



Business Requirements - A Study on Requirement Solutions for Software Development

B. Dhanalaxmi

Asst Prof,

Institute of Aeronautical,

Hyderabad, India

dinnu18@gmail.com

Abstract - Requirement Engineering have become the mainstream of Software development due to their enriched practices. Commonly occurs practices include collaborative, development, meeting evolving requirements with working software, simple design etc. It has been observed that effective set of business requirements rely on expert opinion and historical data of project for estimation of cost, size and duration and also observed that these methods do not consider the vital factors affecting the same of project for estimation. The main aim of the paper works on analysis of requirements specifications and points to write effective business requirements which need to follow in development process and also with agile process. Our analysis also improves the software quality assurance by using this set of business requirements which are effective from existing system.

Keywords – Software Engineering, Requirement analysis, elicitation, functional, non-functional requirements

I. INTRODUCTION

Software systems requirements engineering is the process of discovering by identifying stakeholders and their needs, documenting these in a form that is amenable to analysis, communication, and subsequent implementation. There are a number of inherent difficulties in this process. Stakeholders (including paying customers, users and developers) may be numerous and distributed. Their goals may vary and conflict, depending on their perspectives of the environment in which they work and the tasks they wish to accomplish. Their goals may not be explicit or may be difficult to articulate, and, inevitably, satisfaction of these goals may be constrained by a variety of factors outside their control.

Software engineering seeks to address the problem of ensuring that quality software gets delivered on time and on budget by setting out a systematic and disciplined approach to the development of software. The current software engineering [1] ideas are based on the following principles.

- A sound understanding of development processes, project management and the ability to measure, monitor and control software development;
- A sound understanding of the problem to be solved, the design methods required to solve it and the platforms used to implement the solution;
- A sound understanding of the range of tools and techniques required to support the processes and how they are to be used effectively to support problems.

Engineering approach to developing software – and this comes from an engineering perspective – then there are essentially three phases in the development of an artifact, such as a computer program:

A. Analysis:

In which we must understand what exactly it is that we must build and how it should work in order to be fit for purpose,

B. Synthesis:

In which we must take our analysis and derive designs for building a concrete artifact,

C. Realization:

In which we must make our product a reality, that is, we must somehow build it and test that what we have built is fit for purpose and meets the needs of our client.

Study of requirement analysis focus on the first of these activities, that is, how to analyze a client's problem and their problem domain so that an understanding can be gained of exactly what software needs to be created in order to solve this problem. Such Requirements Analysis requires that we determine all of the functions, constraints, qualifications, and other information relevant to the system, and organize all of the information in a clear and unambiguous manner [2]. Requirements analysis may occur repeatedly during the course of the development of a piece of software. There are many possible paths to understanding requirements engineering but time prohibits us from exploring them all. Our path in this subject will be to view requirements engineering a "problem understanding" and "domain analysis" exercise. The set of requirements for the software come from this analyses. Requirements engineering in these notes begin with two related definitions and add to them as the subject progresses by a requirement. Definition for concreteness, but again, it will be added to as the subject progresses [3].

- A requirement mandates that something be accomplished, transformed, produced or provided.
- Requirements engineering is the discipline concerned with understanding and documenting software requirements.

If we adopt this definition then the aim of requirements engineering is to uncover what the system must accomplish, transform, produce or provide in order to meet a client's needs and that means developing an in depth understanding of a client's problem domain.

II. REQUIREMENT ENGINEERING:

The goal of the Requirements is to describe what the system should do and allows the developers and the

customer to agree on that description. To achieve this, we elicit, organize, and document required functionality and constraints; track and document tradeoffs and decisions. Requirement Engineering is classified into following

A. Requirement Elicitation:

Using an elicitation method can help in producing a consistent and complete set of security requirements. However, brainstorming and elicitation methods used for ordinary functional (end-user) requirements usually are not oriented toward security requirements and do not result in a consistent and complete set of security requirements. The resulting system is likely to have fewer security exposures when security requirements are elicited in a systematic way.

