A Review of Requirement Prioritization Techniques in Agile Software Development

Najia Saher[1], Fauziah Baharom[2] and Rohaida Romli[2]
[1]*Islamia University of Bahawalpur Punjab, Pakistan, {najiasaher@gmail.com}*
[2]*Universiti Utara Malaysia, Malaysia, {fauziah@uum.edu.my, aida@uum.edu.my}*

ABSTRACT

Prioritization is a crucial process in Requirement Change Management (RCM), as erroneous requirements prioritisation may increase the cost of development and lead to project failures. In Agile Software Development (ASD), requirement prioritization (RP) is difficult to maintain and requires more formal process. Changes in the priority list leads towards rework. Moreover, requirement prioritization in Agile is a difficult task due to its volatile nature. Ignorance of critical requirements during prioritization will result in numerous problems like poor quality of product and unsatisfied client. This paper presents a comprehensive review of RP by reviewing the strengths and weaknesses of existing RP techniques. This paper also provides information related to the current state-of-the-art on techniques and practices of RP and the research gaps in related works. These findings will contribute as inputs to construct a framework for selecting suitable RP techniques in ASD which can help software practitioners in choosing suitable prioritization techniques for handling continuous requirement change in ASD.

Keywords: Agile software development, requirement change management, requirements prioritization.

I INTRODUCTION

Requirement prioritization is important in Agile software development. Berander and Andrews (2005) define requirement prioritization as: "Most software projects have more candidate requirements than can be realized within the time and cost constraints. Prioritization helps to identify the most valuable requirements from this set by distinguishing the critical few from the trivial many". According to Racheva et al. (2008), the following criteria should be considered in choosing the prioritization approach: (1) number of stakeholders involved, (2) number of items to be prioritized, (3) sources of information available and (4) level of requirements volatility. The process of requirement prioritization in ASD is quite different from traditional software development approaches. In ASD, customer satisfaction is on high priority throughout the lifecycle and requirements can change even late in the development (Fowler & Highsmith, 2001). Requirements are prioritized before each iteration, developers and customers identify new requirements and reallocate priorities on the basis of customer needs (Racheva et al., 2008). If a requirement is less important, it will be kept on hold for the next iteration (Sillitti & Succi, 2005).

From customer's perspective, continuous requirements reprioritization is an essential activity of Agile approaches. However if the continuous requirement reprioritization practiced without caution, it will lead the project towards instability (Racheva et al., 2008). According to Cao and Ramesh (2008), the main differences between traditional and Agile RE regarding prioritization are: (1) requirements are typically prioritized once in traditional RE however in Agile, requirements are prioritized in every development lifecycle, and (2) in traditional RE, many factors e.g. risks, business value, cost, and implementation dependencies drive requirements prioritization whereas, in Agile RE prioritization is based on single factor of business value as defined by customer.

There are several prioritization techniques identified in literature, nonetheless most of the RP techniques still facing challenges such as complexity, scalability, uncertainty, and time consumption (Achimugu, Selamat, Ibrahim, & Mahrin, 2014).

Hence, this paper aims to discuss on strengths and weaknesses of the existing RP techniques of ASD. The paper is organised as follows: Section II provides an overview of requirements prioritisation process in ASD. Section III describes the well-known prioritization techniques with its strengths and weaknesses. Section IV makes an evaluation of the existing technique by making a comparison on different aspects that is followed by the discussion in section V. Finally, Section VI concludes the paper with a summary of the main findings and future work.

II OVERVIEW OF REQUIREMENTS PRIORITISATION IN ASD

As mention earlier, requirement prioritization in ASD will be carried out in every iteration. Thus, it is critical to perform the process effectively to ensure a successful of the project. Due to the nature of welcoming changes in requirement frequently, that leads towards a lot of reworks on prioritization (Petersen & Wohlin, 2009). Moreover, in ASD only a single dimension of business value has been taken into consideration while doing requirement prioritization (Heikkila, Damian, Lassenius, & Paasivaara, 2015; Ramesh, Cao, & Baskerville, 2010). There are several studies that focused on requirement prioritization in ASD.

