Requirements Anlysis, Refinement and Unambiguation using Requiements gathering and project management tools

Ravleen Kaur[1], Kailash bahl[2]

ABSTRACT

This research paper is based on Requirement Analysis Management practices. Most of systems fail just because of wrong elicitation practice which result in ambiguous and incomplete requirements . Without the requirement analysis techniques it is impossible to find out requirements and the needs of the developing system. This paper highlights how the gap between customer requirements specification and system requirement specification is reduced and how requirements are refined and optimized. It provides base to carry out correct design and implementation practices. Moreover all the companies focus to work within budget so it is very important to follow modern techniques and tools for requirement analysis practices so that project manager can achieve the set goals on time.

Keywords: Requirements Analysis, Traceability Matrix, Agile Process,

1. INTRODUCTION

Requirements analysis in systems engineering and software engineering, encompasses those tasks that go into determining the needs or conditions to meet for a new or altered product, taking account of the possibly conflicting requirements of the various stakeholders, analyzing, documenting, validating and managing software or system requirements.[1] Requirements analysis is critical to the success of a systems or software project[2]. The requirements gathered should be unambiguous, complete, consistent , measurable , testable , traceable , design free and understandable to bridge out the gap between CRS and FRD.

2. REQUIREMENT PROCESS MODEL AND SPECIFICATION

2.1 Requirement analysis

Analysis is the logical breakdown that proceeds from elicitation. Analysis involves reaching a richer and more precise understanding of each requirement and representing sets of requirements in multiple, complementary ways.

2.2 Functional analysis and allocation

Functional Analysis and Allocation is repeated to define successively lower-level functional and performance requirements, thus defining architectures at ever-increasing levels of detail . System requirements are allocated and defined in sufficient detail to provide design and verification criteria to support the integrated system design.

2.3 Design synthesis

Under this complete structure and design of the system are formulated by using the results of requirement and functional analysis. If results are incomplete or ambiguous, again iterations are done to get complete detailed requirements.

2.4 System analysis and control

Requirement analysis, functional analysis and allocation and design synthesis are mapped to system analysis and control process for every requirement in order to interpret results correctly.

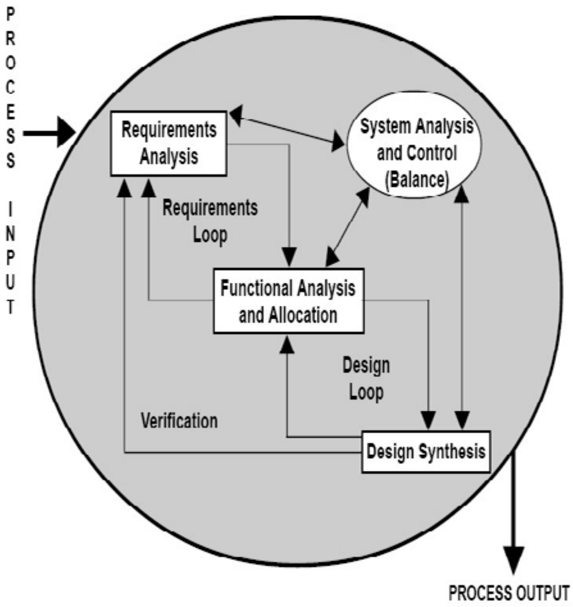

Fig.1 Requirement Process Model and Specification

3. TYPES OF REQUIREMENTS

Requirements are categorized in several ways. The following are common categorizations of requirements that relate to technical management [3]

1. Customer Requirements

Statements of fact and assumptions that define the expectations of the system in terms of mission objectives, environment, constraints, and measures of effectiveness and suitability (MOE/MOS). The customers are those that perform the eight primary functions of systems engineering, with special emphasis on the operator as the key customer. Operational requirements will define the basic need and, at a minimum, answer the questions posed in the following listing.

- *Operational distribution or deployment*: Where will the system be used?
- *Mission profile or scenario*: How will the system accomplish its mission objective?
- *Performance and related parameters*: What are the critical system parameters to accomplish the mission?
- *Utilization environments*: How are the various system components to be used?
- *Effectiveness requirements*: How effective or efficient must the system be in performing its mission?
- *Operational life cycle*: How long will the system be in use by the user?
- *Environment*: What environments will the system be expected to operate in an effective manner?

