©2010 International Journal of Computer Applications (0975 – 8887)
Volume 1 – No. 9

Adding Threat during Software Requirements Elicitation and Prioritization

Mohd. Sadiq
Computer Engineering Section, University Polytechnic, Faculty of Engineering and Technology, Jamia Millia Islamia (A Central University), New Delhi-25 (India)

Mohd. Shahid
M. Tech. Scholar, Department of CSE, Al-Falah School of Engineering and Technology, Dhauj, Faridabad, MDU Rohtak, Haryana (India)

Shabbir Ahmad
Electronics Engineering Section, University Polytechnic, Faculty of Engineering and Technology, Jamia Millia Islamia (A Central University) New Delhi-25 (India)

ABSTRACT

Requirements may be defined as a demand or need. In software engineering, a requirement is a description of what a system should do. System may have dozen to thousands of requirements. Software requirements stipulate what must be accomplished, transformed, produced or provided. In the field of software engineering researchers, academicians and scientist have developed many models and framework to elicit and prioritize the software requirements. It is well documented that requirement engineering saves money. There are several techniques to elicit the software requirements like JAD, misuse, RAD etc. In this paper we have used the JAD approach to elicit the software requirements. In this paper we have proposed a framework to elicit the software requirements and also to prioritize the software requirements. The proposed framework will rank the requirements by the relative level of threat associated with each requirement.

Keywords:

Software Requirements, Elicitation Techniques, Analytic Hierarchy Process, and Quality Function Deployment.

Category and Subject Descriptors:

D2.9 [Software Engineering] Management, Cost Estimation, Productivity, and Programming Team.

General Terms:

Measurement, Experimentation

1. INTRODUCTION

Elicitation is all about determining the needs of stakeholders and learning, uncovering extracting and /or discovering needs of the users and other potential stakeholders [2]. Requirement elicitation is recognized as one of the most critical knowledge intensive activities of the development of software. Studies by [3] indicate that 70% of the system errors are due to the inadequate system specification and 30% of the system errors are due to design issue. The analysis of secure software system based on the system requirements elicited in the form of use case and misuse case. Use cases have proven helpful for elicitation of communication about, and documentation of the function requirements. The integral development of use and misuse cases [8, 10, and 11] provides a systematic way for the elicitation of both the functional and non functional requirements [13]. Using an elicitation method can help in producing a consistent and complete set of security requirements. However, brainstorming and elicitation methods used for ordinary functional (end-user) requirements usually are not oriented toward security requirements and do not result in a consistent and complete set of security requirements. The resulting system is likely to have fewer security exposures when security requirements are elicited in a systematic way. In this paper we have used the JAD approach to elicit the software requirements. A number of requirements elicitations techniques have been developed to extract requirements from a user. The goal of JAD (Joint Application Development) is to involve *all* stakeholders in the design phase of the product via highly structured and focused meetings. Typical participants in the session include a facilitator, end users of the product, main developers, and observers. In the preliminary phases of JAD, the requirements-engineering team is tasked with fact finding and information gathering. Typically, the outputs of this phase, as applied to security requirements elicitation, are security goals and artifacts. The actual JAD session is then used to validate this information by establishing an agreed-on set of security requirements for the product. If JAD has some advantages so it has also some disadvantages. The important disadvantage of JAD is that if there are too many JAD sessions while the project is progressing then user may develop a feeling that the developer are shifting their work and responsibility onto the users. To get the detailed description about the remaining techniques please refer to [2] [18] [21]. The paper is organized as follows: In section 2 we present the background and related work. In section 3 we have proposed the framework that will rank the requirements by the relative level of threat associated with each requirement. In section 4, experimental work is carried out, and finally we conclude the paper in section 5.

