01 Jan 2004

Priority Assessment of Software Requirements from Multiple Perspectives

Xiaoqing Frank Liu
Missouri University of Science and Technology, fliu@mst.edu

Chandra Sekhar Veera

Yan Sun

Kunio Noguchi

et. al. For a complete list of authors, see https://scholarsmine.mst.edu/comsci_facwork/301

Recommended Citation

X. F. Liu et al., "Priority Assessment of Software Requirements from Multiple Perspectives," *Proceedings of the 28th Annual International Computer Software and Applications Conference, 2004. COMPSAC 2004*, Institute of Electrical and Electronics Engineers (IEEE), Jan 2004.
The definitive version is available at https://doi.org/10.1109/CMPSAC.2004.1342872

Priority Assessment of Software Requirements from Multiple Perspectives

Xiaoqing (Frank) Liu, Chandra Sekhar Veera, and Yan Sun
University of Missouri – Rolla
Computer Science Department
fliu@ umr.edu

Kunio Noguchi and Yuji Kyoya
Software Engineering Center
Toshiba Corporation
Japan

Abstract

The development of complex software systems involves collecting software requirements from various stakeholders. Often stakeholder perceptions conflict during the requirements elicitation phase. An effective technique for to resolve such a conflict is needed. In this paper, we presented a framework that prioritizes software requirements gathered from multiple stakeholders by incorporating inter-perspective relationships, which is not addressed by existing priority assessment techniques. We use a relationship matrix to analyze the impact between requirements and facilitate the integration process which assesses their priorities based on their relationships from multiple perspectives. It allows the development team to resolve conflicts effectively and concentrate their valuable time and resources on the critical few requirements from multiple perspectives that directly contribute to high customer satisfaction.

1. Introduction

The pressure on time-to-market and being able to plan for successive releases of software products has posed significant challenges on the requirements engineering process. Effectively negotiating requirements from various stakeholders who have different roles and responsibilities during the early stages of the software development is a key factor of successful software projects [1], because it can reduce variation and wastage of time while optimizing the software product. By addressing high-priority requirements before low-priority ones, one can significantly reduce project costs and duration [4]. It is difficult enough for a customer to decide which of his requirements are most important; achieving consensus among stakeholders with diverse expectations is even more challenging [17]. Factors concerning different stakeholders such as business value, risks, relation to other requirements, etc., should be considered while prioritizing requirements.

Several prioritization methods have been proposed [6], each of which uses different mathematical or analytic approaches for requirements prioritization.

One such approach is AHP proposed by Saaty that uses pair-wise evaluation by hierarchy level [15], which is quite complicated and time consuming. Therefore, it is considered impractical for large projects with many requirements [3]. Zultner proposed a requirements prioritization technique that involves multiple stakeholders or customers [18]. However, he uses AHP in determining the priorities of multiple customers. Karlsson also used the AHP concept and developed a cost-value approach for prioritizing requirements [5]. He compared the requirements in a pair- wise fashion with regard to their relative value of importance and relative cost to implement. However, This approach may result in a computational explosion and it does not accommodate multiple prioritization models that may be in better alignment with stakeholder needs [13].

Moisiadis presented a tool (RPT) for requirement prioritization [11] which prioritizes requirements based on business goals and stakeholder viewpoints. However, it did not overcome the limitations of commonly used requirement prioritization approaches as he listed, and impact relationships between requirements are not considered.

Park, Port and Boehm proposed a distributed collaborative prioritization of software requirements model [13], where disparately located stakeholders negotiate the relative priorities using priority bins Although the model can identify conflicts between requirements during renegotiation process as software

Proceedings of the 28th Annual International Computer Software and Applications Conference (COMPSAC'04)
0730-3157/04 $20.00 © 2004 **IEEE**

evolves, it does not address interdependencies between prioritized sets of software requirements.

Sivzattian and Nuseibeh proposed a portfolio-based approach to prioritize and select requirements [16]. This approach selects requirements based on the trade-off between effort and return. However, treating requirements as capital assets and applying the U.S. capital market risk-free rates and average return rate to the prioritization of requirements deserves more validation. Some other studies proposed the requirements specification from multiple views using "viewpoint" [2, 7, 12], which allows the use of various specification methods. While facilitating the tasks for each viewpoint, the integration of various viewpoint specifications remains a challenging research area. Furthermore, none of them considered requirements prioritization from multiple perspectives.