2. **Architectural Requirements**

Architectural requirements explain what has to be done by identifying the necessary system architecture of a system.[3]

3. **Structural Requirements**

Structural requirements explain what has to be done by identifying the necessary structure of a system.[3]

4. **Behavioural Requirements**

Behavioural requirements explain what has to be done by identifying the necessary behaviour of a system.[3]

5. **Functional Requirements**

Functional requirements explain what has to be done by identifying the necessary task, action or activity that must be accomplished. Functional requirements analysis will be used as the top level functions for functional analysis.[3]

6**. Non-functional Requirements**

Non-functional requirements are requirements that specify criteria that can be used to judge the operation of a system, rather than specific behaviours.[3]

7. **Performance Requirements**

The extent to which a mission or function must be executed; generally measured in terms of quantity, quality, coverage, timeliness or readiness. During requirements analysis, performance (how well does it have to be done) requirements will be interactively developed across all identified functions based on system life cycle factors; and characterized in terms of the degree of certainty in their estimate, the degree of criticality to system success, and their relationship to other requirements[3]

8. **Design Requirements**

The “build to,” “code to,” and “buy to” requirements for products and “how to execute” requirements for processes expressed in technical data packages and technical manuals.[3]

9. **Derived Requirements**

Requirements that are implied or transformed from higher-level requirement. For example, a requirement for long range or high speed may result in a design requirement for low weight.[3]

10. **Allocated Requirements**

A requirement that is established by dividing or otherwise allocating a high-level requirement into multiple lower-level requirements. Example: A 100-pound item that consists of two subsystems might result in weight requirements of 70 pounds and 30 pounds for the two lower-level items.[3]

4. REQUIREMENT ANALYSIS MANAGEMENT

This process is categorized into following two categories:

- Requirement eliciting practices
- Requirement analysis and specification practices

4.1 Requirement Eliciting practices

1. **Analyze and Elicit Requirements**

   Elicitation is the gathering and discovery of requirements from stakeholders and other sources.

   For this first consolidate and affirm the business needs. Analyze the intended use of the system and focus on gathering sufficient data to specify the functional and data requirements. Connect the functional requirements to the data requirements. Diagnose functional and system requirements that are not easily expressed by customer.[4]

- **Stakeholder identification** – anyone who is directly or indirectly concerned with the system. People who operates the system or benefited from the system are identified.
- **Stakeholder Interviews** – stakeholders are interviewed in order to figure out actual business processes taking place.
- **JRD Sessions** - Joint requirement development sessions are facilitated by a business analyst where in stakeholders participate in discussions to elicit requirements, analyze their details and uncover cross-functional implications. A dedicated scribe should be present to document the discussion, freeing up the Business Analyst to lead the discussion in a direction that

generates appropriate requirements which meet the session objective.

3 **Record Customer Requirements**
The requirement analyst is responsible for the recording all the requirements of the system to be developed. Modern techniques and tools should be used to record the requirements. So that gap between CRS and FRD can be easily bridged by modelling requirements.

4 **Articulate Functional Requirement**
The Functional and Data Requirements identification is conducted. Business requirement are mapped to software requirements through this articulation. So as to ensure a complete understanding of the requirements in order to formulate FRD.

5 **Revise previous documentations**
Review and update previous phase documentation if necessary before publishing final documentation to the project team.

4.2 Requirement analysis and specification practices

1. **Functional requirement document** - Serves as the foundation for system design and development; captures user requirements to be implemented in a new or enhanced system; the systems subject matter experts document these requirements into the requirements traceability matrix, which shows mapping of each detailed functional requirement to its source. This is a complete, user oriented functional and data requirements for the system which must be defined, analyzed, and documented to ensure that user and system requirements have been collected and documented. [4]

- **Modeling and analyzing requirements** - Modeling is the construction of abstract descriptions that are amenable to interpretation – is a fundamental activity in RE. So much so that a number of RE textbooks (e.g., [6]), focus almost entirely on modeling methods and their associated analysis techniques. Models can be used to represent a whole range of products of the RE process. Moreover, many modeling approaches are used as elicitation tools, where the modeling notation and partial models produced are used as drivers to prompt further information gathering. The key question to ask for any modeling approach is "what is it good for?", and the answer should always be in terms of the kind of analysis and reasoning it offers. We suggest below some general categories of requirement engineering modeling approaches, and give some example techniques under each category. We then suggest some analysis techniques that can be used to generate useful information from the models [11].

- **Communicating Requirements** - Requirement engineering is not only a process of discovering and specifying requirements, it is also a process of facilitating effective communication of these requirements among different stakeholders. The way in which requirements are documented plays an important role in ensuring that they can be read, analyzed, (re-)written, and validated. The focus of requirements documentation research is often on specification languages and notations, with a variety of formal, semi-formal and informal languages suggested for this purpose [6]. From logic [7] to natural language [2], different languages have been shown to have different expressive and reasoning capabilities.[11]

- **Agreeing Requirements** - As requirements are elicited and modeled, maintaining agreement with all stakeholders can be a problem, especially where stakeholders have divergent goals. Recall that validation is the process of establishing that the requirements and models elicited provide an accurate account of stakeholder requirements. Explicitly describing the requirements is a necessary precondition not only for validating requirements, but also for resolving conflicts between stakeholders.

