Software Requirement Analysis: Research Challenges and Technical Approaches

Senay Tuna Demirel
NETAŞ A.Ş., 34912, Kurtköy – Pendik
Istanbul / Turkey
stuna@netas.com.tr

Resul Das
Dept. of Software Engineering, Technology Faculty, Firat Univ.,
23119, Elazig / Turkey
rdas@firat.edu.tr

***Abstract*— Requirement analysis is one of the key challenges in software development projects. Customer requirement specification and management entails various impacts to software projects and still is an improvement area on both academic and industrial fields. Models like CMMI also uncovers requirement development and management and specifies the specific goals and practices for them. In this paper, key challenges and issues of requirement management are listed with respect to a standardization activity, namely CMMI.**

Keywords—Requirement Analysis and Management; Software Delivery Life Cycle (SDLC); Agile; Waterfall; Capability Maturity Model – Integration (CMMI).

I. Introduction

In software development projects, understanding and fulfilling the customer needs from the first stage to the delivery of the project has been recognized as one of the biggest challenges among many others [1]. Misunderstanding the customer needs, wrong elicitations and assumptions cause significant impacts to the projects by means of time, quality and budget. In addition, this is the reason why requirement management is an activity based on a multidisciplinary effort that involves marketing, analysis, engineering, product design, verification, and validation.

Requirements can be classified as (functional) business and technical (non-functional) requirements [2]. A business (functional) requirement commonly specifies a behavior from an end-user perspective. It describes a user behavior or an action that is performed by the user of the system and does not constrain by any system limitations. Whereas technical requirements specify the result of the end user behaviors. And can be constrained by platform, environment, design limitations or dependencies of the system. Business requirements are general in nature, while requirements at low levels in the hierarchy are very specific[3], [4].

There are different approaches in Software Delivery Life Cycle (SDLC) methodologies that welcome the requirement changes at any stage or that try to freeze the requirements in the initial stages and tries to avoid any changes in upcoming cycles. Requirement engineering is mostly seen an effort to occur in early stages of development lifecycle [5]. In the development of small systems, this may be even true. But it seems impossible (in practice) to develop a requirement set that is accurate and stable throughout the long time intervals of in the SDLC of large and complex systems.

The two most popular SDLC methodologies, waterfall and agile, behave totally different ways in requirement analysis and management. Waterfall tries to analyze the requirements from the early beginning and does not let implementation begin until it is very well understood, documented and almost froze. Unlike waterfall, Agile methodologies accept the fact that requirements may not de detailed enough at the beginning of the project but they will evolve over time and stakeholders will gain more insight about what they need [6]. So agile welcomes the requirement changes but focuses on correctly handling them [7]. It allows customers to continuously involve in the project and evolve the requirements.

This paper aims first to understand the issues and challenges in software requirement analysis. Throughout the paper, we will try to understand the different approaches for requirement management in different SDLC methodologies and examine the standards to understand better what needs to be done for successful requirement management.

The paper is organized into five chapters. After the introduction, challenges in requirement management are identified and listed. Then standardization activity is described by referencing the CMMI model. Moreover, the paper conclude with a description of technical approaches.

II. Challenges

Requirement identification really starts with the initial stage of the projects when business needs are stated and begin to be identified. The requirements analysis phase is mentioned as `analysis and planning`[2]. The software-engineering community had focused its efforts on the problem-analysis phase, which is what many decomposition methods address [8]. During the planning phase, high-level requirements are refined and detailed into technical requirements. By building a traceability matrix, requirements are committed and mapped into the product requirements that will be implemented in the design phase. Project management and change management processes are highly interrelated with the requirements and used to

978-1-5386-3449-3/18/$31.00 ©2018 IEEE

manage and control them. In general, the Requirements Management Process covers phases, as shown in figure 1:

- Identify Requirements;
- Evaluate Requirements;
- Validate and Prioritize Requirements;
- Obtain Approval and Baseline;
- Conduct Analysis and Detailing of Requirements;
- Control Changes to Requirements.

