Study of Tools for Managing Changing User Requirement for Software Development

[1]Pooja, [2]Kamna Solanki
[1,] *Research Scholar, M.D.University, Rohtak*
[2]*Assistant Professor, M.D.University, Rohtak*

Abstract: **Requirement change management is very critical and the most important aspect in the software development. User Requirements keep on Changing during all the stages of software development. Hence, there must be some effective techniques to tackle these changing user requirements. Basically, change is a transition from current way of working to another looked-for way and this nature of the change coupled with complexity of services create problems. This paper depicts the main hurdles in the change management like dependability, traceability etc. and efficient tools to cope these changes so that it would not affect the stability. Ultimate aim is to propose a framework to manage this important trait in the process of software development.**

Keyword: **Framework, changing requirements, requirements management.**

1. INTRODUCTION

Software is usually developed following a specific method that has some defined phases, which it goes through. Most methods have phases for analysis, design, test and implementation, after that product is released and enters the final phase of maintenance. The maintenance phase can be thought of as mini cycles of the main development phases where a new requirement (or a bug) requires a little additional analysis, a small change to the design, modifications to the implementation and updated test cases. All the products of the main cycle (specifications, design, code, test cases) are affected by changes in the maintenance phase and hence it is very important that products should be made with maintainability in mind and that Change management is able to follow changes through all phases.

Changing requirements have been considered as a challenging area of research by the software engineering community [1]. It has been observed that requirements change during different phases of software development life cycle (SDLC) [2] and this change plays a vital role in success or failure of any project [3]. The fact is that more than half of the system's requirements will change before the actual deployment of the system [4]. Most of the software failures are attributed to poor requirements engineering in which ambiguous and incomplete requirements lead to changes throughout the SDLC [5]. In recent years the trend has changed from blaming the problem towards identifying the cause of that problem [6]. There is considerable overlap and confusion between change management, change control and configuration management. The definition below does not yet integrate these areas. Change management has been embraced for its ability to deliver benefits by improving the affected system and thereby satisfying "customer needs," but has also been criticized for its potential to confuse and needlessly complicate change administration. In some cases, notably in the Information Technology domain, more funds and work are put into system maintenance (and change management) than into the initial creation of a system. Typical investment by organizations during initial implementation of large ERP systems is 15 to 20 percent of overall budget [7].

Change management is also of great importance in the field of manufacturing, which is confronted with many changes due to increasing and worldwide competition, technological advances and demanding customers. Because many systems tend to change and evolve as they are used, the problems of these industries are experienced to some degree in many others. Requirements management is the process of documenting, analyzing, tracing, prioritizing and agreeing on requirements and then controlling change and communicating to relevant stakeholders. It is a continuous process throughout a project [8]. A requirement is a capability to which a project outcome (product or service) should conform. The purpose of requirements management is to ensure that an organization documents, verifies, and meets the needs and expectations of its customers and internal or external stakeholders.

Requirements management begins with the analysis and elicitation of the objectives and constraints of the organization [9]. Requirements management further includes supporting planning for requirements, integrating requirements and the organization for working with them (attributes for requirements), as well as relationships with other information delivering against requirements, and changes for these. The traceability thus established is used in managing requirements to report back fulfilment of company and stakeholder interests in terms of compliance, completeness, coverage, and consistency. Traceability also support change management as part of requirements management

in understanding the impacts of changes through requirements or other related elements (e.g., functional impacts through relations to functional architecture), and facilitating introducing these changes. Requirements management involves communication between the project team members and stakeholders, and adjustment to requirements changes throughout the course of the project. To prevent one class of requirements from overriding another, constant communication among members of the development team is critical. For example, in software development for internal applications, the business has such strong needs that it may ignore user requirements, or believe that in creating use cases, the user requirements are being taken care of.

2.RELATED WORK

In a systematic review, the main research questions, the methodological steps, and the study retrieval strategies are explicitly defined. In 2004, the procedures for performing a Systematic Literature Review (SLR) in Software Engineering were first proposed by Kitchenham [12]. In the software engineering research literature, there are a few examples of reviews on agile methods:

(Jeffery and Paech, 2002) presented literature review on various requirements engineering process models that exist in literature. With the help of qualitative questionnaire a structured interview was conducted. The data obtained from the interview was discussed with respect to requirements engineering process at two Australian companies and an illustrative requirement engineering process model was constructed and compared with three existing requirements engineering process models [7].

(Kontio et al, 2004) identified some critical factors that affect organizations requirements engineering processes and indicated that organizations can gain benefit by basic RE practices as well as human factors such as motivation, commitment and enthusiasm [8].