- **Evolving Requirements** - unsuccessful software systems always evolve as the environment in which these systems operate changes and stakeholder requirements change. Therefore managing change is a fundamental activity in requirements engineering [8]. Changes to requirements documentation need to be managed. Minimally, this involves providing techniques and tools for configuration management and version control [9], and exploiting traceability links to monitor and control the impact of changes in different parts of the documentation. Typical changes to requirements specifications include adding or deleting requirements, and fixing errors. Requirements are added in response to changing stakeholder needs, or because they were missed in the initial analysis. Requirements are deleted usually only during development, to forestall cost and schedule overruns, a practice known as requirements scrubbing [10].

- **Use case diagrams** - A use case is a structure for documenting the functional requirements for a system, usually involving software, whether that is new or being changed. Each use case provides a set of scenarios that convey how the system should interact with a human user or another system, to achieve a specific business goal. Use cases typically avoid technical jargon, preferring instead the language of the end-user or domain expert. Use cases are often co-authored by requirements engineers and stakeholders

- **Activity diagrams** - Activity diagram is basically a flow chart to represent the flow form one activity to another activity. The activity can be described as an operation of the system. So the control flow is drawn from one operation to another. This flow can be sequential, branched or concurrent. Activity diagrams deals with all type of flow control by using different elements like fork, join etc.

2. Requirements traceability Matrix

A requirements traceability matrix may be used to check to see if the current project requirements are being met, and to help in the creation of a software requirements specification, various deliverable documents, and project plan tasks.

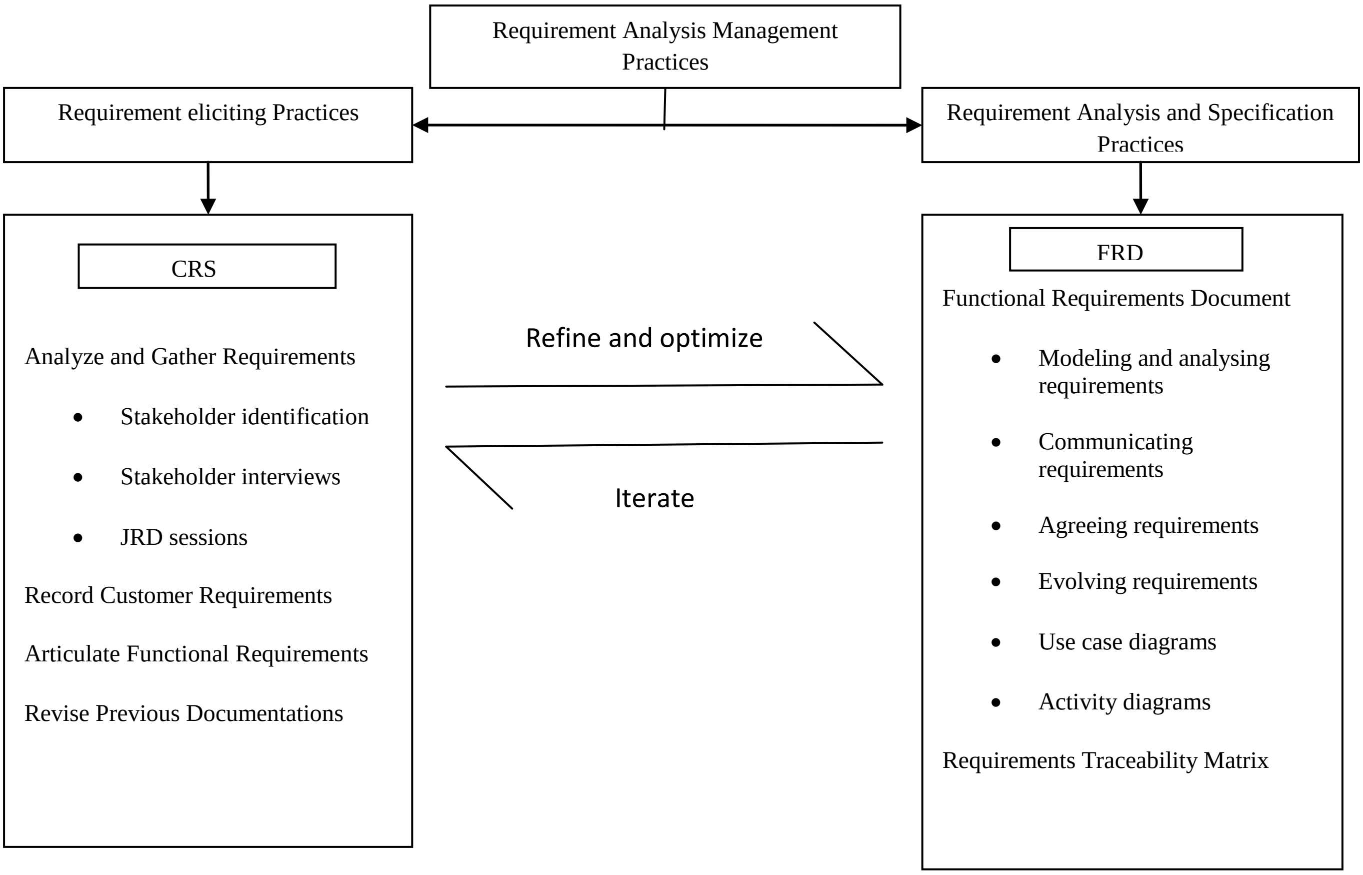
Requirement Analysis Management Practices
Requirement eliciting Practices
Requirement Analysis and Specification Practices
CRS
Analyze and Gather Requirements
Stakeholder identification
Stakeholder interviews
JRD sessions
Record Customer Requirements
Articulate Functional Requirements
Revise Previous Documentations
Refine and optimize
Iterate
FRD
Functional Requirements Document
Modeling and analysing requirements
Communicating requirements
Agreeing requirements
Evolving requirements
Use case diagrams
Activity diagrams
Requirements Traceability Matrix