2. Our research approach

This framework helps bring together and prioritize diverse requirements from many sources and achieve global stakeholder agreement on software requirements before actual development of software products. Some studies have been done to analyze requirements with the help of QFD [8, 9, 10]. It is noted that requirements from customers may have relationships with requirements from other stakeholders. Sometimes different stakeholder requirements go in harmony with each other, i.e., they share similar attributes thereby facilitating an increase in the priority level of those requirements. In such cases, the degree of consensus determines the level of assurance in the priority value of a requirement.

Our framework uses relationship matrix to integrate and prioritize requirements from different perspectives. The relationship matrix helps us to understand the relationships between various stakeholder requirements, i.e., how one stakeholder's requirements positively contribute to other stakeholders' requirements. It helps eliminate miscommunication by graphically classifying objectives and interactions for easier use and effective decision making (figure 3).

Identification of priorities of business needs and customer requirements is generally looked upon as important within requirements prioritization. It will help the project group to allocate resources effectively. When business needs are mapped against customer requirements in the relationship matrix, we gain valuable information about missing business needs by showing requirements that do not contribute to any of the existing business needs. On the other hand, important business needs that do not have adequate functional requirements to support them will force the project group to come up with more requirements that contribute to addressing these needs.

The remainder of the paper is structured as follows. Section 3 describes the process of prioritizing requirement sets obtained from multiple perspectives. The integration and reassessment of requirement priorities is explained in section 4. An application example for an order processing system is summarized in section 5. Section 6 discusses the validation of the framework and the concluding section provides a discussion on its significance.

3. Prioritization scheme

During the requirements elicitation phase, software requirements are collected from multiple stakeholders (users, developers, managers, etc.). These sets of low level requirements can usually be grouped into some high level requirements. In order to ensure that the software system reflects multiple stakeholder needs, we prioritize and integrate their requirements to focus the limited resources in satisfying those requirements. Initially, low level requirements together with their associated high level requirements are organized in the form of hierarchies (Figure 1). High level requirements form the root in every hierarchical structure.

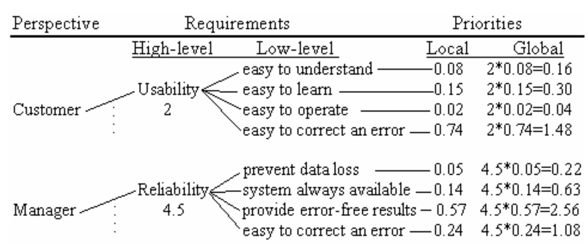

Figure 1. Prioritizing a hierarchy of requirements

The prioritization of software requirements can be performed either absolutely or relatively. In the case of absolute evaluation, each requirement is assigned a value between 1 and 5. This process requires less effort compared with relative evaluation. However, using absolute evaluation, stakeholders tend to assign high values to all requirements, which ultimately affects the quality of the results. Hence, the following relative evaluation is recommended instead.

Step 1: Establish a linkage between each pair of high level requirements in the set by identifying their relative dominance value. If stakeholders can specify exactly an amount of satisfaction degree of a requirement that they are willing to sacrifice for an increase in the satisfaction degree of another requirement, we can obtain the relative priority [9]. For instance, if stakeholders can accept an alternative which increases the satisfaction degree of requirement R_1 by 1 and decreases the satisfaction degree of requirement R_2 by 2, while the satisfaction degrees of other requirements remain unchanged, the relative dominance of R_1 over R_2 is given by the equation, $rd_{1,2} = n_1/n_2$, where n_1 is the satisfaction degree of R_1, and n_2 is the satisfaction degree of R_2. The numeric value of satisfaction degree typically comes from a consensus by all stakeholders. If the values given by customer representatives vary, discussions are needed and a uniformly agreed value has to be generated.