(Li Jiang, 2005) proposed a Framework for Requirements Engineering Process Development. Developed model is used to build an appropriate RE process model and RE techniques for software project[9].(Ikram et al, 2006) presented a critical study of goal oriented requirement engineering techniques(GORE) that provide an incremental approach for elicitation, analysis, elaboration, refinement, specification and modeling of requirements. They evaluated the underlying concepts, process and advantages of GORE with respect to requirement engineering activities [10].
(Atlee et al, 2007) outlined the aspects of RE research with respect to requirements technologies. It also identified numerous research challenges along with research areas that call for further investigation [11]. (Aftab, 2008) explored requirements modelling in agile framework. The paper highlighted on just-in-time requirements in agile development. The framework had three main phases. First phase involve Initial Envisioning (Functionality Analysis, User Story Analysis, Architectural Analysis) while second phase is Proof of Concept Modeling Through TDD(Test Driven development) and last is Reviews. It proof that by using TDD in requirements analysis significantly reduces project risk and development time [13].
(Hasnain , 2010) conducted a systematic literature review to identify the agile practices as well as the human and technical factors pointed out in agile studies, published within 2003–2007. The review revealed that agile RE practices had only been discussed in the literature from the overall perspective of agile methods and not in the context of any particular methods such as Scrum, test-driven development, etc. Hasnain's findings suggest that more empirical results are required on agile methods, in particular XP (Extreme Programming) (Beck, 1999) and Scrum (Schwaber & Beedle, 2001), in order to discuss the details from the practitioner's point of view [15].

(Silva et al, 2011) conducted a systematic literature review on the topic of the integration of agile methods and usercentred design approaches. The review focused on usability issues in agile methods with respect to design. The findings show that usability issues in agile methods can be addressed by incorporating a user centred design specialist (UCDS) role in agile teams. The authors also defined practices to resolve usability issues in agile methods such as Little Design Up Front, Big Design Up Front, low fidelity prototypes, user testing, interaction models, and close collaboration[20].

(Helmy et al, 2012)described in detail about architecture related issues in agile requirements engineering process and proposed methodology to guide and assist practitioners adopting agile requirements engineering in the complete development process [21]. (Rizvi, 2013) conducted a systematic literature review on distributed agile software development. The review aimed to study the way in which organisations adopted distributed agile software development. In addition, the review focused on the challenges and their solutions from 2007 to 2012. Rizvi's findings revealed communication, collaboration, coordination and cultural differences as major challenges of distributed agile development. The review also emphasised the importance of having an

infrastructure for communication and collaboration to address the identified challenges [22].

(Minhas et al , 2014) improved framework for requirement change management in global software development (RCM_GSD) has been presented. The objective is to manage the change in requirement specifically in global software development in an appropriate manner. The proposed framework RCM_GSD follows the required processes of RCM and reduces the concerns of GSD. Systematic Literature Review (SLR) was conducted for exploration of relevant research [23].

3. TOOLS ASSESSMENT:

In general, a tool is a process that designed to achieve a specific purpose, especially if the item is not consumed in the process . There are different tools, frameworks and models available in the market and can be classified into a certain number of categories in order to assess and identify their weaknesses and strengths. Many project developers used these tools to manage their software change requirements management. However, these tools are consisting of a variety of processes. Some of these tools are commercial off-the-shelf software applications such as RequireIt, RequisitePro, DOORS, RTM SLATE, Ultra-lightweight and Lightweight.

TOOLS	Strengths	Weakness
Borland CaliberRM	Borland is a tool used to manage requirements . It can be divided into two important points: Caliber Define IT, which stands as software requirement at the first stage of the project. Second point is traceability links, it stores by checking which artifact are linked and its direction.	As defined, CaliberRM tool can be supported only traceability approach for controlling requirements management and change request. The change can be traced only via traceability links. There are no change implementation and verification process.
Heavyweight (IBM Rational Requisite pro)	RequisitePro is software change and requirement management tool which is under IBM's Rationale Suite .It supports to model software change and store them in a relational database.	Unfortunately, heavyweight tools are complex, inflexible and costly. Normally, tools with a lot of features are complex because it needs to train the staff very high cost and takes time to understand and how to perform impact analysis and how to use it as well.
RTM (Integrated chipware)	RTM (Requirements and Traceability Management) from Integrated Chipware is a software change and requirement control tool designed to support a large integrated software project development.	RTM does not help object-oriented properties. More specifically, when the project becomes complex it difficult to trace backward.
RequireIt (Telelogic AB)	RequireIt is a software change and Requirements Management tool, which is based entirely on MS-Word. This tool designed for novice users. it gives change to the users to utilize its existing, Familiar interface .	RequireIt tool limits to get change history and identification of a past change request approaches. Further, it ignores database administration to keep the project requirement in a traceable way.
DOORS AND DOORSNET (TELELOGIC AB)	DOORS, is a software change requirement management tool that designed to use bi-directional traceability. It also allows to change impact analysis.	This tool Provides only a single-database repository . As DOORS tool user views, the tool configuration management does not support with high project requirements churn i.e. If for example, 70% of the project requirements in a database change in a short period of time.