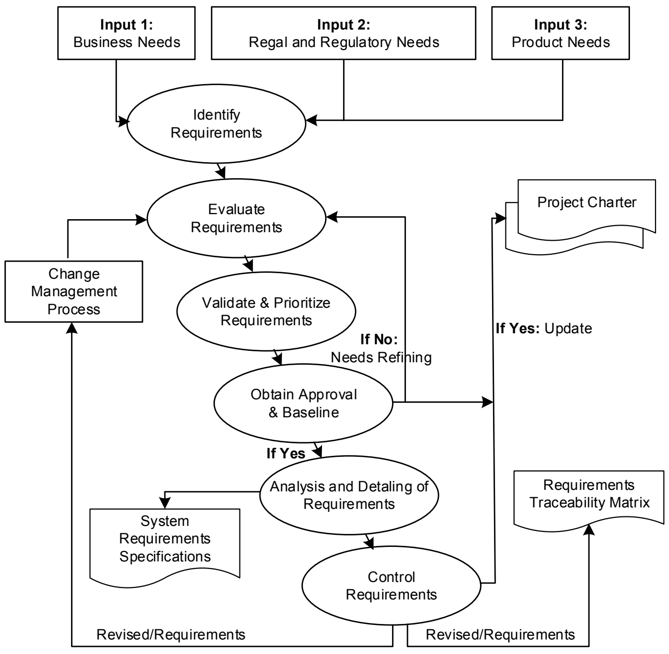
Input 1: Business Needs
Input 2: Regal and Regulatory Needs
Input 3: Product Needs
Identify Requirements
Evaluate Requirements
Project Charter
Change Management Process
Validate & Prioritize Requirements
If No: Needs Refining
If Yes: Update
Obtain Approval & Baseline
If Yes
Analysis and Detaling of Requirements
Requirements Traceability Matrix
System Requirements Specifications
Control Requirements
Revised/Requirements
Revised/Requirements

Figure 1: Requirement Management Flowchart

Regardless of how requirement management is processed (close to the steps above or far beyond) it is related to the elicitation, definition, analysis and specification stages. During these phases, there have been some challenges faced, regardless of which SDLC model you reference.

Ambiguous Requirements

Customer requirements may not be detailed enough at the beginning of the project and stakeholders have limited insight into what they really need. In most cases, customers' states the business needs but they do not have any idea of the detailed scenarios, race conditions or any conflict between different stakeholders needs. So requirements are normally underdeveloped and tend to be ambiguous. Thus, there should be an identification step on the requirement process that one of the elicitation techniques (e.g. workshops, interviews or meetings etc.) are applied. Customer needs should then analyzed and evolved to compose the business requirements (from high-level requirements)

Requirement Definitions

Frequently, each of the stakeholders use different contexts to explain and state the requirements. Differences users in terminology and semantics may weaken the ability to transfer information from customers to designers effectively. In general, customers or business people tend to use the native language terms to explain the actual need whilst the technical people want to see the more logical structure to understand the intended behavior precisely (e.g. Unified Modeling Language UML). There should be a distinguish in terms of business needs and functional requirements as well as document them. And that effort should be included in the project phases during the early stages of the product development.

Requirement Prioritization

In most cases, requirements are negotiable and may conflict with one another. Requirements should be reviewed by key stakeholders, there should be prioritization of all the requirements, and conflicting requirements should be re-evaluated. The reviewed and approved priority needs to be documented to make the outcome visible to each participant in the project team.

Requirement Changes

There may be changes in existing requirements as project evolve or new requirements may arise. Requirement changes have potential impact on project design, plans, resource allocations or schedule. Each requirement change should be tracked with proper tools/methods in order to make a risk and affect analysis first and then to decide to handle the change or not. Each stakeholder should be aware of the impacts and approve the change. Otherwise, it is impossible to track the changes and control with a plan within the project cycle.