Fig 2: Block diagram showing requirement analysis management process

5. REQUIREMENTS OPTIMIZATION AND REFINING

5.1 Requirements Optimization.

Once an initial set of requirements has been gathered by requirements elicitation, there is a business-level analysis problem: choices have to be made to identify optimal choices and trade–offs for decision makers. For example, one important goal is to select near optimal subsets from all possible requirements to satisfy the demands of maximum no. of customers first. Hence optimization have to be done for selecting requirements for the next release.[12]

5.2 Requirement Refining

Requirements refinement is not only a process to derive specifications, but also a necessary means towards preparing architectural designs. Refinement framework comprises methods and process, to move from requirements to design. The framework consists of multiple abstraction levels capturing requirements of different kind and granularity, allows the derivation of designer and user perspectives separately, introduces stages of activities to be carried out in the process, includes refactoring-based methods to carry out the refinement, and provides a structure that allows easy traceability between requirements.[13]

6. CONCLUSIONS

Requirements are prone to issues of ambiguity, incompleteness, and inconsistency. Techniques such as rigorous inspection have been shown to help deal with these issues. Moreover various tools should be used for requirement elicitation practices. Ambiguities, incompleteness, and inconsistencies that can be resolved in the requirements phase typically cost orders of magnitude less to correct than when these same issues are found in later stages of product development. Requirements analysis strives to address these issues. Refer to annexure 1 displaying report using agile tool. This report can be submitted to the client for authenticating the completeness and un-ambiguity. So it is very important to optimize and refine requirements in order to bridge out gap between CRS and FRD.

REFERENCES

[1] Kotonya, G. and Sommerville, I. 1998. *Requirements Engineering: Processes and Techniques* Chichester, UK: John Wiley and Sons.

[2] Executive editors: Alain Abran, James W. Moore; editors Pierre Bourque, Robert Dupuis, ed (March 2005). "Chapter 2: Software Requirements ". Guide to the software engineering body of knowledge (2004 ed.). Los Alamitos, CA: IEEE, Computer Society Press. ISBN: 0-7695-2330-7.

[3] *Systems Engineering Fundamentals* Defense Acquisition University Press, 2001Conference on Human Factors in Computing Systems

[4] Information Resources Management , January2003 , System Development Life Cycle Guidance Document http://www.justice.gov/jmd/irm/lifecycle/ch6.htm

[5] http://ocw.metu.edu.tr/pluginfile.php/401/mod_resource/content/0/lecture5.pdf

[6] Davis, A. (1993). Software Requirements: Objects, Functions and States. Prentice Hall..

[7] http://www.swebok.org/ch2.html. Retrieved 2007-02-08. "It is widely acknowledged within the software industry that software engineering projects are critically vulnerable when these activities are performed poorly."

[8] Bohner, S. A. & Arnold, R. S. (Ed.). (1996). Software Change Impact Analysis. IEEE Computer Society Press

[9] Estublier, J. (2000). Software Configuration Management: A Roadmap. In this volume. decisions", Journal of Systems and Software, 2005, in press.

[10] Boehm, B. (1991). Software Risk Management: Principles and Practices. IEEE Software, 8(1): 32-41.

[11] B. Lal et al., International Journal of Advanced Research in Computer Science and Software Engineering 3(2), February-2013,pp.311-316

[12] http://www0.cs.ucl.ac.uk/staff/a.finkelstein/papers/refsqmh.pdf

[13] Wen Qian Liu (2008) ACM symposium on Applied computing Pages 658-659