Hoff et al. (2008) suggested several core values that stakeholders have considered in terms of relevancy and importance that increased the insights of project planners to choose which requirements should be prioritized higher. These core values increase the overall stakeholder satisfaction by reducing project risk. The core values as proposed by Hoff et al. (2008) include these requirement prioritization decision factors: (1) cost-benefit to the organization, (2) fixes errors, (3) complexity, (4) requirement dependencies, and (4) delivery date/schedule. All these elements must be considered when determining the value of the requirements, prioritizing them and selecting them for future releases.

Furthermore, Racheva et al. (2008) review the Agile Requirement Prioritization (RP) methods, identify issues and challenges in Agile RP and develop a conceptual model from client perspective for inter-iteration prioritization. They identified 15 methods used in Agile prioritization with their core idea and context of use such as (1) Round-the-group prioritization, (2) Ping Pong Balls, (3) $100 allocation, (4) Multi-voting system, (5) MoSCoW, (6) Pair-wise analysis, (7) Weighted criteria analysis, (8) Analytic Hierarchy Process (AHP), (9) Dot voting, (10) Binary Search Tree (BST), (11) Ranking based on product definition, (12) Planning Game, (13) Quality functional deployment (QFD), (14) Wiegers' matrix approach, and (15) Mathematical programming techniques for release planning.

Subsequently, in 2010 Racheva et al. (2010) present two conceptual models by reviewing the approaches of RP and suggest that there are five aspects that the client should consider when making decision on requirements prioritization i.e. (1) effort estimation and size measurement, (2) business value, (3) learning experience, (4) risk, and (5) external change.

Further, Popli et al. (2014) discussed the limitation of validate learning, Moscow method, walking skeleton and business value based methods that are not efficient due to the lack of reflection on the importance of user's stories provided by customers. They consider *importance* related and *efforts* related factors during prioritization and calculated the importance and effort ratio to decide the priority of user-stories.

By concluding the above mentioned studies, it has been observed that different conceptual models and frameworks have been proposed in literature using multiple techniques of RP in Agile without any consensus.

III REQUIREMENT PRIORITIZATION TECHNIQUES

From the existing literature, some prioritization techniques with their respective strengths and weaknesses are mentioned in Table 1.

Table 1. Requirement Prioritization Techniques in Agile Software Development.

RP Techniques	Description	Strength	Weakness
Analytic hierarchy process (AHP)	AHP technique calculates the relative importance of each requirement by using pair-wise comparison matrix (Rida, Nazir, Tabassum, & Asim, 2017); (Khan, Rehman, Hayat Khan, Javed Khan, & Rashid, 2015); (Achimugu, Selamat, Ibrahim, & Mahrin, 2014).	• Ability to resolve conflicting objectives. • Provide reliable result (Achimugu, Selamat, Ibrahim, & Mahrin, 2014).	• Time consuming at higher number of requirements (Duan et al., 2009). • Not scalable so problematic for larger project (Karlsson & Ryan, 1997).