2. BACKGROUND AND RELATED WORK:

D. Firesmith [6], have worked for prioritization dimensions, prioritization approach, prioritization techniques and processes. This paper does not explain how the software requirements will be prioritize mathematically? It has only a list of prioritization techniques. In [5] C. Kuloor and A. Eberlrin have explained the requirements engineering for software product lines. It has limited number of elicitation techniques. This paper does not include ontology framework, misuse cases, rapid application development etc. In [12] J.Karlsson, C Wohlin, and B.Regnell have evaluated six different methods for prioritization software requirements. In this paper, authors have found that Analytic Hierarchy Process to be the most promising method. But in literature we have found the some weaknesses of the AHP. The limitation of the AHP is that it only works because the matrices are all of the same mathematical form-known as positive reciprocal matrix. To create such a matrix requires that, if we use the number 9 to represent A is absolutely more important than B then we have to use 1/9 to define the relative importance of B with respect to A. Some people regard that as reasonable: others are less happy about it. This paper does not include any elicitation technique. We know that without eliciting any requirements we can not prioritize it. So in order to prioritize the requirements, there should be a list of elicitation techniques. Researchers, scientist and academician in the field of software engineering have proposed several techniques to elicit the software requirements. In [19] authors have proposed an approach for the software requirements elicitation. They have used the several steps like training sessions to eliminate "lack of user input" and "poor understanding", recording keywords, pictorial representation of needs and wants to reduce language barriers etc. but this approach does not have the information that how we will prioritize the requirements? In [1] the authors have provided the different elicitation technique and criteria for its selection. In [15] the authors have proposed a framework to elicit and prioritize the software requirements using AHP and QFD [16, 17, and 22] but this framework does not rank the requirements by the relative level of threat. In [23] the authors have presented an approach for requirements prioritization using B tree. In this paper the authors have mentioned that AHP is most promising method, although it may be problematic to scale up and they have also discussed that AHP are not useful for project that have large number of requirements. They have included AHP, Hierarchical AHP, spanning tree matrix, bubble sort, binary search tree, priority groups, and B tree in the same category. But with out having any data we can not prioritize anything. So AHP is a technique which is used to find out the importance weight of the requirements, after applying the AHP on the given set of requirements, we can use spanning tree matrix, bubble sort, binary search tree, priority groups, and B tree. It means we have to divide the given approaches into 2 groups. In the first group we have considered only AHP and Hierarchical AHP, and in the second category we will have to consider the spanning tree matrix, bubble sort, binary search tree, priority groups, and B tree. In the continuation of the earlier work we have proposed a framework that will elicit the software requirements and also prioritize it and also rank the requirements by the relative level of threat associated with each requirement.

3. PROPOSED FRAMEWORK

3.1 In this section we have proposed a framework that will rank the requirements by the relative level of threat associated with each requirement. This framework overcomes the problems that we had discussed in the last section.

1. Elicit the software requirements with the help of the following
   1.1 Collect information about user expectations.
   1.2) Train the Clients, Users and Managers.
   1.3 Write the description of the user need for the proposed system.
   1.4 Now you can apply Misuse cases or JAD or RAD, or Ontology framework.
2. In this framework we are using AHP technique for prioritization.
   For using AHP
   {
   Create the overall performance matrix
   }
   Then calculate the Eigen vector (Importance Weight)
3. Find out the risk associated with each requirement.
4. Compare the values of the importance weight of software requirements with step 3 and then rank or prioritize the requirements.

(Proposed Framework)

1 Elicit the Software Requirements with the help of the following [15]:

1.1 Collect information about user expectations: Software Requirement Specification is the first step in the software development which is used to capture the requirement of the client. Before the designing phase SRS team write the user manual i.e. SRS and from this SRS we collect the information about the user need and expectations. This careful compilation of information will be used in the next phase to train the clients/ user and make them aware of what they can and can not expect from the software developers. In this stage stakeholder also learn about the limitations of the computer resources and functionalities, and availability of other resources.

1.2 Train the Clients, Users, and Managers: Once we have collected the information about the user need and expectation; the next step is to train the clients, users and managers. At this stage, missing user input can be supplemented.