Requirement change management has a strong relationship with baselining [9]. And requirements management is a continuous activity that can perform after development and during maintenance because requirements may continue to change [10], [11].

Traceability

Traceability is one of the main parts of requirement management, which refers ability to describe and follow the life of a requirement and its relations with other items in both ways (forwards and backward) [12], [13]. Requirements traceability will allow tracking the relation from a requirement, to a design component, to a test case and validation procedure that is related to that requirement. That relationship is critical to understand how a change (or change request) to a requirement will affect other requirements, upstream and downstream teams. It should be implemented so that, whenever any requirement is changed, all parent and child requirements, software components, and test cases impacted by the change are identified.

Technical Solution

Requirements are direct inputs for the technical solution that will be implemented in the design phase. There should be a mapping between the requirements and software components to keep an interrelation through the development lifecycle. It should be traced how an update in a software component or a reported problem in a specific subsystem, will affect the related requirements. This also helps the re-use of subsystems or work components to handle the similar requirements.

Requirements Verification and Validation

The purpose of requirements validation is to certify that the requirements are an acceptable description of the system to be implemented[14]. It should be noted that traceability from requirements to test cases are also an important area to be covered. Requirements verification consists of those activities that confirm that the product of a system development process meets its technical specifications. Requirements validation consists of activities that confirm that the behavior of a developed system meets its user needs [12]. Without verification and validation traces, it cannot be ensured that the requirements are met or not.

Project Management

Even when requirement management is handled in initial phases of the projects, traceability between requirements and project lifecycle may be getting tight in the later phases of projects. There should be a total awareness that requirement management is not only required in initial phases of the projects but a fully integrated activity throughout the delivery of the project. The correlation between the requirements and risk management, project plans, technical solution, configuration management, project integration, verification, and validation project areas should be well covered in project management overall.

III. Standardization activities

In this section, CMMI, requirement development and requirement management concepts are outleined respectively.

CMMI

Capability Maturity Model – Integration was developed by Carnegie Mellon University`s Software Engineering Institute (SEI) and is known as CMMI. It is basically a process improvement model that helps organizations to improve performance, process maturity and achieve the organizational goals[15].

CMMI for Development (CMMI-DEV): Product and service development.

CMMI for Services (CMMI-SVC)- Service establishment, management.

CMMI for Acquisition (CMMI-ACQ)- Product and service acquisition.

Throughout this paper, we focus on CMMI-DEV, which is the model used for software engineering and has two main "process areas" for requirements (and will use the term CMMI instead of CMMI-DEV). Process areas are defined as "A cluster of related practices in an area that, when implemented collectively, satisfies a set of goals considered important for making improvement in that area." in CMMI Dev v1.3 [15]. There are 21 process areas categorized into 5 maturity levels in CMMI-DEV. (The higher the rating, the higher the maturity of the organization [4].) Requirements Management is also a process area for CMMI Level 2 and Requirements Development is required for Level 3, which can be summarized as multi-stakeholder requirements evolution, as shown in Table I.

TABLE I. CMMI Process Areas [14]

Level	**Focus**	**Process Area**
5 - Optimizing	Continuous Process Improvement	- Organizational Performance Management - Casual Analysis & Resolution
4 - Quantitatively Managed	Quantitative Management	- Organizational Process Management - Quantitative Project Management
3 - Defined	Process Standardization	- Requirements Development - Technical Solutions - Product integration - Verification - Validation - Organizational Process Focus - Organizational Process Definition - Organizational Training - Integrated Project Management Risk - Management - Decision Analysis & Resolution
2 - Managed	Basic Project Management	- Requirements Management - Project Planning - Project Monitoring & Control - Supplier Agreement Management - Measurement & Analysis - Process & Products Quality Assurance - Configuration Management
1 - Initial		

Requirement Development

Requirement development (RD) is to understand and analyze the stakeholder's needs and produce customer, product and component requirements. It is critical in any

SDLC, as most of the software defects and issues are related to incorrect or misunderstood requirements. In addition, without understanding the exactly customer needs it will be impossible to have happy customers or successful project deliveries in spite of all the efforts, time or budget spent.