Binary Search Tree (BST)	BST ranks requirement in a hierarchical order of parent-child relationship. First it analyses all the elicited requirements and then rank it (Kaur & Bawa, 2013); (Duan et al., 2009); (Achimugu et al., 2014).	• BST could easily scale up to thousands of requirements, and still be a very fast candidate (Kaur & Bawa, 2013).	• BST does not assign any priority values rather only a simple ranking of requirements (Duan et al., 2009), • BST shows which requirement is more favourable but the extent to which the requirement is important cannot be known and therefore the comparison is just ordinal (Kaur & Bawa, 2013).
Cost-value ranking	CV assesses cost and value of each requirement from an implementation perspective. Cost value ranking prioritize requirements based on implementation cost and their perceived value (Thakurta, 2013); (Achimugu et al., 2014).	• Capability to combine the judgments of both cost and value of requirements for implementation (Karlsson & Ryan, 1997),.	• Time consuming and un-scalable (Karlsson & Ryan, 1997). • Requirements computational complexity increases in managing interdependencies as the number of requirements increases (Thakurta, 2013).
Cumulative Voting	Cost value is a ratio-scale RP technique. Customers and stakeholders are given a fixed number of units which are used for prioritization of requirements by giving vote to the requirements on the basis of customers or stakeholder's preference (Achimugu et al., 2014); (Racheva et al., 2010); (Rida et al., 2017).	• Simplicity of the approach (Hatton, 2008).	• Not suitable for large number of requirements (Berander & Andrews, 2005). • Does not permit evaluation of the relative priority difference among the requirements (Thakurta, 2013).
Kano Model	Kano is a comparison of customer satisfaction. Kano model classifies it into five categories; must be quality, satisfaction quality, attractive quality, indifferent quality and reverse quality (Rida et al., 2017); (Achimugu et al., 2014); (Racheva et al., 2010); (Cohn, 2006).	• Kano is more concerned to the customer preferences for customer "Trustworthiness". • Kano method is the fastest way to prioritize requirements (Fehlmann, 2008).	• It can only be used for analysing the effects. • It is not for suggesting new product features, something that is quite difficult to achieve.
MoSCoW	Requirements prioritization on the basis of the most immediate business benefits early. MoSCoW stands for: M: must have requirement S: should have requirements, C: could have requirements, W: won't have requirements. "M" being the highest and "W" being the lowest (Waters, 2009); (Achimugu et al., 2014); (Racheva et al., 2010).	• It is consistent, less difficult, less effort required and able to handle large number of alternative (Hatton, 2008). • Easily scalable, as it is suitable for both small and large numbers of requirements (Hatton, 2008).	• The problem comes with its lack of grading within categories. It is difficult to know which *SHOULD* or *COULD* requirements are more important than others. Better suited to product with less customers (Zacarias, 2016).
Planning Game	In Planning game clients categorize requirements into the categories of essential, conditional and optional. The complete process is based on two criteria: technical risk determines by the developers and the business value determine by the clients (Duan et al., 2009); (Achimugu et al., 2014).	• Planning game has a • better modification of numerical computation. • Easy and Fast to complete the prioritization process (Ahl, 2005).	• Problematic with large number of requirements (Duan et al., 2009).
Pair wise analysis	Requirements are compared in pairs to rank till the top requirements appear at the top of the stack. Pairwise comparison of requirements is based on the importance (Achimugu et al., 2014).	• Criteria for comparing options can remain informal, thereby basing judgments on participants' experiences (Thakurta, 2013).	• Tedious, complicated and provide unreliable results. • Ignores level of detail or sophistication of a multi-criteria analysis. • Limitation in scalability (Thakurta, 2013).

Quality Functional Deployment (QFD)	QFD technique is based on matrices where the client's expectations are represented in chronological order to determine how these expectations are to be met by the developers. Matric is created as the "house of quality," which reflects both what (customer needs) and how (designer needs) (Achimugu et al., 2014); (Crow, 2009), (Fehlmann, 2008); (Raharjo, Xie, & Brombacher, 2006).	•	QFD is a structured methodology for customer needs in the form of "voice of the customer".	•	Mainly applied in small systems. • Limitation in inconsistencies and scalability.
Value-oriented prioritization (VOP)	VOP is based on the stakeholder ratings by linking them to identified business values. Requirements are prioritized according to their impact on organization recognized business values (Azar, Smith, & Cordes, 2007); (Achimugu et al., 2014); (Rida et al., 2017).	•	Organization business value is taken into consideration in the prioritization process (Thakurta, 2013).	•	Ignores requirement dependencies (Thakurta, 2013). • Not suitable for larger project.

IV EVALUATION ON REQUIREMENT PRIORITIZATION TECHNIQUES

The detailed findings of requirement prioritization in Table 2 are comprehensively discussed and evaluated in this section. On the basis of overview of some common prioritization techniques and approaches, the comparison of requirement prioritization techniques with different aspects is presented in Table 2.

