12- Points Requirement Prioritization Framework- Based on Mathematical Model and Pitfalls of Existing Prioritization Techniques

Arvind Kumar

Abstract—Software development or any other projects facing multiple requirements, budgetary constraints, and tight deadlines often necessitate the need to prioritize stakeholder's requirements. At some point, it's usually necessary to make decisions on which set of requirements need to be implemented first and which ones can be delayed till a later release. On a small project, the stakeholders can probably agree on requirement priorities informally. Larger or more contentious projects need a more structured approach, which removes some of the emotion, politics, and guesswork from prioritization. Requirements prioritization is an important activity and numerous different techniques to prioritize requirements exist and debated time to time but most of them were more focused on need of business requirements and accordingly prioritized but impact of prioritization was not factored. Through this paper, author has proposed new prioritization framework which consider both need of requirements along with impact of prioritized requirements.

Index Terms—Requirements, Requirements Prioritization, Prioritization Techniques, 12-Points Prioritization Framework

1 Introduction

WHEN it comes to the process of assigning a requirement's priority, any requirement may be prioritized at any point in its lifecycle, according to BABOK (BABOK 2.0 devotes a whole section to prioritizing requirements, noting, "Prioritization of requirements ensures that analysis and implementation efforts focus on the most critical requirements." Numerous methods on how to prioritize requirements have been developed and debated. While some work best on a small number of requirements, others are better suited to very complex projects with many decision-makers. But regardless of when it is done, before a requirement can be prioritized, an analyst must consider why requirement is most important from a business standpoint and what would be the impact of this on overall system. There are a number of possible business considerations, including value, cost, risk, and improve customer experience, stakeholder agreement and urgency variables but most of all prioritization methods focused on 'Why' (Why the work is important?) factor. Prioritizing business requirements requires a holistic approach.

The approach has to be framed on a set of principles and mathematical model linked together through a consistent framework while ensuring the outcome is aligned with the long term business goals, IT strategy and in-flight programs of the organization. Considering needs, I tried to come up with generic mathematical prioritization model which had six points linked to 'Why' and 6 points associated to 'What', hence named it as "12- Point Requirement Prioritization Framework"

The remainder of the paper is organized as follows. The next section presents the overview of existing prioritization techniques. In the subsequent section described proposed requirements framework.

2 Prioritizing Methods

There are numerous different techniques presented in this section that how to prioritize requirements. It might be difficult to pick the most suitable method because of the large number of them. Some methods are more time consuming than others but provide more accurate results. Some methods scale well to be used with larger number of requirements but provide
very coarse results. In other words, none of the techniques can really be considered the best one they all were based on need of requirements but not based on impact.

2.1 Numerical Assignment

Numerical Assignment prioritization method is mention by larger number of studies such as [1], [2], [3], [4], [5], [6]. Numerical Assignment is fundamental technique for requirement prioritization in which several groups of requirements prioritization are made and then requirements are assigned to one of these groups on the bases of their priority. Groups can such as low, medium and high are common groups. These groups should be clearly defined so that stakeholders do not have a different understanding of each during the prioritization exercise. To prevent stakeholders from putting all requirements in one category, the percentage of requirements that can be placed in each group should be restricted.

Pitfalls

Disadvantage to this, however, is the fact that requirements in each group will then have the same priority with no unique priority assigned per requirement.

• *Arvind Kumar working as IT Program/Project Management capacity and worked for various major IT companies and clients, Arvind is also a PMI certified Project Management professional and fellow member of professional body - IETE (The Institution of Electronics and Telecommunication Engineers) PH-+1 847-505-8617.*

E-mail: Arvind.kumar.aky@gmail.com

2.2 MoScoW Technique

MoScoW is a type of numerical assignment technique which is described in [7], [8]. Four priority groups which are MUST have, SHOULD have, COULD have and WONT have, are made in MoScoW. To prioritize requirements, each requirement will be place in one of the group based on their priority. Here are the pitfalls, I have seen with MoScoW. It is dependent on individual's inputs. Sometime this technique works well when you bring in a variety of stakeholders and talk through their different opinions on what needs to be done.