1.3 Write the description of the user need for the proposed system: After the successful completion of the above steps, each stakeholder will write the description of his/ her needs of the system that the clients want to develop. Since the clients and customers are already educated about the computer limitations and availability of resources through the training sessions. In this stage expectations of the development process become clearer.

3.2. Analytic Hierarchy Process: In this paper we have used Analytic Hierarchy Process. The Analytic Hierarchy Process (AHP**)** is a structured technique for dealing with complex decisions. Rather than prescribing a "correct" decision, the AHP helps the decision makers find the one that best suits their needs and their understanding of the problem. Based on mathematics and psychology, it was developed by Thomas L. Saaty [20] in the 1970s and has been extensively studied and refined since then. The AHP provides a comprehensive and rational framework for structuring a decision problem, for representing and quantifying its elements, for relating those elements to overall goals, and for evaluating alternative solutions. It is used around the world in a wide variety of decision situations, in fields such as government, business, industry, healthcare, and education. In this section we have first explain how the AHP would be used to prioritize the software requirements. Suppose a university wishes to buy a piece of software of certain type and has four aspects in mind which will govern its purchasing choice. (i) Expense, E (ii) Operability, O (iii) Reliability, R (iv)

Adaptability for other uses or Flexibility , F. Competing manufactures of that equipment have offered three options, X, Y, and Z . The software engineers have looked at these options and decided that X is cheap and easy to operate but is very reliable and could not easily be adapted to other users. Y is somewhat more expensive, is reasonable and easy to operate, and is very reliable and not very adaptable. Finally, Z is very expensive, not easy to operate, is a little less reliable than Y but is claimed by the manufacturer to have a wide range of alternatives uses. Each of X, Y, and Z will satisfy the firm's requirements to differing extents so which, overall, best meets this firm's needs? This is clearly an important and common class of problem and AHP have numerous applications. We first provide an initial matrix for the firm's pair wise comparisons in which the principal diagonal contains entries of 1, as each factor is as important as itself [16].

Table-1

	E	O	R	F
E	1			
O		1		
R			1	
F				1

There is no standard way to make the pair wise comparison but let us suppose that the firm decide that O is slightly more important than cost. In the next matrix that is rated as 3 in the cell O, E and I/3 in E, O. They also decide that reliability is far more important than cost, giving 9 in R, E and 1/9 in E, R. Similarly we enter the information into the given matrix on the basis of the Saaty Rating scale. This forms the completed matrix, which we will term the Overall Preference matrix (OPM) is

Table- 2

	E	O	R	F
E	1	1/3	1/9	1/5
O	3	1	1	1
R	9	1	1	3
F	5	1	1/3	1

The eigenvector (importance weight) of the relative importance or value of E, O, R, and F is (0.058, 0.2620, 0. 454, and 0.226). Thus R is most valuable, O and F are behind, but roughly equal and E is very much less significant. So in this way we can easily prioritize the customer's requirements.

3.3. Software Risk: In the previous section we have explained how AHP would be used to prioritize the software requirements. Risk management is a process that is systematically and continuous and it can be best described by the SEI risk management paradigm. There are six paradigm of risk management: (i) Identify (ii) Analyze (iii) Plan (iv) Track (v) Control and (vi) Communication. Risk management is a discipline for living with the possibility that future events may cause adverse effects. Risk management partly means reducing uncertainty **[7].** Reducing uncertainty has a cost associated with it. We need to balance such costs we could incur if the risk is not addressed. It may not be cost effective to reduce uncertainty too much. Risk management standard is the result of work by a team drawn from the major risk management organization. It is a central part of any organizations strategic management. It is the process whereby organizations methodically address the risk attaching to their activities with the goal of achieving sustained benefit within each activity and across the portfolio of all the activities .There are three dimension of software risk. (i) Technical risk (ii) Organizational risk (iii) Environmental risk. The technical dimension results from uncertainty in the task and procedure. The organizational dimension results from poor communication and organizational structure. The environmental dimension results from rapidly changing environment and problems with external relationship with software developers and/or users.