Table 2. Comparison of Requirement Prioritization Techniques using Different Factors

Prioritization Techniques	Scalability	Complexity	Scale	Time Consumed	Aspect
Analytic hierarchy process (AHP)	Not scalable	Very complex	Ratio	Very slow	Strategic Importance, Penalty
Binary Search Tree (BST)	Scalable	Easy	Ordinal	Slow	------
Cost-value ranking	Not scalable	Complex	Ordinal	Fast	Customer importance
Cumulative Voting	Not scalable	Easy	Ratio	Fast	Customer importance
Kano Model	Not scalable	Easy	------	Fast	Customer preference
MoSCoW	Scalable	Very Easy	Nominal	Fast	Business benefit
Planning Game	Not scalable	Easy	Ordinal	Fast	Business value, Technical risk
Pair wise analysis	Not scalable	Complex	Ordinal	Fast	Judgments on participants experiences
Quality Functional Deployment (QFD)	Not scalable	Complex	Ordinal	Slow	Voice of the customer
Value-oriented prioritization (VOP)	Not scalable	Complex	Ratio	Fast	Business value

Referring to Table 2, scalability, complexity, time consumed and aspects are important criterion/factors for evaluating RP techniques. It can be categorized as scalable to not scalable in term of the number of requirements, easy to complex in term of the complexity, fast to very slow in term of the time consumed. Furthermore, the key factors of prioritization process are importance, cost, risk, penalty, time, volatility,

strategic importance, customer importance and business value.

On the basis of these comparisons, practitioner will be able to choose the most suitable prioritization technique(s) on the basis of different aspects such as customer importance, customer preference, strategic importance, judgments on participant's experiences, business value, risk and penalty.

V DISCUSSION

Generally, this paper has investigated several issues regarding RP techniques. From the literature review, most of the prioritization techniques still

suffer from different limitations such as complexity, uncertainty, scalability, and time consumption. Some of the techniques are successful in small projects, but do not scale well when the number of requirements increase in large projects as mentioned in Table 2.

The analysis of RP techniques as depicted in Table 1 shows that the techniques like analytical hierarchy process (AHP), planning game (PG), cumulative voting (CV), binary search tree (BST), quality functional deployment (QFD) and cost value approach (CVA) are the most used and well-known techniques. Furthermore, the techniques like binary search tree, kano model, MoSCow, value-oriented prioritization (VOP) have also gained adequate attention (Achimugu et al., 2014). Moreover, the techniques like AHP, cost value ranking, Kano model, PG, QFD, VOP and pairwise comparisons lacking the capacity of scalability. These are suitable for small number of requirements. However, as requirements grow over the time, effort in contrast grows proportionally. AHP is the widely used and cited technique as compared to others techniques.

According to the Karlsson (1998), AHP provides most reliable prioritization results due to its ability to compute consistency ratios across requirements. However, many researchers determined that, AHP lacking of the ability to scalability (Achimugu et al., 2014; Khan et al., 2015; Rida et al., 2017). This is due to the reason that AHP makes a pairs wise comparison of requirements that becomes difficult as the number of requirements increases. The value-oriented prioritization as proposed by Azar et al. (2007) in which, requirements are prioritized on the basis of business value.

Generally the existing prioritization techniques are time consuming in the real scenario (Karlsson, Wohlin, & Regnell, 1998). Among different prioritization technique, AHP is the slowest as well as BST (Khan et al., 2015) and QFD, whereas CV, Kano, MoSCow, PG, VOP are fast in execution. An experiment done by Ahl (2005) showed that PG is the fastest while AHP was the slowest. In term of the complexity MoSCoW, PG, Kano are the easiest methods, followed by BST and cumulative voting methods (Fehlmann, 2008; Khan et al., 2015). AHP is considered to be the most challenging method amongst the selected prioritization techniques. In term of the accuracy and reliability cumulative voting, BST and AHP are the most accurate methods (Achimugu et al., 2014; Khan et al., 2015).