Step 2: Calculate the local priority of each requirement at the same level. Suppose that there are n requirements, $R_1, R_2, \ldots, R_n$, in a decreasing order of importance. Let $rd_{i,j}$ denote the relative dominance of R_i over R_j. Initially, W_{Rn} is assigned to be one. The local priorities of all remaining requirements can be determined recursively using the equation [9]:

$$\text{For } 1 \leq i < n,\ W_{Ri-1} = W_{Ri} * rd_{i,\, i-1}$$

While assessing the local priorities of requirements, one should ensure that these priorities are consistent. The local priorities are said to be consistent if and only if there exists $1 \leq i, j, k \leq n$ such that,

$$rd_{i,k} \neq rd_{i,j} * rd_{j,k} \text{ [9]}$$

Step 3: Normalize all local priorities.

Step 4: For every level in the hierarchy, if sub-level requirements exist, the sub-level requirements are compared and prioritized using the steps 1, 2 and 3.

Step 5: The local priorities are multiplied by the priority of the parent requirement to obtain global priorities (figure 1). For instance, from the customers' perspective, *usability* is a high level requirement containing four low level requirements, among which *easy to understand* receives a normalized local priority of 0.08. This number is multiplied by 2, which is the priority of *usability*, to obtain a global priority of 0.16.

Above steps are applied to all requirements in the remaining hierarchies.

4. A framework to integrate priorities from multiple perspectives

This section discusses an approach to integrate stakeholder requirements from multiple perspectives into one concise set of prioritized requirements. Perspectives represent the views that stakeholders hold on different areas of concern. A simple relationship matrix forms the basis for requirements integration. The following steps discuss how to complete different components of a relationship matrix (Figure 2).

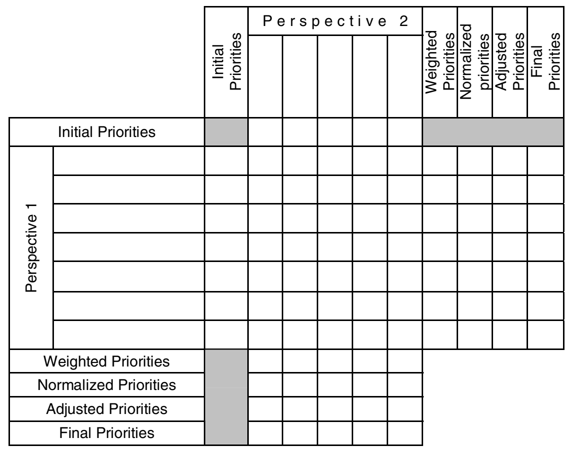

Figure 2. Relationship Matrix

1. Enter stakeholder requirements from two perspectives.
2. Enter initial priorities: These are the set of normalized global priorities obtained in the previous section.
3. Determine the Impact Relationships: The requirements from two perspectives are carefully examined and correlations are assigned using the impact relationship symbols as shown in Table 1.

Table 1. Types of impact relationships

Impact	Symbol	Value
Strong	◈	0.9
Medium	□	0.3
Weak	▽	0.1

4. Calculate weighted priorities by considering the cross impact. Each impact relationship defined in the previous step is multiplied by the initial priority values of the two requirements involved. The results are summed up for each individual requirement in the matrix. For requirement X from perspective 1 and requirement Y from perspective 2, the weighted priority is calculated as:

   Weighted Priority (X) = Σ_y (Initial priority(X) *correlation(Y,X) * Initial priority (Y))

Proceedings of the 28th Annual International Computer Software and Applications Conference (COMPSAC'04)
0730-3157/04 $20.00 © 2004 **IEEE**

Weighted Priority (Y) = Σ_x (Initial priority(X) *correlation(Y,X) * Initial priority (Y))

5. Calculate normalized priorities: For requirement X from perspective 1 and requirement Y from perspective 2, normalized priority is calculated as:
   Normalized Priority(X) = Weighted priority(X) / $\Sigma_{k(P1)}$Weighted Priority($k_{(P1)}$)
   Normalized Priority (Y) = Weighted priority(Y) / $\Sigma_{k(P2)}$Weighted Priority($k_{(P2)}$)