RD specifies three types of requirements: customer requirements, product requirements, and product component requirements. Whole together these are used to specify the needs of the stakeholders, applicable to various phases of the SDLC (e.g., verification, validation or acceptance testing) and attributes of the project (e.g., safety, reliability, and maintainability). It also specifies constraints caused by the platform dependencies or selected design solutions (e.g., open source code).
CMMI describes three specific goals for developing the requirements and provides specific practices for each goal, shown in Table II.

TABLE II. CMMI REQUIREMENT DEVELOPMENT PROCESS AREA[15]

Requirements Development – Specific Goals & Practices
SG 1 Develop Customer Requirements - "Stakeholder needs, expectations, constraints, and interlaces are collected and translated into customer requirements."
SP 1.1 Elicit Needs
SP 1.2 Transform Stakeholder Needs into Customer Requirements
SG 2 Develop Product Requirements - "Customer requirements are relined and elaborated to develop product and product component requirements."
SP 2.1 Establish Product and Product Component Requirements
SP 2.2 Allocate Product Component Requirements
SP 2.3 Identify Interface Requirements
SG 3 Analyze and Validate Requirements - "The requirements are analyzed and validated."
SP 3.1 Establish Operational Concepts and Scenarios
SP 3.2 Establish a Definition of Required Functionality and Quality Attributes
SP 3.3 Analyze Requirements SP 3.4 Analyze Requirements to Achieve Balance
SP 3.5 Validate Requirements

CMMI aims to determine the specific goals first in order to develop the requirements correctly. Then for each specific goal, it determines the specific practice to achieve the goal. The important point about CMMI is, it is not focused on "how to do" and focuses on "what to do". For example, for developing customer requirements CMMI specifies that needs should be indicated. However, does not specify how to do that. Each organization can use its own way to achieve that.

Requirement Management

Required Management(REQM) aims to handle the inconsistencies between requirements, project plans, and work products. Requirement engineering is used to gather the information, define requirements and is related to analyses and documentation of the requirements [17]. Requirements Development has the goal of setting correct requirements for the project and requirement management aims that they are managed correctly through the life cycle of the project. As they may affect the project plans directly and scope, timeline as well as the budget, REQM specifies the appropriate steps to ensure that requirements are reviewed with appropriate stakeholders, they are well understood, incorporated into the project plans and are being executed within the plan. Once the requirements are altered/evolved, there should be a commitment from the project participants before they are incorporated into the project plans. It should be documented with tracked the changes as they evolve and keep traceability between the product components and tests.

CMMI determines one specific goal for managing the requirements and provides five specific practices for that goal, which are in Table III.

TABLE III. CMMI REQUIREMENT MANAGEMENT PROCESS AREA[15]

The purpose of REQM is to manage the requirements of the project's components and to identify inconsistencies between those requirements and the work products
SG 1 Manage Requirements
SP 1.1 Obtain an Understanding of Requirements
SP 1.2 Obtain Commitment to Requirements
SP 1.3 Manage Requirements Changes
SP 1.4 Maintain Bidirectional Traceability of Requirements
SP 1.5 Identify Inconsistencies Between Project Work and Requirements

IV. PRINCIPLES AND TECHNICAL APPROACHES

The basic principle while developing a software is *"If requirements are unclear, incomplete or wrong, then the architecture will be equally wrong".*

The basic principle while developing a software is "If requirements are unclear, incomplete or wrong, then the architecture will be equally wrong".
In order to overcome the challenges and issues in requirement management, each specific goal related requirement definition or management– as also stated by CMMI – should be planned and handled properly within the SDLC.
In a waterfall project, requirement specification is the initial phase of the project. The aim is to define what needs to be delivered to the customer from the beginning and state that clearly so that everyone knows the scope and detailed descriptions. Requirement definition effort ends up with the collection of documents and diagrams, such as functional specifications or use case diagrams. These definitions are referred throughout the delivery of the project and should be kept up to date if any change occurs. Change requests often have a high impact on both resources, budgets, and timelines.