Suppose in a software project we identified three different types of risk i.e. products recall situation, significant product rejection and competitive strike. The information about probability of risks occurring and the total loss if it occurs are given in the table-3.

Table-3

Risk	Probability of Occurring	Total loss if it occurs
Product recall situation	2%	80K
Significant product rejection	0.1%	1000K
Competitive Strike	10%	25K

The rank of risk is estimated using risk exposure and the value of the highest risk exposure indicate the most serious risk. Table-4 contains the calculated values of risk exposure and the ranking of risk.

Table-4

Risk	Probability of Occurring	Total loss if it occurs	RE	RP
Product recall situation	2%	80K	1600	II
Significant product rejection	0.1%	1000K	1000	III
Competitive Strike	10%	25K	2500	I

In the above table RE is the Risk Exposure and RP indicates the risk priority. We have explained the step 4 in section 4.

4. EXPERIMENTAL WORK:

AHP provides a good quantitative basis for making a decision about the relative priority of a given set of requirements. However it does not factor in the additional dimensions of the risks. In order to that all risks associated with each requirement have to be identified and assessed for likelihood and impact [24]. Consider the following overall performance matrix (OPM) that is derived from customer needs statement for MSNI i.e. a

Mini Software for Numerical Integration. We have elicited the software requirements for MSNI and the important list of software requirements are given below. (a) Total numbers of existing algorithm like Simpson's 1/3 rule, Simpson's 3/8 and Trapezoidal rule. (EA); (b) Total numbers of algorithms proposed by the researchers (PA); (c) Accuracy (AC); (d) Graphical User Interface (GI); (e) Functionality of the algorithm (FA); (f) Passwords (PW) (g) Risk (RI); (h) Number of inquiries (NI).

Table-5 (OPM)

C		PA	AC	EA	FA	GI	RI	PW	NI
1	PA	1	1/5	1/3	1/7	3	3	1/5	1/3
2	AC	5	1	1/5	1/9	3	3	1/3	1/5
3	EA	3	5	1	1/7	3	3	3	5
4	FA	7	9	7	1	9	1/5	1/3	5
5	GI	1/3	1/3	1/3	1/9	1	5	3	1/5
6	RI	1/3	1/3	1/3	5	1/5	1	3	5
7	PW	5	3	1/3	3	1/3	1/3	1	5
8	NI	3	5	1/5	1/5	5	1/5	1/5	1

After applying the AHP we have got the importance (I) weights and it is summarized in table- 6.

Table-6

Category	I. Weight
1	0.0569
2	0.0674
3	0.221
4	0.2667
5	0.0627
6	0.100
7	0.141
8	0.083

In the above table functionality of the algorithm has the highest priority because its importance weight is 0.2667 i. e. category 4. As we have already discussed that there are several techniques that are used to prioritize the software requirements like Spanning tree, Binary search tree, B trees etc. According to [23] B tree is better than the remaining techniques so we have used B tree in order to prioritize the value distribution and cost distribution of the requirements. According to figure 1, three most valuable requirements are SR-3, SR-4 and SR-7. The three least valuable requirements are SR-1, SR-2, and SR-5. Figure – 1 and figure-2 shows the value distribution and Cost distribution of the requirements respectively.

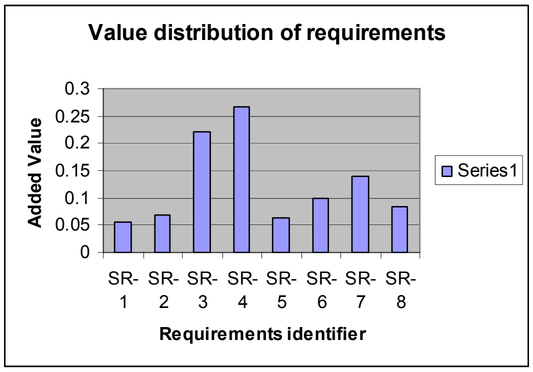

Figure 1. Value distribution of requirements

Figure-2 shows that requirements SR-2, SR-5, and SR-8 are three most expensive and the three least expensive requirements are SR-1, SR-3, SR-6