B. Requirement Analysis:

Requirements analysis involves frequent communication with system users to determine specific feature expectations, resolution of conflict or ambiguity in requirements as demanded by the various users or groups of users, avoidance of feature creep and documentation of all aspects of the project development process from start to finish. Energy should be directed towards ensuring that the final system or product conforms to client needs rather than attempting to mold user expectations to fit the requirements.

C. Requirement Validation:

Validation works with a final draft of the requirements document i.e. with negotiated and agreed requirements

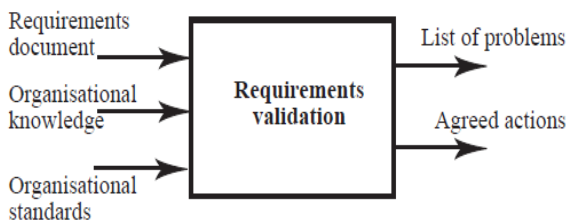


Figure 1 represents the validation of requirements document

- a. Should be a complete version of the document, not an unfinished draft. Formatted and organized according to organizational standards

Organizational Knowledge

- b. Knowledge, often implicit, of the organization which may be used to judge the realism of the requirements

Organizational Standards

- c. Local standards e.g. for the organization of the requirements document

Problem list

- d. List of discovered problems in the requirements document

Agreed Actions

- e. List of agreed actions in response to requirements problems. Some problems may have several corrective actions; some problems may have no associated actions

III. PROBLEM DOMAIN

Problem define in this article is how to analyze a set of business requirements, first understanding the business requirements in the manner in which they were intended can be a very challenging proposition. If the requirements are going to do their job well they need to be understood by several target audiences, namely business analyst, project managers and leaders, subject matter experts and quality

assurance experts and developers/testers. Each of these groups needs to be able to read the business requirements and extract what they need out of them to be able to contribute to the end product.

This article presents several requirements and phrases that might be misunderstood by the various target audiences. It assumes that to know how to write effective, measurable business requirements or going to analyze other people's requirements. This technique will help to reduce the number of incorrect and misunderstood business requirements.

A. Analyzing the Business Requirements

a. Identify Key Stakeholders

Identify the key people who will be affected by the project. Start by clarifying exactly who the project's sponsor is. This may be an internal or external client. Either way, it is essential that you know who has the final say on what will be included in the project's scope, and what won't.

Then, identify who will use the solution, product, or service. These are your end-users. Your project is intended to meet their needs, so you must consider their inputs.

We have several methods to understand and to capture the requirements. Using stakeholder interview: Communication with each stakeholder individually allows to understand specific views and needs. Using Joint interviews or groups: This will help to understand how information flows between different divisions or departments and ensure that hand-overs will be managed smoothly.

Using "use-cases": This technique walk through the whole system or process step by step as user, it helps to understand how the system or service would work. This is good technique for gathering functional requirements but may need multiple "use-cases" to understand the functionality of the system.

Building Prototypes: Build a model of the system or product to give users an idea of what the final product will look like. Using this users can address feasibility issues and they can help identify any inconsistencies problems. For example after the interviews finalize the list of requirements then can build a prototype of the system or product.

b. Categorize requirements:

Grouping of the requirements is classified into four Functional Requirements that define those features of the system that requirements will specifically satisfy a Consumer need, or with which the consumer will directly interact. Technical Requirements that identify the technical constraints or Requirements define conditions under which the system must perform. Operational Requirements that define those "behind the scenes" Requirements functions that are needed to keep the system operational over time.

Transitional Requirements that define those aspects of the system that requirements must be addressed in order for the system to be successfully implemented in the production environment, and to relegate support responsibilities to the Performing Organization.

c. Interpret and Record Requirements:

Once gathered or categorized all the requirements, determine which requirements are achievable and how the system or product can be deliver them. To interpret the

requirements, consider the following in the analysis Define requirements precisely – ensure that the requirements are:

- Not ambiguous or vague.
- Clearly worded.