Hence, on the basis of the 10 prioritization technique chosen in Table 1 with their strength and weaknesses, the comparison of requirement prioritization techniques using different factors are extracted in Table 2.

Therefore, the main issue while adopting these techniques is deciding as: when and how to use which technique/approach. The purpose of making this comparison is to make prioritization process more explicit, objective and systematic, and to make an efficient decision after the requirement change management in Agile Software Development.

VI CONCLUSION

This paper has provided information about the current state-of-the-art techniques and practices for requirement prioritization and the research gaps in existing works. The strengths, weaknesses and comparison of well-known prioritization techniques for ASD are identified. This evaluation will be helpful for the practitioners in making dynamic decision after the requirement change in ASD. Better decisions of requirement prioritization will lead to a better planning which will increase the chance of project success. Future work will include the complete guideline for the selection of suitable prioritization technique(s) after the process of requirement change in ASD.

REFERENCES

Achimugu, P., Selamat, A., Ibrahim, R., & Mahrin, M. N. (2014). A systematic literature review of software requirements prioritization research. *Information and Software Technology*, *56*(6), 568–585. https://doi.org/10.1016/j.infsof.2014.02.001

Ahl, V. (2005). An experimental comparison of five prioritization methods : Investigating ease of use, accuracy and scalability. Retrieved from http://www.diva-portal.org/smash/record.jsf?pid=diva2%3A833611&dswid=-1024#sthash.LS4e9Nrm.dpbs

Azar, J., Smith, R., & Cordes, D. (2007). Value-Oriented Requirements Prioritization in a Small Development Organization. *IEEE Software*, *24*(1), 32–37. https://doi.org/10.1109/MS.2007.30

Berander, P., & Andrews, A. (2005). Requirements Prioritization. In *Engineering and Managing Software Requirements* (pp. 69–94). Berlin/Heidelberg: Springer-Verlag. https://doi.org/10.1007/3-540-28244-0_4

Cao, L., & Ramesh, B. (2008). Agile Requirements Engineering Practices: An Empirical Study. *IEEE Software*, *25*(1), 60–67. https://doi.org/10.1109/MS.2008.1

Chatzipetrou, P., Angelis, L., Rovegard, P., & Wohlin, C. (2010). Prioritization of Issues and Requirements by Cumulative Voting: A Compositional Data Analysis Framework. In *2010 36th EUROMICRO Conference on Software Engineering and Advanced Applications* (pp. 361–370). IEEE. https://doi.org/10.1109/SEAA.2010.35

Cohn, M. (2006). *Agile estimating and planning*. Prentice Hall Professional Technical Reference.

Crow, K. (2009). Customer-focused development with QFD. *Annual Quality Congress Proceedings-American Society for Quality Control*.

Duan, C., Laurent, P., Cleland-Huang, J., & Kwiatkowski, C. (2009). Towards automated requirements prioritization and triage. *Requirements Engineering*, *14*(2), 73–89. https://doi.org/10.1007/s00766-009-0079-7

Fehlmann, T. M. (2008). New Lanchester Theory for Requirements Prioritization. In *2008 Second International Workshop on Software Product Management* (pp. 35–40). IEEE. https://doi.org/10.1109/IWSPM.2008.6

Fowler, M., & Highsmith, J. (2001). The agile manifesto. *Software Development*, *9*(August), 28–35.

Heikkila, V. T., Damian, D., Lassenius, C., & Paasivaara, M. (2015). A Mapping Study on Requirements Engineering in Agile Software Development. In *2015 41st Euromicro Conference on Software Engineering and Advanced Applications* (pp. 199–207). IEEE. https://doi.org/10.1109/SEAA.2015.70

Hoff, G., Fruhling, A., & Ward, K. (2008). Requirement prioritization decision factors for agile development environments. Retrieved from https://nebraska.pure.elsevier.com/en/publications/requirement-prioritization-decision-factors-for-agile-development

Karlsson, J., & Ryan, K. (1997). A cost-value approach for prioritizing requirements. *IEEE Software*, *14*(5), 67–74. https://doi.org/10.1109/52.605933

Karlsson, J., Wohlin, C., & Regnell, B. (1998). An evaluation of methods for prioritizing software requirements. *Information and Software Technology*, *39*(14–15), 939–947. https://doi.org/10.1016/S0950-5849(97)00053-0

Kaur, G., & Bawa, S. (2013). A Survey of Requirement Prioritization Methods. *International Journal of Engineering Research & Technology (IJERT)*, *2*(5), 958–962.