Pitfalls

1) Managers are worried that their requirements will fall into "should", and won't get done, so they make up reasons why their requirement is a "must". This ends up delaying business-critical functionality. (This is usually caused by, or exacerbated by, bad KPIs at an organizational level)

2) Time is spent discussing things that "should", "could" or "would" happen, delaying progress on the things which are absolutely essential. This is exacerbated by managers worried about the elements of the project which would help them achieve their goals, but which are not immediately essential (KPIs, again)

3) Architects and people in charge of other shared dependencies spend time creating support for all possibilities, instead of focusing on what "must" be done. Because it takes longer to actually create all the "shoulds" and "woulds", it takes longer to prove the architecture, so the feedback on whether the architecture is appropriate takes ages too. This is one of the things that cause architects to act as such strict gatekeepers, and to spend ages designing architectures in the first place.

2.3 Simple Ranking

Berander and Andrews [3] and Hatton [6] suggested simple ranking requirement prioritization technique. In simple ranking requirements are ranged from 1 to n where n is any integer value. Higher priority requirement are ranked by 1 and lower priority requirement are ranked by n.

Pitfalls

1) This technique has high volatility and prioritized based on what is to be done but lacking impact

2) Major drawback of this technique is that it does not provide criteria for categorization [9]. It is also difficult to implement on large number of requirements.

2.4 Bubble Sort

Hopcroft, Aho and Ullman [8], Karlsson [1] prioritize software requirements first by using Bubble sort technique. In bubble sort prioritization, two requirements are taken and then compared manually; if the person doing the comparison feels that 1st requirement should have higher priority than the other requirement then he/she swaps the priority and continues this process until all the requirements have been compared. The result will be a prioritized set of requirements.

Pitfalls

1) This prioritization technique doesn't work well with high number of requirements.

2) The major drawback of the bubble sort method is that it is difficult to evaluate the relative priority differences among the requirements [10].

2.5 Binary Search Tree

The binary search tree is a tree that has a parent-child relationship. The parent node usually contains at most two children [8]. Each parent node in the tree represents a requirement. Requirements with lower priority are arranged on the left side of the parent node while requirements placed on the right side of the parent node are of higher priority. A parent node with no child node is referred to as a leaf.

Pitfalls

1) It is however difficult to evaluate the relative priority differences among requirements [11].

2) In addition, the binary search tree produces unreliable results [12].

3) Another major criticism of the binary search tree is that it only provides a simple ranking of requirements without assigning any priority values [12]

2.6 Hundred Dollar Method

The hundred dollar method is otherwise known as proportional technique. It was introduced by Leffing well et al. [13]. The hundred dollars method was designed to determine important requirements by distributing a fabricated $100 note across requirements according to their degree or level of importance. The requirements are then sorted in ascending order in order to determine the number of dollar notes each requirement has earned.

Pitfalls

1) The drawback of this technique is that it is not suitable for a large number of requirements

2.7 Analytic hierarchy process (AHP)

The AHP was introduced by Thomas Saaty in 1980 for complex decision making. AHP was applied to software engineering by Joachim Karlsson and Kevin Ryan in 1997 [14]. AHP involves the pair-wise comparison of requirements in order to determine which of the two is of higher priority and to what extent. If n requirements are to be prioritized using AHP, then n*(n-1)/2 pair-wise comparisons are required. AHP results in an n by n matrix for n requirements. AHP uses a preference scale which generally ranges from 1 to 9, where 1 indicates requirements of equal value and 9 indicates extreme value

[15]. The AHP method according to Mead [16] consists of five basic steps which include:

1) The review of candidate requirements for completeness.

2) The application of pair-wise comparison method to assess the relative value of the candidate requirements.

3) Assessing the relative cost of implementing each candidate requirement.

4) Calculating the relative value and implementation cost of each requirement candidate and plot each on a cost-value diagram.

5) Using the cost-value diagram as a map for analyzing the candidate requirements.

Pitfalls

1) One of the major drawbacks of this technique is that it is not reliable in environments with multiple stakeholders. Hence, the use of AHP seems not feasible with large number of requirements.

2) In addition, AHP is very time consuming. It has also been observed that AHP contains a huge amount of redundancy [17].

2.8 Cost-Value Approach

This approach was introduced by Karlsson and Ryan [18]. This approach uses the AHP technique to compare requirements in pair-wise manner based on the relative values and cost of implementing the requirements.

Pitfalls

1) Studies have shown that the Cost-Value approach is time consuming [19].

3 New Proposed Mathematical Model Framework

This framework has mix of existing of prioritization and is based on top six categories ('Why') with which each stakeholder can relate their requirements and top six categories related to impact ('What') of each requirement category 'Why' hence twelve points are multi-dimensional. This framework works well when good number of stakeholders involved in prioritization phase and each stakeholder aligned with project benefits. This framework can be customized based on industry where it is to be applied. Categories described in below framework are mostly industry independent.