Table 1: Assessment of a Change requirement management tools

4. ASSESSMENT OF EXISTING MODELS

In this context, process model, defines what we are going to do (activities), who will be accountable (role) and what it should be (the input and output) [15]. These are known as the items or elements of the model [16]. These elements are the constructs of all models, which we have integrated set a framework [17]. Mostly, different researchers have provided several items and there is no consensus for their items [17], [18] but when we look such as activities, roles and artifacts which are mostly discussed in the literature. In the following Table 1 shows activities, artifacts and roles/actors.

RCM Model / **Activities**	Olsen's Model	V like Model	Ince's Model	Sprial Model
Change impact functionality		YES		
Change implemention	YES	YES	YES	
Update documeent			YES	
verification	YES			
Problem understanding		YES		YES
Solution analysis		YES		YES
Solution specification		YES		YES
Regration testing		YES		

Table 2: Existing activities in a most software change management model from literature

In V-like model [17], planning the resource on the change control is missing, as whenever there is a change request, it is necessary to estimate the effect of the change to allocate the required cost for implementation, this kind of change is practically possible. Further, impact analysis activity has not been discussed in this model; this activity is used to identify the impact of the change.

Ince's Model [17] totally ignored the type of decisions to be taken, who will decide the change, what is the strategy for the change and what should be the process of having decision, what kind of details is needed to have proper decision and how the change will look like?. Limited artifacts were discussed and still the content of the artifacts mentioned are not enough [19]. However, in this mode, decision making activity is missing and it is difficult to know whether the related requirements are needed to update in future or not and process of what approach should be used for the change implementation. There are no testing activities discussed in this model to verify and validated changes.

spiral shaped model for the management req. [24]Change. That comprises 4 cycles or steps. In very 1st round; few alterations are asked as adding new features or fixes of bugs in the existing system. Risk analysis is important phase so requires expert people. It is not beneficial for smaller projects. Spiral may go infinitely. Documentation is more as it has intermediate phases. It is costly for smaller projects[25].

In Olsen Model the changes requested by users are managed by the change manager in change management phase. Once the changes have been accepted, they are sent to the implementation stage. Requested changes are implemented at

implementation stage. With the help of testing changes can be verified. Once changes have been verified, change mangers are informed to release new changes in the software. The change is highlighted as a basic element of life cycle of a software development, in change model[26] . On the other hand activities which are required to manage change are not considered in this model.

5. FUTURE SCOPE

As the Requirement change is of most importance. Most of the project fails due to the inefficient change managements. Although the studies have been conducted, but still a lot has to be done to handle these changes effectively, the area of change management lacks research. There is a still lack of some efficient enough and effective framework, to handle these consecutive changing requirements during all phases of SDLC. Therefore, we strongly recommend to study further effective techniques to handle changes and identifying and evaluating the causes of requirements change and their relationship with each other to propose an efficient enough framework to handle requirement changes, as they are considered to be an impact factor for the success or failure of software projects.

CONCLUSION

This study set out to explore, classify and compare the causes of requirements change during software development. The major contributions of this thesis are a formal framework for the effective management of changing software requirements and new methods for treating completeness and handling inconsistency in evolving models of requirements. This research thus offers a rigorous approach to reasoning about requirements evolution and an important starting point for defining semantically well-founded methods and tools for the effective management of changing software requirements.

REFERENCES

[1] M. G. Christel and K. C. Kang, "Issues in Requirements Elicitation", Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania, Technical Report CMU/SEI-92-TR-012, 1992.

[2] R. J. Costello and D. B. Liu, "Metrics for requirements engineering," *Journal of Systems and Software*, vol. 29, no. 1, pp. 39-63, April 1995.

[3] B. Curtis, H. Krasner, and N. Iscoe, "A field study of the software design process for large systems," *Communications of the ACM*, vol. 31, no. 11, pp. 1268-1287, November 1988.