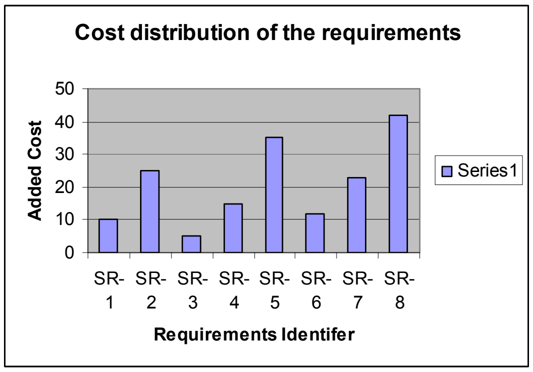

Figure 2. Cost distribution of requirements

From Figure-1 and Figure-2 we conclude, in terms of Cost-Value ratio distribution that SR-1, SR-3, SR-4, SR-6 are identified as high priority and SR-2, SR-5, and SR-8 are identified as low priority**.** In our study we have computed the value of the RE for each requirement. Figure-3 represents the comparative threat levels per requirements. Referring to the result of AHP ranking it is possible to think about these requirements in a different way. SR-1 which was identified as high priority is also a high risk. SR-3 and SR-6 which were identified as high priority are shown only to represent the moderate risk. SR-4 which is high priority requirements is a low risk.

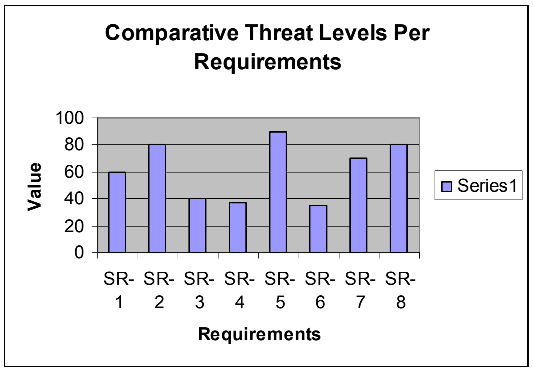

Figure-3

With the help of above result, decision makers want to revisit or perhaps re-prioritize on potential risk. It suggests that SR-1 which is approaching high risk must be revisited. We can Prioritize SR-3, SR-4, and SR-6 based on their threat index and due o high risk we can drop SR-2, SR-5 and SR-6, if resources constraint arises.

5. CONCLUSION

In this paper we have proposed a framework to elicit and prioritize the software requirements. In this paper we have shown that AHP is used only to evaluate the importance weight of the requirements not to prioritize the requirements, after this we apply the existing prioritizing techniques. We have used B tree in order to prioritize the software requirements. After this we have add the threat level analysis during the requirements prioritization. After adding threat with the requirements we conclude that prioritizations through AHP results are not only suitable for decision makers, they must think about the prioritization of the requirements using threat level analysis. This analysis will give the clear picture that how many requirements should be neglected and how many should be included, and it will also give the picture that how many requirements would be re-prioritize.

6. ACKNOWLEDGEMENTS:

The authors would like to thank *Mr. Iqbal Azam*, Principal, University Polytechnic, Faculty of Engineering and Technology, Jamia Millia Islamia (A Central University), New Delhi-25, India ; and *Mr. Jawad Ahmad Siddiqui*, Chairman, Al-Falah School of Engineering and Technology, Dhauj, Faridabad, Haryana, India, for his valuable support , guidance and encouragement

REFERENCES:

1. A.M. Hickey, A.M. Davis, "Elicitation Technique Selection: How Do Experts Do It?" Proceedings of the 11th IEEE International Requirements Engineering Conference, 2003.
2. Ann M. Hickey, Alan M. Davis, "Requirements Elicitation and Elicitation technique selection: A Model for Two knowledge-Intensive Software development Process", Proceedings of the 36th IEEE International Conference on System Sciences, 2002.
3. Beichter F. et al, " SLAN-4-A Software Specification and Design Language", IEEE Transaction on Software Engineering, SE- 10,2, 1994, pp 155-162.
4. Bruce White, "QFD for small business", Transaction from the 18 Symposium on QFD, 2006.
5. C.Kuloor, Armin Eberlein, "Requirements Engineering for Software Product Lines", The University of Calgary, Canada.
6. D. Firesmith, " Prioritizing Requirements", Journal of Object Technology, Volume 3, No.8, September 2004
7. Daya Gupta, Mohd Sadiq, "Software Risk Assessment and Estimation Model", International Conference on Computer Science and Information Technology, IEEE Computer Society, Singapore, 2008. pp 963-967
8. Gunnar Peterson, John Steven, "Defining Misuse within the Development Process", IEEE Security and Privacy, 2006.
9. http://en.wikipedia.org/wiki/Analytic_Hierarchy_Process
10. Ian Alexander, "Misuse Cases Help to Elicit Non-functional Requirements", Computing and Control Engineering 2003.
11. J. Karlsson, "Software Requirements Prioritizing", Proceedings of the International Conference on Requirement Engineering, 1996.
12. J. Karlsson, C. Wohlin, B. Regnell, "An Evaluation of Methods for Prioritizing Software Requirements", Elsvier Journal of Information and Software Technology, 1998, pp. 939-947.
13. J.J.Pauli, D.Xu, "Misuse Case-Based design and Analysis of secure Software Architecture", Proceedings of the IEEE International Conference on Information Technology: Coding and Computing (ITCC05), 2005.
14. LI Zong-yong, WANG Zhi-xue, YANG-ying, WU Yue, LIU Ying, " Towards multiple ontology Framework for Requirements Elicitation and Reuse", 31st IEEE Annual International Computer Software and Application Conference, 2007.
15. Mohd. Sadiq, Mohd. Shahid, "Elicitation and Prioritization of Software requirements", International Journal of Recent Trends in Engineering, Finland, 2009.
16. Mohd. Sadiq, Shabina Ghafir, Mohd. Shahid, "An Approach for Eliciting Software Requirements and its Prioritization using Analytic Hierarchy Process", **IEEE** International Conference on Advances in Recent Technologies in Communication and Computing, 2009, ACEEE annual world congress on Engineering and Technology , Kerala, India**.**
17. Mohd. Sadiq, Shabina Ghafir, Mohd. Shahid," A Framework to Prioritize the software Requirements using Quality Function Deployment", National Conference on Recent Development in Computing and its Application, 2009, organized by Jamia Hamdard, Delhi, India.
18. Nancy R. Mead, "Requirements Elicitation Introduction", Software Engineering Institute Carnegie Mellon University, 2008-2009.
19. P.Rajagopal, R.Lee, Thomas Ahlswede, Chia-Chu Chiang, D. Karolak, " A New Approach for Software Requirements Elicitation", Proceedings of the 6th IEEE International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/ Distributed Computing, 2005.
20. T. L. Saaty, "The Analytic Hierarchy process", New York, McGraw-Hill, 1980.
21. W.R. Friedrich, J. A. Van der Poll, "Towards a Methodology to Elicit Tacit Domain knowledge from Users", Interdisciplinary Journal of Information, Knowledge, and Management, Volume2, 2007.
22. Xiaoqing frank Liu, "Software Quality Development", IEEE Potentials, 2008.
23. Md. Rizwan Beg, Qamar Abbas, Ravi Prakash Verma, " An approach for Requirements Prioritization using B-Tree", IEEE First International Conference on Emerging Trends in Engineering and Technology 2008.
24. Nancy R. Mead, Dan Shoemaker, Jeffrey Ingalsbe, " Ensuring Cost Efficient and Secure Software through Student Case Studies in Risk and Requirements Prioritization", IEEE Proceedings of the 42 Hawaii International Conference on System Sciences-2009.