In several studies, CMMI and agile methodologies are compared [25], and mappings between them are proposed [17-22]. In an agile project which is an iterative approach to software development, customers do not need to work out all their requirements in details up front. "User stories" states the customer needs and defined as a representation of a small piece of functionality that provides some value to the business. It states what needs to be delivered to the customer for that functionality and can be often enough to be a few sentences. A good user story should determine who needs the requirement, why needed and what needs to do. In each iteration, sufficient requirements are analyzed and refined to support the delivery of that iteration's functionality. As customers get the delivery of iterations frequently, they are involved in the project progress directly and can refine their requirements for the next iterations.

The important point about requirements are the prioritization as if there are new prioritized requirements for that iteration, then there must be some requirements to push down the priorities. Agile welcomes change requests any time but require proper planning for each iteration[23, 24].

Beside of the SDLC model you use either waterfall or agile, software requirement management should cover below:

– Requirements should be elicited based on stakeholder needs and transformed into customer requirements.
– Product and work component requirements should be stated according to customer requirements.
– Requirements should be analyzed and validated with relevant stakeholders.
– Requirement definitions should be clear and well stated for each party. There should be a prioritization of each requirement to achieve balance and commitments need to be obtained for a requirement.
– Requirement changes should be analyzed, managed and tracked. There should a bi-directional traceability between requirements to work components of the technical solution, also to a test case and validation procedure that is related to that requirement.
– There should not be any inconsistencies between the project work components and the requirements.
– Project plans should allow managing the requirements and its related product areas in any phase throughout the project timeline.

V. CONCLUSION

The requirement engineering requires more focus in SDLC and in industrial development projects. Requirement engineering is defined as a systematic, multidisciplinary effort that collects needs from different stakeholders and maps them in software development stages. It needs to be managed through the SDLC and system development.

Requirement engineering covers mainly two activities: requirement development and management. Development is focused on actions of the investigation, analysis, specification, documentation, verification, and validation of the requirements. Whereas requirement management is related to traceability, change, and management of the requirements. In traditional SDLC models, like the waterfall, requirement development is performed mostly at the beginning of the development cycle and resistant to changes throughout the project lifecycle. In newer models, like agile, it is recognized that developing a requirement set in the beginning and keep it stable throughout the development is almost impossible and not effective. Therefore, a newer approach to define the requirements incrementally and iteratively in order to let stakeholders involve in each iteration and let the system to adopt is assimilated. Therefore, in agile, requirement change is more welcomed and requirement management is more important to ensure that changes are managed properly. In order to standardize different approaches, there are quality models that can be referenced, like CMMI. CMMI is not a process but a model and an approach that helps to reduce the risks and to improve the quality. CMMI includes two PAs (process areas) related to requirements, which are requirement development and requirement management. For this two process areas, CMMI specifies 'what' needs to be done but does not specify 'how to' do it. Therefore, it is applicable regardless of the SDLC model that you use. Moreover, it tries to ensure that proper REQM/RD can save on development costs if they are managed properly.