   In the above equations, $k_{(P1)}$ and $k_{(P1)}$ are number of requirements in perspectives 1 and 2, respectively.
6. Calculate adjusted priorities using an adjustment factor α between 0 and 1 to adjust the relative importance of impact relationship. If the cross impact is considered as important as the initial priority, α = 1 is used. As the relative importance of cross impact decreases, α value decreases accordingly, until it becomes 0, which means the cross impact can be ignored. For requirement X from perspective 1 and requirement Y from perspective 2, the adjusted priority is calculated as:
   Adjusted Priority(X) = Initial priority(X) + α * Normalized priority(X)
   Adjusted Priority(Y) = Initial priority(Y) + α * Normalized priority(Y)
7. Calculate final priorities: For requirement X from either perspective 1 or perspective 2, the final priority is calculated as:
   Final Priority(X) = Adjusted priority(X)/ $\Sigma_{k(P1)}$ Adjusted Priority($k_{(P1)}$)+ $\Sigma_{k(P2)}$ Adjusted Priority($_{k(P2)}$)

The final priorities are used to integrate the two sets of requirements into one. This new set integrated with requirements from the third perspective and so on. Suppose that there are 'n' perspectives. We start by constructing a relationship matrix between the first two perspectives and perform the impact relationship analysis. Next, we calculate final priorities and use them to construct a second relationship matrix with the third perspective. This procedure is repeated until all 'n' perspectives are integrated (Figure 3).

When a large number of perspectives have to be considered, it is easier to construct relationship matrices for perspectives one and two, perspectives three and four, and so on, up to perspectives n-1 and n. Next, construct the relationship matrices for requirements sets obtained in the previous step, i.e., construct relationship matrix for perspective one and two together versus perspective three and four and so on. This process is repeated until all perspectives are integrated into one set (Figure 4).

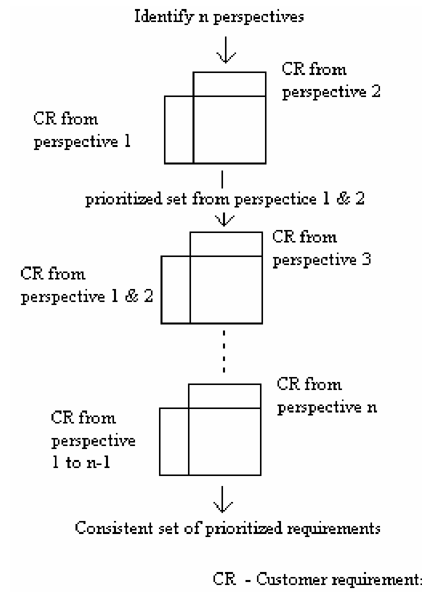

Figure 3. Prioritization of requirements from 'n' perspectives: Framework 1

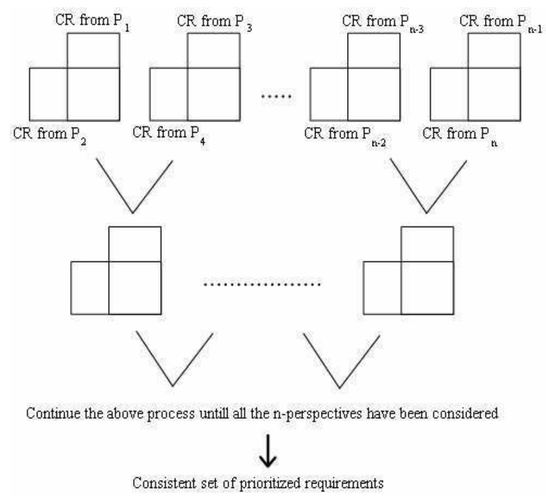

Figure 4. Prioritization of requirements from 'n' perspectives: Framework 2

5. Application example

The priority assessment of requirements from multiple perspectives framework is illustrated using an "Order Processing System", where we assume that requirements are elicited from three perspectives -- customers, developers and managers.

Suppose that *usability, maintainability, reusability,* *portability, efficiency,* and *reliability* form the high level requirements. After prioritizing them using the scheme described in section 3, we identify *usability, reliability* and *efficiency* as high priority requirements which are selected for lower level requirements prioritization and integration. Low level requirements from each perspective that link to *usability, reliability* and *efficiency* are shown in Table 2.