Khan, J. A., Rehman, I., Hayat Khan, Y., Javed Khan, I., & Rashid, S. (2015). Comparison of Requirement Prioritization Techniques to Find Best Prioritization Technique. *International Journal of Modern Education and Computer Science*, *7*(11), 53–59. https://doi.org/10.5815/ijmecs.2015.11.06

Olson, D. (2014). Matrix Prioritization | BAwiki. Retrieved August 10, 2017, from http://www.bawiki.com/wiki/techniques/matrix-prioritization/

Petersen, K., & Wohlin, C. (2009). A comparison of issues and advantages in agile and incremental development between state of the art and an industrial case. *Journal of Systems and Software*, *82*(9), 1479–1490. https://doi.org/10.1016/j.jss.2009.03.036

Popli, R., Chauhan, N., & Sharma, H. (2014). Prioritising user stories in agile environment. In *2014 International Conference on Issues and Challenges in Intelligent Computing Techniques (ICICT)* (pp. 515–519). IEEE. https://doi.org/10.1109/ICICICT.2014.6781336

https://doi.org/10.1177/004057368303900411

Hatton, S. (2008). Choosing the Right Prioritisation Method. In *19th Australian Conference on Software Engineering (aswec 2008)* (pp. 517–526). IEEE. https://doi.org/10.1109/ASWEC.2008.4483241

Racheva, Z., Daneva, M., & Buglione, L. (2008). Supporting the Dynamic Reprioritization of Requirements in Agile Development of Software Products. In *2008 Second International Workshop on Software Product Management* (pp. 49–58). IEEE. https://doi.org/10.1109/IWSPM.2008.7

Racheva, Z., Daneva, M., Herrmann, A., & Wieringa, R. J. (2010). A conceptual model and process for client-driven agile requirements prioritization. In *2010 Fourth International Conference on Research Challenges in Information Science (RCIS)* (pp. 287–298). IEEE. https://doi.org/10.1109/RCIS.2010.5507388

Raharjo, H., Xie, M., & Brombacher, A. C. (2006). Prioritizing quality characteristics in dynamic quality function deployment. *International Journal of Production Research*, *44*(23), 5005–5018. https://doi.org/10.1080/00207540600547414

Ramesh, B., Cao, L., & Baskerville, R. (2010). Agile requirements engineering practices and challenges: an empirical study. *Information Systems Journal*, *20*(5), 449–480. https://doi.org/10.1111/j.1365-2575.2007.00259.x

Rida, A., Nazir, S., Tabassum, A., & Asim, S. (2017). The Impact of Analytical Assessment of Requirements Prioritization Models: An Empirical Study. *International Journal of Advanced Computer Science and Applications*, *8*(2). https://doi.org/10.14569/IJACSA.2017.080240

Sillitti, A., & Succi, G. (2005). Requirements Engineering for Agile Methods. In *Engineering and Managing Software Requirements* (pp. 309–326). Berlin/Heidelberg: Springer-Verlag. https://doi.org/10.1007/3-540-28244-0_14

Thakurta, R. (2013). A framework for prioritization of quality requirements for inclusion in a software project. *Software Quality Journal*, *21*(4), 573–597. https://doi.org/10.1007/s11219-012-9188-5

Waters, K. (2009). Prioritization using MoSCoW.

Zacarias, D. (2016). 20 Product Prioritization Techniques : Retrieved July 11, 2017, from https://foldingburritos.com/product-prioritization-techniques/