Sufficiently detailed so that everything is known (problems usually come from unknowns that were not identified or sufficiently well-analyzed)

- Related to the business needs

Listed in sufficient detail to create a working system or product design

B. Decomposition of Requirements:

Once the actual problem is identified the next step in analysis is to decompose the problem into smaller distinct elements, and to further refine their individual characteristics. The intent is to gain additional insights into the problem, and subsequently, a better understanding of the customer needs. Decomposition and refinement are activities of Decomposition.

Decomposition involves a series of steps by means of which a set of needs is obtained, from which the requirements are derived. Like its predecessor, Decomposition is an iterative process that begins with root cause analysis. This presumes, of course, that the actual problem has been identified and has a well-defined problem statement. In addition, recommend that the analyst be educated about the problem-specific aspects of the customer's domain and the environment within which the new/modified system will eventually function. This indoctrination either outlines or provides the basis for (a) defining the solution boundaries and (b) identifying the constraints to be imposed on the solution space, e.g., economic, political, technical, etcetera.

Problem Decomposition is an iterative process that often involves numerous meetings with the customer. Prior to each meeting the analyst needs to ensure that he/she has identified the appropriate set of stakeholders (or meeting participants), set a meeting agenda, and has outlined each participant's role and responsibilities prior to the meetings. To support problem refinement, each meeting must have a focused objective and employ structured activities that support the achievement of that objective, e.g., recording decomposition components and evolved needs, and monitoring/controlling the meeting process. Finally, at the end of each meeting, the identified "set of needs" are evaluated relative to their correctness, completeness and non-ambiguity. The final set of customer needs can be "validated" (in a loose sense) against the Con-Ops document or against the set of high-level requirements formed during the systems engineering process. As illustrated in Figure 1, the set of customer needs are then provided as input to the requirements elicitation process and form the basis from which the requirements (or solution specification) are derived.

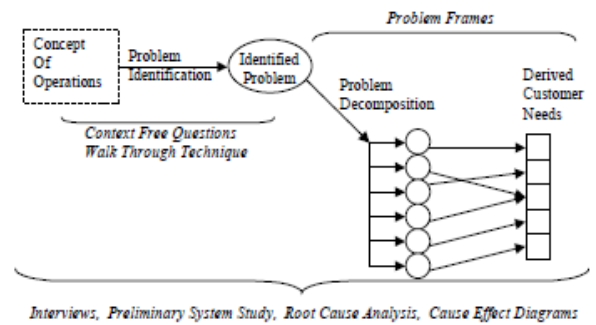


Figure 2 provides an illustration of the Problem analysis and decomposition

C. Business Requirement Verification:

All the requirements should be verifiable and test. Verification method should be used instead (eg analysis demonstration or inspection review of design) certain requirements by their very structure are not verifiable. These include requirements that say the system shall never or always exhibit a particular property. Proper testing of these requirements would require an infinite testing cycle. Such requirements must be rewritten to be verifiable.

Non-functional requirements which are unverifiable at the software level must still be kept as a documentation of customer intent, however they may be traced to process requirements that are determined to be practical way of meeting them. For example a non-functional requirement to be free from backdoors may be satisfied by replacing it with a process requirement to use pair programming

D. Prioritize Requirements:

Although many requirements are important, some are more important than others and budgets are usually limited. Therefore identify which requirements are the most critical and which are nice-to-haves.

Analyze the impact of change – carry out the impact analysis to make sure that fully understand the consequences of the project will have existing product or project.

Resolve Conflicting issues – Communicate with the key stakeholder and resolve any conflicting requirements issues. Find scenario analysis helpful in doing this as it will allow all those involved to explore how the proposed project would work in different possible "futures"

Analyze Feasibility – Determine how reliable and easy-to-use the new product or system will be. A detailed analysis can help to identify any major problems.