[4] B. H. C. Cheng and J. M. Atlee, "Research directions in requirements engineering," *Future of Software Engineering*, pp. 285-303, 2007.

[5] J. V. Buren and D. A. Cook, "Experiences in the adoption of requirements engineering technologies," *Journal of Defense Software Engineering*, pp. 3-10, 1998.

[6] B. W. Boehm, "Software Engineering Economics", *Upper Saddle Rive*r, NJ: Prentice Hall PTR, ch. 28, pp. 484-485, 1981.

[7] S. Martin, A. Aurum, R. Jeffery and B. Paech, "Requirements Engineering Process Models in Practice*", The Seventh Australian Workshop on Requirements Engineering: proceedings AWRE* ,Deakin University, School of Information Systems, Deakin University Melbourne, Victoria, 2002.

[8] M. Kauppinen, M. Vartiainen, J. Kontio, S. Kujala and R. Sulonen, "Implementing requirements engineering processes throughout organizations: success factors and challenges*", lsevier (Science direct) Information and Software Technology volume-* 46, issue 14, pp 937–953, 2004.

[9] L. Jiang, "A Framework For The Requirements Engineering Process Development", *Phd. Thesis Department Of Electrical And Computer Engineering Calgary*, Alberta August, 2005.

[10] S.Anwer And N. Ikram, "Goal Oriented Requirement Engineering", *A Critical Study Of Techniques, Xiii Asia Pacific Software ngineering Conference* (Apsec'06), IEEE, 2006.

[11] H.C. Betty, M. Joanne, Atlee, "Research Directions in Requirements Engineering", *IEEE*,2007.

[12] B.Kitchenham, "Procedures for undertaking Systematic Reviews", Joint Technical Report,Computer Science Department, Keele University (TR/SE-0401) and National ICT Australia Ltd (0400011T.1), July 2007.

[13] T. Aftab, "Requirement Modeling In Agile Framework*", EPHLAX, White paper*, October, 2008.

[14]B. J. Williams, J. Carver, and R. Vaughn, "Change Risk Assessment: Understanding Risks Involved in Changing Software Requirements," *in Proc. International Conference on Software Engineering Research and Practice*, Las Vegas, Nevada , 2006.

[15] E. Hasnain, "An overview of published agile studies", A systematic literature review. *In Proceedings of the national software engineering conference* (pp. 1–6), 2010.

[16]L. Luigi and G. Valetto,. "Enhancing requirements and change management through process modelling and measurement". *Requirements Engineering, Proceedings. 4th International Conference on. IEEE*, 2000.

[17] L. Hattori., D. Guerrero, J. Figueiredo, J. Brunet and J. Damasio, "On the Precision and Accuracy of Impact Analysis Techniques", in 7th *IEEE/ACIS International Conference on Computer and Information Science*, Portland, Oregon, USA, IEEE Computer Society, 2008.

[18] F. Peter, S .Watts, "Software process development and enactment, Concepts and definitions. Software Process, Continuous Software Process Improvement," *Second International Conference on the. IEEE*, 2010.

[19] B. Nejmeh and W. Riddle, "Concepts for process definition and support, Software Process Workshop, Support for the Software Process", *Proceedings of the 6th International. IEEE*, 1990.

[20]S. Silva, T. Martin, A. Maurer and M. Silveira , "User-centered design and agile methods" . *A systematic review. In Agil. Conf*, (pp. 77–86), 2011.

[21] W. Helmy, A. Kamel and O. Hegazy, "Requirements Engineering Methodology in Agile Environment",*International Journal of Computer Science Issues*, Vol. 9, Issue 5, No 3, 2012.

[22]B. Rizvi , "A systematic review of distributed agile software engineering". Alberta: Athabasca University,2013.
[23] N. Minhas, Q. Ain, Z. Islam and A. Zulfiqar, "An Improved Framework for Requirement Change Management in Global Software Development", *Journal of Software Engineering and Applications*, 7, 779-790, 2014.
[24]G. Kotonya and I. Sommerville, "Requirements Engineering: Processes and Techniques". Chichester, UK: John Wiley and Sons , 1998.
[25]S. Ferreira, J. Collofello, D. Shunk, and G. Mackulak, "Understanding the effects of requirements volatility in software engineering by using analytical modeling and software process simulation," *Journal of Systems and Software* , vol. 82, no. 10, pp. 1568-1577, 2009.
[26]D. Zowghi and N. Nurmuliani, "A study of the impact of requirements volatility on software project performance*," in Software Engineering Conference*, Ninth Asia-Pacific , 2002, pp. 3-11, 2002.