REFERENCES

[1] Jianxin (Roger) Jiao and Chun-Hsien Chen, "Customer Requirement Management in Product Development: A Review of Research Issues", Vol 14, Issue 3, 2006.
[2] P. Jalote, "An Integrated Approach to Software Engineering", 3rd edition, Narosa Publishing house, India, 2005.
[3] P. Parviainen, H. Hulkko, J. Kaariainen, J. Takalo & M. Tihinen, "Requirements Engineering", Inventory of Technologies, VTT Publications, Espoo, 2003.
[4] Sommerville & P. Sawyer, Requirements Engineering: A Good Practice Guide. John Wiley & Sons, 1997.
[5] R. H. Thayer & W. W. Royce, Software Systems Engineering, IEEE System and Software Requirements Engineering. IEEE Software Computer Society Press Tutorial. IEEE Software Society Press. Los Alamos, California, 1990.
[6] W. W. Royce, "Managing the Development of Large Software Systems" Proceedings of IEEE Wescon, Reprinted in Proceedings 9th International Conference Software Engineering IEEE Computer Society Press, Los Alamitos. California. USA, 1987, pp. 328-338.
[7] Cho, Juyun. Issues and Challenges of agile software development with SCRUM. Issues in Information System.VOL IX, No. 2. 2008.
[8] J. Siddiqi, "Requirement Engineering: The Emerging Wisdom", IEEE Software, 1996, pp.15- 19.
[9] M.E.C Hull, K. Jackson & A. J. J Dick, Requirements Engineering. Springer-Verlag. Berlin, 2002.
[10] R. Stevens, P. Brook, K. Jackson & S. Arnold, Systems Engineering - Coping with Complexity, Prentice Hall, London, 1998.
[11] S. Blanchard & W. J. Fabrycky, Systems Engineering and Analysis, Prentice-Hall, 1981.
[12] S. Lauesen, Software Requirements: Styles and Techniques, AddisonWesley,2002.
[13] H. Berlack, Software configuration management. John Wiley & Sons, 1992.
[14] F. Paetsch, A Eberlein, F Maurer - "Requirements Engineering and Agile Software Development", 2003.
[15] CMMI-DEV, CMMI for Development, V1.3 model, CMU/SEI-2010-TR-033. Software Engineering Institute, 2010.
[16] T. B. Alakuş, R. Daş, and İ. Türkoğlu, "Yazılım Geliştirme Süreçlerinin Analizi: Zorluklar, Tasarım Prensipleri ve Tekniksel Yaklaşımlar," in 2017 International Artificial Intelligence and Data Processing Symposium (IDAP), Inonu University, Malatya, 2017, pp. 1–10.
[17] D. Pandey, U. Suman, A. K. Ramani, "Impact of Requirement Engineering Practices in Software Development Processes for Designing Quality Software Products", National Conference on NCAFIS, DAVV, Indore, 2008.

[18] Anderson D. J., "Stretching Agile to fit CMMI Level 3 the story of creating MSF for CMMI Process Improvement at Microsoft Corporation" presented at Agile2005 Conference, [Online] http://www.agilemanagement.net/Articles/Papers/Agile_2005_Paper_DJA_v1_5.pdf. (December 2006)

[19] Kähkönen T. and P. Abrahamsson, "Achieving CMMI Level 2 with Enhanced Extreme Programming Approach," In proceedings of the 5th International Conference on Product Focused Software Process Improvement, pp. 378392, 2004

[20] Nawrocki J., W. Bartosz, and A. Wojciechowski, "Toward Maturity Model for eXtreme Programming," In proceedings of the 27th Euromicro Conference, pp. 233239, 2001.

[21] Paulk M. C., "Extreme Programming from a CMM Perspective," Software, vol. 18, issue 6, pp. 1926, 2001.

[22] Turner R., and A. Jain, "Agile Meets CMMI: Culture Clash or Common Cause," In proceedings of the Second XP Universe and First Agile Universe Conference on Extreme Programming and Agile Methods XP/Agile Universe, pp. 153165, 2002.

[23] M. Lang & J. Duggan, "A Tool to Support Collaborative Software Requirements Management" Requirements Engineering Journal 6(3), 2001, pp. 161–172.

[24] Agile Manifesto, Manifesto for Agile Software Development, [Online] http://agilemanifesto.org (Accessed date: 10.01.2018)

[25] G. Gurgoze, R. Das, and İ. Türkoğlu, "Comparison of software development process models, The 8th International Advanced Technologies Symposium (IATS-2017), Firat University, Elazig, Turkey, 2017, pp. 1923– 1932.