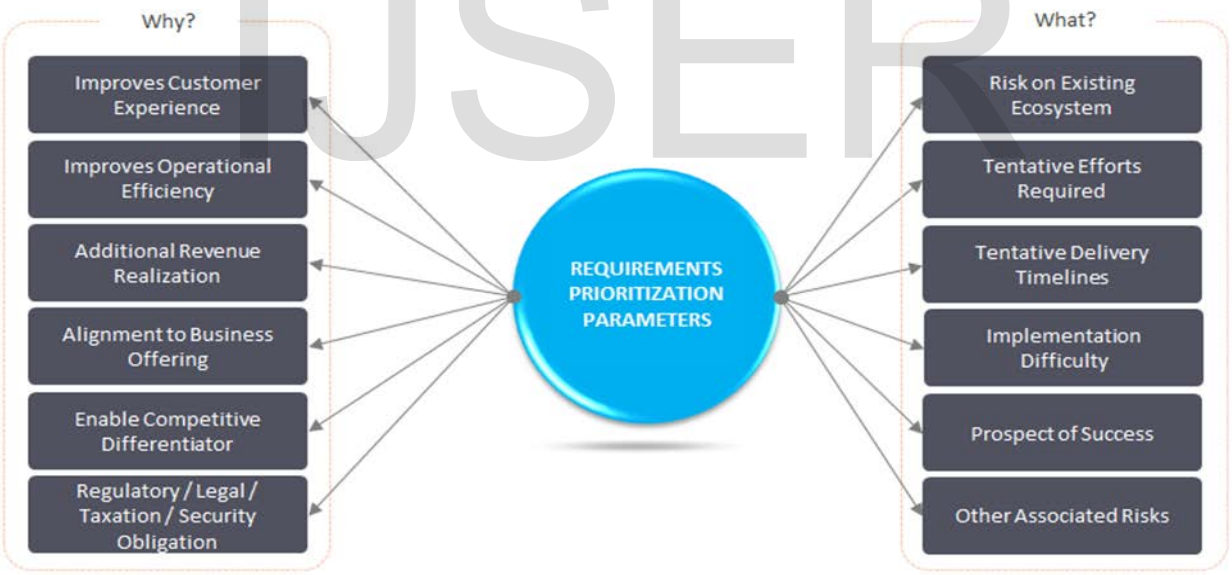

Fig. 1 Generic requirements prioritization parameters

3.1 Improves End Customer Experience

This approach focuses on the end user or customer experience of any given requirement; the requirements that will return the greatest "Wow" factor are given the highest priority. This focus on value helps to ensure "quick wins" for the organization.

3.2 Improves Operational Efficiency

Requirements often intermingle in complex relationships of other systems and sometime hamper operational efficiency. With this approach, requirements that support more operationally efficient system are also given high priority i.e. more of non-functional requirements.

3.3 Additional Revenue Realization

With an eye toward funding, this approach may be implemented a number of ways- implementing the least expensive requirements first or first implementing requirements with the greatest ROI (return on investment) or requirements which may lead to additional revenue.

3.4 Alignment to Business Offering

With this approach review the critical alignment between your product and your marketing strategy. Your product or service tactics should be interactive and interdependent with your marketing tactics. One of the most common mistakes caused by lack of prioritization is developing product or service features for their own sake, or because the team wants them, without regard to the target market, why people buy, and what they want and need.

3.5 Enable Competitive Differentiator

With this approach, requirements which really make difference for competitor are given higher priority.

3.6 Regulatory/Legal/Taxation/Security Obligation

With this approach, the requirements that are needed to meet legal and/or regulatory requirements are given highest priority. If an organization has a high priority (for marketing or legal reasons) to incorporate certain regulation such as Section 508 compliance, requirements that force accordance with section 508 would be given highest priority.

3.7 Risks on Existing System

This approach deprioritizes the riskiest requirements first, with the logic that they will fail; this approach often makes sense when a controversial or untested initiative is planned and has a high impact on overall system.

3.8 Tentative Efforts Required

This would deprioritize the requirement which needs higher efforts for implementation and more rigor planning to launch features or functions. These requirements can be decomposed based on design and implementation complexities.

3.9 Tentative Delivery Timelines

This approach would deprioritize the requirement which needs longer timelines for implementation. For continuous delivery, requirements should be decomposed based on design and implementation complexities and can be prioritized again in subsequent requirement prioritization phase.