Table 2. Low-level requirements used in application example

Perspective	Low-Level Requirements
Customer	Easy to understand Easy to learn Easy to operate Easy to correct an error
Manager	Prevent data loss System is always available Provides error-free results System is highly secure
Developer	Provide fast response Be resourceful Provide good throughput Adhere to standards

These low-level requirements from perspective 1 and 2 are integrated using relationship matrix 1 (Figure 5). Based on the impact relationships given by stakeholder representatives, we calculate priorities for all requirements. The ranked and re-normalized final priorities are used in relationship matrix 2 (Figure 6) to integrate with requirements from perspective 3. An α value of 1 is used consistently in the two relationship matrices. In this example, in order to prevent computational explosion, four requirements with higher priorities out of eight from the result of Figure 5 are used for the integration with perspective 3.

6. Validation

Our framework of priority assessment for requirements from multiple perspectives has been validated in two other case studies in addition to the one introduced in this paper. One case study successfully applied this framework to the prioritization of requirements from multiple perspectives in object-oriented software development. In the other case study, the same framework was used to assess priorities of software process requirements for software process improvement based on CMM [14] using QFD. In these case studies, the researchers found this framework helpful in identifying key requirements and assessing their priorities objectively

	Initial Global Priorities	prevent data loss	system is always available	provides error-free results	system is highly secure	Weighted Priorities	Normalized Priorities	Adjusted Priorities	Final Priorities
Initial Global Priorities		0.12	0.35	0.35	0.18				
easy to understand	0.33			▽	▽	0.17	0.08	0.41	0.21
easy to learn	0.23	○		○		0.32	0.15	0.38	0.19
easy to operate	0.11	▽		▽		0.05	0.02	0.13	0.07
easy to correct an error	0.33	▽	○	●	○	1.60	0.74	1.07	0.54
Weighted Priorities		0.14	0.35	1.44	0.24				
Normalized Priorities		0.06	0.16	0.67	0.11				
Adjusted Priorities		0.18	0.51	1.02	0.29				
Final Priorities		0.09	0.26	0.51	0.15				

Figure 5. Relationship Matrix 1: customer vs. manager

	Initial Global Priorities	provide fast response	be resourceful	provide good throughput	adhere to standards	Weighted Priorities	Normalized Priorities	Adjusted Priorities	Final Priorities
Initial Global Priorities		0.28	0.17	0.33	0.22				
always available	0.17	○	●	○	▽	0.61	0.26	0.43	0.22
erro-free results	0.34		○	○		0.51	0.22	0.56	0.28
easy to understand	0.14		○		▽	0.10	0.04	0.18	0.09
easy to correct an error	0.36	○	○	○	○	1.08	0.47	0.83	0.41
Weighted Priorities		0.45	0.69	0.86	0.31				
Normalized Priorities		0.19	0.30	0.37	0.13				
Adjusted Priorities		0.47	0.47	0.70	0.35				
Final Priorities		0.24	0.23	0.35	0.18				

Figure 6. Relationship Matrix 2: customer and manager vs. developer

and accurately. They also found it helpful in understanding the relationships between requirements from multiple perspectives.

Case studies have shown that impact relationships play an important role in assessing requirement priorities. If a requirement has more and stronger impacts on the satisfaction of other requirements, its priority is increased. If adequate resources can be allocated to satisfy this requirement, a higher level of overall requirement satisfaction can be achieved. This is reasonable because satisfying this requirement contributes to the satisfaction of other requirements.

This methodology is developed to integrate requirements from multiple perspectives each of which has its own area of concern. Obviously, some requirements are functional while others are non-functional. This framework provides a means to correlate and integrate all these functional and non-functional requirements together so that higher levels of requirement satisfaction can be achieved.

Following small scale experiments in our case studies, the framework will be further validated by the Toshiba Corporation in a large scale environment. The training on and application of our novel software QFD framework which contains this priority assessment method is currently being carried out. We will report the application results in the future.