Once everything is analyzed present the key results and detailed report of the business needs is a written document. Circulate this document among the key stakeholders, end-users and developer teams with a realistic deadline for feedback. This can help to resolve any remaining stakeholder conflicts and can form part of a agreement between analyst and stakeholders.

E. Requirement Tools:

Requirement Documentation Template, requirements engineering tools used to develop requirements, however, this flexibility requires programming in Software Engineering.

IV. SYSTEM REQUIREMENT SPECIFICATION

The purpose of System Requirements Analysis is to obtain a thorough and detailed understanding of the business

need as defined in[6] Project Origination and captured in the Business Case.

A. Functional Requirements:

Functional requirement defines a function of a Product system or its component. A function is described as a set of inputs, the behavior, and outputs. Functional requirements may be calculations, technical details, data manipulation and processing and other specific functionality that define what a system is supposed to perform. Behavioral requirements describing all the cases where the system uses the functional requirements are captured in use-cases. In requirements engineering, functional requirements specify particular results of a system. This should be contrasted with non-functional requirements which specify overall characteristics such as cost and reliability. Functional requirements drive the application architecture of a system, while non-functional requirements drive the technical architecture of a system.

B. Non-functional Requirements:

Non-Functional Requirements in software engineering presents a systematic and pragmatic approach to building quality into software systems. Systems must exhibit software quality attributes, such as accuracy, performance, security and modifiability. However such non-functional requirements are difficult to address in many projects, even though there are many techniques to meet functional requirements in order to provide desired functionality.

C. Agile Process Business Requirements:

Agile process model launching new generation products, adapts a highly flexible requirements. We propose the business requirements for agile methodologies Agile Unified Process for product development. This is a combination of agile techniques of XP, Scrum, Lean. This model enables us to model and document in an agile manner, thereby giving us a flexible approach so that we can easily adapt to the changing requirements. Agile methodology enables us to release the entire product in versions. There are two kinds of releases in this methodology.

a. Development release:

Development releases are small releases and have the potential of being released on the production server. However, they have not undergone preproduction steps like testing and deployment.

b. Production release:

These releases have been made on the production server. Incremental releases are made by the development releases, whereas major functionalities are released in every production release. The first production release is greater in time than others.

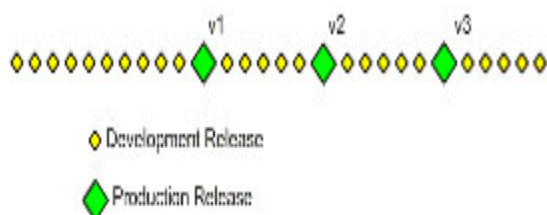


Figure 4. types of releases in the Agile process

Agile Requirements: Every requirement (including change requirements) that needs to be implemented is prioritized and stacked. Any new requirement is added into this stack after prioritizing. An existing requirement can be removed from the stack. In addition, the priorities can change with time and therefore, the stack is reshuffled. Every iteration implements those number of highest priority task that can be implemented in the current iteration.

Four phases of Agile Mythologies

Inception
Technical Design - Elaboration
Construction - Coding
Transition

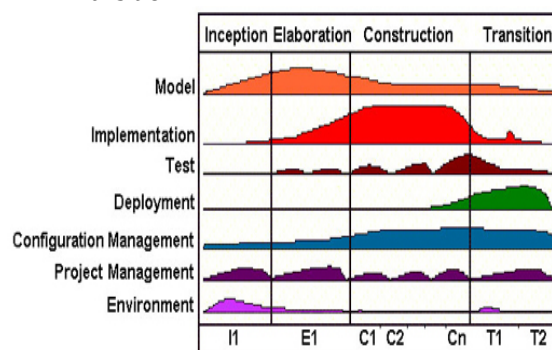


Figure 5. quality overview for agile process

c. Inception:

First phase of the life cycle where the initial scope of the entire project is identified. Based on the initial requirements analysis, proposes the potential architecture to the client, come with an agreement and obtain the initial funding for the project.

d. Elaboration:

Second phase of the life cycle consists of proving the potential architecture of the system. Making the technical design - both High Level Design and Low Level Design. HLD will focus on laying the overall architecture and framework for the project. It results in Project Decomposition into modules/functions/entities/classes etc. LLD incorporates the pseudo code and definition of all technical interfaces of the project.

e. Construction:

Third phase consists of construction phase where we build working software on a regular, incremental basis. Hence, construction phase consists of series of development releases. Development releases are small releases and have the potential of being released on the production server. However, they have not undergone pre-production steps like testing and deployment.

f. Transition:

Last phase includes validation for quality assurance and deployment to the production server. It differs from the transition phase of unified process as instead of releasing the entire product in one release, the product is released in versions. The first production release is greater in time than others. The entire life cycle of agile unified process is repeated for every production release with few modifications. After every production release, the business requirements for the project are modified. Any new change requirements, after prioritizing, are added in the stack. The

stack may be reprioritized. Changes, if needed are made in the technical designs. Then, the construction and the transition phases follow, where every iteration implements those number of tasks that can be implemented in the current iteration.

Model - The business requirements are understood and solution is identified to address problem domain. Implementation The program source code is developed and unit testing is done.

Test - This discipline ensures quality of the system developed. It consists of finding bugs, ensuring that the system works as per the design of the system and meets all requirements mentioned in the specification documents. Deployment - This includes planning and executing delivery of software and supporting documentations ready to be deployed and making the system available to the user.

Configuration Management - This includes managing baselines of the project, changing and delivering configuration items and managing releases. Project Management - This includes assigning tasks, managing risks, tracking progress etc to ensure on time and within budget delivery of the product. Environment - This includes ensuring proper tools are available when we require.

V. CONCLUSION

Requirement analysis stresses rapid iterations and frequent releases and evolving processes facilitated by direct user involvement in the development process. In this paper Requirements analysis provides a method set of business requirements to visualize scope, orchestrate and repetitive development tasks and enforce process. Requirement management phase has many activities which are elicitation, analysis, validation and system requirements specifications

categorized into functional and non-functional requirements are analyzed. My research business requirements steps presents effective method to existing systems. Future work to this article analyzes the metrics for our business requirements.

VI. REFERENCES

- [1] M. Cotterell and B. Hughes. Software Project Management. International Thomson Press, 1995.
- [2] Davis, A.M. Software Requirements: Objects, Functions and States. Prentice Hall, 1993. Revised Edition.
- [3] D. Kulak and E. Guiney. Use Cases: Requirements in Context. ACM Press, 2000.
- [4] Barry W. Boehm, A Spiral Model of Software Development and Enhancement, Computer, May 1988, IEEE, pp.61-72
- [5] Barry W. Boehm, Anchoring the Software Process, IEEE Software, 13, 4, July 1996, pp. 73-82.
- [6] Grady Booch, Object Solutions, Addison-Wesley, 1995.
- [7] Grady Booch, Ivar Jacobson, and James Rumbaugh, Unified Modeling Language 1.3, White paper, Rational Software Corp., 1998.
- [8] Alan W. Brown (ed.), Component-Based Software Engineering, IEEE Computer Society, Los Alamitos, CA, 1996, pp.140.
- [9] Michael T. Devlin, and Walker E. Royce, Improving Software Economics in the Aerospace and Defense Industry, Technical paper TP-46, Santa Clara, CA, Rational Software Corp., 1995
- [10] Ivar Jacobson, Magnus Christerson, Patrik Jonsson, and Gunnar Övergaard, Object-Oriented Software Engineering—A Use Case Driven Approach, Wokingham, England, Addison-Wesley, 1992, 582p.