3.10 Implementation Difficulty

The focus on difficulty of implementation, places the highest priority on the requirements that are the easiest to implement—the safest bets. The benefit of this approach is that it allows a project to get some project benefits deployed quickly, enabling customers and other stakeholders to become familiar with the project and give critical feedback before the organization moves forward to deploy more difficult aspects of the project.

3.11 Prospect of Success

Similar to difficulty of implementation, this method is frequently employed when a project is divisive and needs to shore up stakeholder support. It places highest priority on requirements with a high probability of success.

3.12 Other Associated Risks

Any other unknown-known risk can be categorized for this last point

4 WEIGHTAGE OF CATEGORIES AND STEPS

Weightages assigned to each category of *'Why'* and *'What'*. First six *'Why'* has positive number assigned from 05 to 30 and it's depends on business priority, weightage can be updated accordingly. Another six *'What'* has negative weightage assigned to categories.

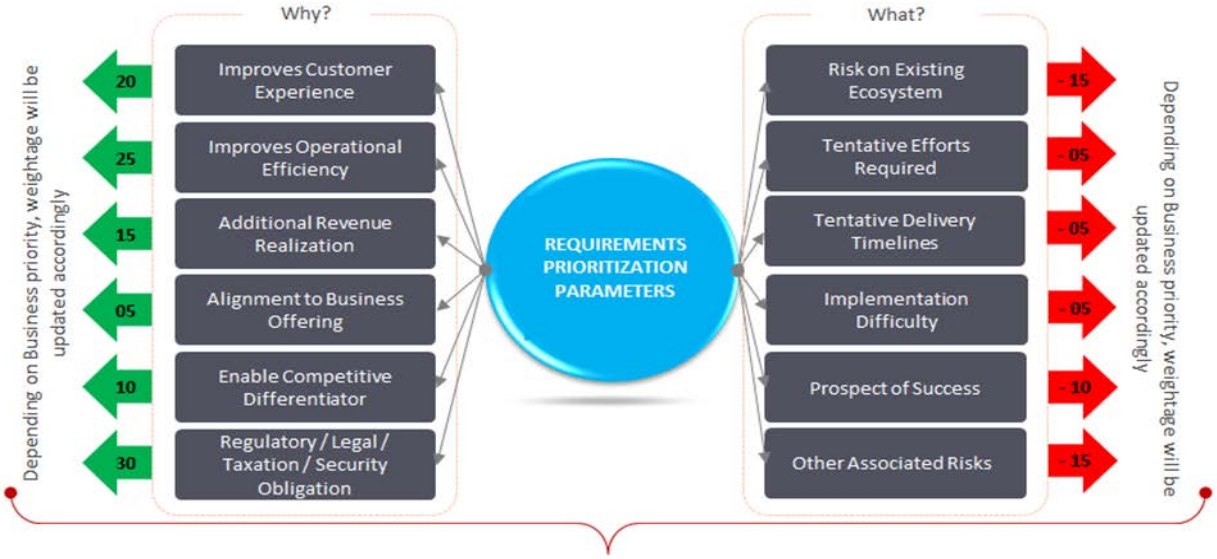

Fig. 2 Weightage of Categories

TARGET REQUIREMENTS AND PRIORITIZATION

	WEIGHTAGE	20		25		15		5		10		30		-15		-5		-5		-5		-10		-15		80			
	RATING	1		2		3		4		5																			
		Very Low		Low		Medim		High		Very High																			
		PRIORITIZATION PARAMETERS																											
		WHY												WHAT															
	HIGH LEVEL LIST OF FUNCTIONALITIES / REQUIREMENTS	STAKEHOLDER RESPONSIBLE FOR	IMPROVES CUSTOMER EXPERIENCE		IMPROVES OPERATIONAL EFFICIENCY		ADDITIONAL REVENUE REALIZATION		ALIGNMENT TO BUSINESS OFFERING		ENABLE COMPETITIVE DIFFERENTIATOR		LEGAL / TAX / REGULATORY / SECURITY OBLIGATION		RISK ON EXISTING ECOSYSTEM TO INTRODUCE THIS		TENTATIVE EFFORT FOR DELIVERING THE REQUIREMENT		TENTATIVE TIMELINES TO DELIVER THE REQUIREMENT		IMPLEMENTATION DIFFICULTY TO DELIVER THE REQUIREMENT		PROSPECT OF SUCCESS OF THE REQUIREMENT		OTHER ASSOCIATED RISK TO DELIVER THE REQUIREMENT		OVERALL SCORE	TARGET RELEASE
			RATING	SCORE	RATING	SCORE	RATING	SCORE	RATING	SCORE	RATING	SCORE	RATING	SCORE	RATING	SCORE	RATING	SCORE	RATING	SCORE	RATING	SCORE	RATING	SCORE	RATING	SCORE	RATING	SCORE		
1	Approval Management																													
1.1	Requirement -A	Arvind Kumar	2	40	1	25	3	45	1	5	2	20	3	90	2	-30	5	-25	2	-10	2	-10	2	-20	2	-30	100	Release 2.0
1.2	Requirement -B	Arvind Kumar	1	20	2	50	3	45	2	10	2	20	4	120	4	-60	4	-20	4	-20	4	-20	2	-20	4	-60	65	Release 3.0
1.3	Requirement -C	Arvind Kumar	4	80	2	50	1	15	1	5	3	30	1	30	3	-45	3	-15	3	-15	3	-15	3	-30	3	-45	45	Release 3.0
1.4	Requirement -D	Arvind Kumar	1	20	1	25	3	45	3	15	4	40	5	150	1	-15	5	-25	5	-25	5	-25	3	-30	5	-75	100	Release 2.0
1.5	Requirement -E	Arvind Kumar	2	40	2	50	4	60	2	10	3	30	1	30	3	-45	4	-20	4	-20	4	-20	4	-40	4	-60	15	Release 3.0
1.6	Requirement -F	Arvind Kumar	4	80	2	50	4	60	2	10	3	30	3	90	2	-30	4	-20	4	-20	4	-20	4	-40	4	-60	130	Release 1.0
1.7	Requirement -G	Arvind Kumar	5	100	1	25	5	75	2	10	3	30	4	120	1	-15	5	-25	5	-25	5	-25	1	-10	3	-45	215	Release 1.0
1.8	Requirement -H	Arvind Kumar	5	100	5	125	1	15	2	10	3	30	2	60	2	-30	2	-10	2	-10	2	-10	2	-20	2	-30	230	Release 1.0

Fig. 3 Sample Spreadsheet for Mathematical Model

Step 1.List all of the requirements, features, or use cases that you wish to prioritize in a spreadsheet; All of the items must be at the same level of abstraction.

