. In other words, negligible resources are allocated for non functional requirements.
- Non functional requirements shall never be prioritized with respect to functional requirements, as competitive requirements. If this happens, non functional requirements are guaranteed to get lower priority than functional requirements.
- Non functional requirements can be prioritized individually, i.e. Not in competition to functional requirements, but however their selection needs to be balanced with the selected functional requirements.

Non functional requirement prioritization approaches

The prioritization may employ existing requirement prioritization techniques using either of the two approaches:

- **Approach 1 (A1)**: Prioritization of non functional requirements together with functional requirements. This option is not a good option because non functional requirements are guaranteed to lose in competition to functional aspects.
- **Approach 2 (A2)**: Prioritization of non functional requirements separately from functional requirements. This approach is the good approach as mostly non functional requirements are prioritized by developers rather than users. But this is challenging because the selection of non functional requirements depends on the selection of functional requirements with which they are associated.
- **Approach 3 (A3)**: Hybrid of two approaches A1 and A2. In such a scheme the non functional requirements are given separate consideration, but are selected in accordance with the prioritized functional requirements. There is no competition between non functional and functional requirements for getting implemented in the current release. Thus the selection is separate for both the two requirements, although selection depends on the functionality of the system.

Aim and objectives of the paper

The aim of the paper is to examine the effectiveness of the three prioritization approaches (A1, A2 and A3) for non functional requirement prioritization for different complexity project versions. To fulfil the aim, this paper is based on the two objectives, first, to examine the accuracy of the prioritization approaches by using the suitable prioritization technique on suitable software versions and second, to analyze the impact of software complexity on the accuracy of prioritization approaches.

Experimentation

To meet the objectives, experimentation is conducted using suitable software versions, employing a suitable requirement prioritization technique for each prioritization approach (A1, A2 and A3). The Analytical Hierarchical Process (AHP) based cost-value prioritization technique (Karlsson and Ryan, 1997) is applied on three different complexity versions of same industrial software projects i.e. versions belonging to low, medium and high complexity. This technique is employed because pairwise comparison based prioritization technique had been found accurate by Karlsson (1996), Karlsson et al. (1998), Perini et al. (2009). The time limitation for performing the prioritization was relaxed to control the scalability variable. The scalability variable would otherwise have influenced the relation between complexity and accuracy as pairwise comparison based prioritization technique suffers from scalability issues as reported in Achimugu et al. (2014), Voola and Babu (2013), Perini et al. (2009), Ahl (2005), Karlsson et al. (1998), Karlsson et al. (2004), Lehtola and Kauppinen (2006), Ribeiro et al. (2011). The experimentation units are summarized in Table 1 as under.

The three versions of the selected project have 13 requirements (low complexity), 34 (medium) and 56 requirements (high complexity), to be subjected for prioritization.

Table 1 Experimentation details.

S. No.	Experimentation units category	Description of units
1.	Requirement prioritization technique	Analytical Hierarchical Process (AHP) in the form of cost-value approach
2.	Projects number and complexity	01 projects three versions, one of low complexity, one of medium and one of higher
3.	Independent variable	Complexity
4.	Dependent variable	Accuracy
5.	Control variable	Scalability