7. Conclusion

The presented work is aimed at priority assessment of requirements from multiple perspectives. Considering relationships between requirements from different stakeholders during the prioritization process allows us to create a common understanding of the problem domain, and their needs for the software system. We try to develop a framework which can help organizations to identify key issues which they ought to pay attention to from various perspectives by allowing them to prioritize requirements elicited from multiple stakeholders. Using this framework, requirements with more and stronger impacts on other requirements from multiple perspectives have higher priorities. Satisfying these requirements can increase the overall requirement satisfaction. The collaborative assessment of requirement priorities provides a valuable roadmap to alleviate project risks that are detrimental to the software project, and to improve software quality that is mutually beneficial to all stakeholders.

References

[1] Boehm, Barry , Hoh In, "*Identifying Quality-Requirement Conflicts", IEEE Software*, March 1996, pp. 25-35.

[2] Easterbrook, Steve and Bashar Nuseibeh, "Using ViewPoints for Inconsistency Management", *IEE Software Engineering Journal*, November 1995.

[3] Finnie, G.R., G.E Wittig, and D.I. Petkov, "Prioritizing software development productivity factors using the Analytic Hierarchy Process"*, Journal of Systems and Software*, vol. 22, 1995, pp. 129-139.

[4] Hofmann, Hubert F. and Franz Lehner*,* "Requirements Engineering as a Success Factor in Software Projects"*, IEEE Software,* July/August 2001, pp. 58-66

[5] Karlsson, J., K. Ryan, "A Cost-Value Approach for Prioritizing Requirements", *IEEE Software,* Sept. 1997, pp. 67-74.

[6] Karlsson, J., C. Wohlin and B. Regnell*,* "An Evaluation of Methods for Prioritizing Software Requirements", *Journal of Information and Software Technology,* 1998, Vol. 39, No. 14-15, pp. 939-947.

[7] Kotonya, Gerald and Ian Sommerville, "Viewpoints for Requirements Definition", *Software Engineering Journal,* November 1992, pp. 375-387.

[8] Liu, Xiaoqing Frank and John Yen, "An Analytic Framework for Specifying and Analyzing Imprecise Requirements", Proc. of *the 18th IEEE International Conference on Software Engineering (ICSE-1996) (long paper)*, pp. 60-69, Berlin, Germany, March, 1996.

[9] Liu, Xiaoqing Frank, "A quantitative approach for assessing the priorities of software quality requirements", *Journal of Systems and Software,* 1998, pp. 105-113.

[10] Liu, Xiaoqing Frank, Kunio Noguchi, and Weihua Zhou, ``Requirement Acquisition, Analysis, and Synthesis in Quality Function Deployment", *International Journal of Concurrent Engineering: Research and Applications*, Vol. 9, No. 1, March 2001.

[11] Moisiadis, Frank, "The Fundamentals of Prioritising requirements", *Systems Engineering, Test& Evaluation Conference*, Sydney, Australia, October 2002.

[12] Nuseibeh, Basher, Steve Easterbrook, and Alessandra Russo, "Leveraging Inconsistency in Software Development", *Computer*, 33(4):24-29, IEEE Computer Society Press, April 2000.

[13] Park, Jung-Won, Daniel Port and Barry Boehm, "Supporting Distributed Collaborative Prioritization for WinWin Requirements Capture and Negotiation", *In Proceedings of the sixth Asia Pacific Software Engineering Conference (APSEC),* 1999, pp. 560--563.

[14] Paulk, Mark C., Bill Curtis, Mary Beth Chrissis, Charles V. Weber, "Capability Maturity Model for Software, Version 1.1", *Technical Report, CMU/SEI-93-TR-024, ESC-TR-93-177*, February, 1993.

[15] Saaty, Thomas L., "Fundamentals of Decision Making and priority theory with the Analytical Hierarchy Process", *RWS Publications,* Pittsburgh, PA, 1994.

[16] Sivzattian, Siv and Bashar Nuseibeh, "Linking the Selection of Requirements to Market Value: A Portfolio-Based Approach", *In Proceedings of 7th International Workshop on Requirements Engineering: Foundation for Software Quality (REFSQ 2001),* June 2001.

[17] Wiegers, Karl E.*, Software Requirements,* Microsoft Press, Redmond, Washington, 1999.

[18] Zultner, R.E., 'Project QFD - Managing Software Development Better with Blitz QFD', *9th Symposium on QFD,* 1997, pp. 15-26.