Step 2.Estimate the relative benefit that each functionality/requirement provides to the customer or the business on a rating from 1 to 5, with 1 indicating very little benefit and 9 being the maximum possible benefit. These benefits indicate alignment with the product's business requirements. Your customer representatives are the best people to judge these benefits.

Step 3.In order to derive score against a respective Parameter, multiple provided rating and weightage. Score column in spreadsheet would represent calculated score for each parameter.

Step 4.Rating (one to five) for 'Why 'category will result positive score whereas rating (one to five) for 'What' will results negative score so if you have higher ratings for 'What', this will deprioritize your requirements.

Step 5.Then device the total scores against requirement by adding all the score. That will be overall score for requirements.

Step 6.Based on total score, map the Release. Wherein the score is higher, it has a highest priority. Sort the list of features in descending order by calculated overall score. The functionalities/requirements at the top of the list have the most favorable balance of value, cost, risk and timelines and thus should have higher implementation priority. The key customer and developer representatives should review the completed spreadsheet to agree on the ratings and the resulting sequence based on score. Based on Figure 3 sample spreadsheet, requirements F, G and H has higher score hence they can be mapped as release 01.

5 BENEFITS AND PITFALLS

5.1 Benefits

1) Software development process is usually a collaborative effort of diverse stakeholders such as IT business, users, designers, software architects and the coders. However, the number of stakeholders as well as their requirements may increase during a software development life cycle. Hence, there is a need for a software requirement prioritization technique that will be able to handle large number of stakeholders and their requirements. The above mentioned framework can handle large number of requirements along with large number of stakeholders.

2) The development of software systems is required to be very fast. This is because the software industry is evolving fast and thus a variety of particular software is easily available off-the-shelf. It is therefore necessary to complete a software development project within a stipulated time by using a software requirement prioritization technique that requires less time and moreover stakeholders should agree on prioritization results based on mathematical number rather politically influenced.

3) Stated framework is easy to use and very effective considering the project's limited resources such as cost and

stakeholders time. Hence, the need for a requirement prioritization technique that is simple, effective, easy to use as well as boosts the confidence of users.

4) Consistency check is a measure that is used to show consistency in the judgment of decision makers since human judgment can be inconsistent. Framework support bidirectional prioritization which is based on need of requirements and impact of those requirements.Impact value nagates the overall priorization score and impact value reviewed with all stakeholders to confirm the judgement.

5) This framework proposed on how to drive beneath the emotions and politics of who is driving what requirements and driving for a value add disucssion of the impact of each requirement against the project scope, contraints of cost, time and quality.

6) This framework helps in daily business analysis tasks and provide helpful support for agile release planning

7) Parameters can be easily tailored to different types of projects.

8) Framework supports quantitative and objectively prioritization and could be applied to manage projects, programs and project portfolios.

9) Framework support priorization agility means prioritized requirements would be only those requirements which are really needed by business without having impact on existing user base or running system.

5.2 Pitfalls

1) This framework demands stakeholders to be functionally and technically aware of project or exsiting system on which they are participating for prioritization activity.

2) Getting agreement on requirements classifications can be difficult for all stakeholders.

6 Conclusion

This paper gave an overview of existing prioritization methods and associated disadvantages. Customers are never thrilled to find out they can't get all the functionalities they want in release 1.0 of a new software product (at least, not if they want the functionalities to work). However, if the development team cannot deliver every requirement by the scheduled initial delivery date, the project stakeholders must agree on which subset to implement first. Any project with resource limitations has to establish the relative priorities of the requested functionalities/requirements, use cases, or non-functional requirements. Prioritization helps the project manager resolve conflicts, plan for continuous deliveries, and make the necessary trade-off decisions.
Any actions you can take to move requirements prioritization from the political arena into an objective and analytical one will improve the project's ability to deliver the most important functionality in the most appropriate sequence.

References

[1] J.karlsson, C.Wolin and B. Regnell, An evaluation of methods for prioritizing software requirements, information and software technology, pp 939-947,2007.
[2] IEEE-STD 830-1998, "IEEE recommended practice for software requirement specifications.", IEEE computer society.
[3] S. Brender, Key words for use in RFC's to indicate requirements levels, RFC 2119.
[4] D. Leffingwell & D. widring, managing software requirements - A unified approach, upper Saddle River: Addison- Wesley.
[5] I. Sommerville & P. Sawyer, Requirements engineering, A good practice guide, Vhichester: John wiley and sons, May 5, 1997.
[6] S. Hatton, Early prioritization of goals, M.K Jean-Luchainaut, Elke A. Rundensteiner Ed., Springer Berlin Heidelberg, 2007.
[7] Dsdm Public version 4.2, from www.dsdm.org, Tech. Rep., Retrieved, 6 June, 2009.
[8] A.V. Aho, J.D. Ullman & J.E.Hopcroft, data structure and algrithems, Reading, MA: Addison-Wesley, January 11, 1983.
[9] Ramzan M, Jaffar A, Shahid A. Value based intelligent requirement prioritization (VIRP): expert driven fuzzy logic based prioritization technique. International Journal of Innovative Computing. 2011; 7(3):1017–1038
[10] Achimugu P, Selamat A, Ibrahim R, Naz'ri Mahrin M. A systematic literature review of software requirements prioritization research. Information and Software Technology. 2014;56:568–585
[11] Thakurta R. A framework for prioritization of quality requirements for inclusion in a software project, Software Quality Journal. 2012;21:573–597.
[12] Duan C, Laurent P, Cleland-Huang J, Kwiatkowski C. Towards automated requirements prioritization and triage. Requirement Engineering. 2009;14(2):73–89.
[13] Leffingwell D, Widrig D. Managing software requirements: A unified approach, Addison-Wesley Longman Inc; 2000
[14] Saaty TL. The analytic hierarchy process. New York: McGraw-Hill; 1980.
[15] Sadiq M, Ahmed J, Asim M, Suman Q, More on elicitation of software requirements and prioritization using AHP. International Conference on Data Storage and Data Engineering, IEEE; 2010.
[16] Mead NR. Requirements prioritization introduction. Software Engineering Institute Web Publication Carnegie Mellon University, Pittsburgh; 2006
[17] Franceschini F, Rupil A, Rating scales and prioritization in QFD. International Journal of Quality Reliability and Management. 1999;16(1):85–97
[18] Karlsson L, Höst M, Regnell B. Evaluating the practical use of different measurement scales in requirements prioritization. Proceedings of the 2006 ACM/IEEE International Symposium on Empirical Software Engineering. 2000;326–335.
[19] Achimugu P, Selamat A, Ibrahim R, Naz'ri Mahrin M. A systematic literature review of software requirements prioritization research. Information and Software Technology. 2014;56:568–585.