2

Requirements Engineering Principles

CHAPTER COVERAGE

1. *Introduction*
2. *What is Requirements Engineering?*
3. *Importance of Requirements*
4. *Types of Requirements*
5. *Steps Involved in Requirements Engineering*

2.1 INTRODUCTION

As discussed in the previous chapter, in software development, the software project life cycle starts with capturing the requirements of the project. Developing a product or executing a project is nothing but catering and taking the requirements to the next level. Gathering, documenting, disseminating and managing the requirements is the key to success of any project. Although it sounds easy, gathering the requirements effectively and managing them efficiently is a complex task and needs to be handled in a systematic manner. Requirements engineering principles help to do this task in a better way.

According to industry average, more than 90% of the software projects fail due to faulty or incomplete requirements capturing. This certainly gives an idea of how important this step is for the success of any software project. In this chapter, we will cover the different aspects of requirements engineering with steps and guidelines.

2.2 WHAT IS REQUIREMENTS ENGINEERING?

Before getting into the details of requirements engineering, let us understand what "requirement" is.

According to IEEE Standard 610.12-1990, requirement is defined as "a condition or capability needed by a user to solve a problem or achieve an objective" or "a condition or capability that must be met or processed by a system or system component to satisfy a contract, standard, specification, or other formally imposed documents."

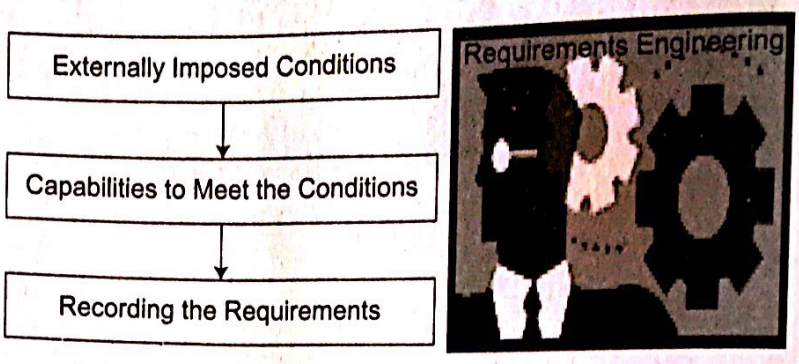


Figure 2.1 Requirements Engineering

This means that the requirements originate from the user's needs and are then captured and tracked to satisfy a contract or a specification.

Now, let us look into the definition of "requirements engineering."

"Requirements engineering is the discipline that explores the externally imposed conditions on a proposed computer system and tries to identify the capabilities that will meet those imposed conditions and recording the same in the form of documentation called the requirement document of the computer system."

The whole gamut of activities related to requirements from inception to understanding, tracking and maintaining the requirements is termed requirements engineering (Figure 2.1).

2.3 IMPORTANCE OF REQUIREMENTS

Requirements are the stepping stones to the success of any project. If software projects get started without properly understanding the user's needs or without exploring the multiple dimensions of the requirements, there will be misalignment at the end between the final result delivered and the user's expectations of the project resulting in a lot of rework.

In software development, primary focus is usually given to the construction phase, which leads to a lot of problems at the end.

A software project has to be completed within a specified time frame and budget. However, in some cases, rework has to be done due to incomplete requirements understanding, which is the primary cause of schedule and budget overruns.

Figure 2.2 shows the relative cost of repairing a defect at each stage of the software project life cycle. It is evident that the earlier the error is detected, the easier and cheaper it is to fix the error.

For example, if a defect in the requirements is found at the requirements review stage, then it will cost less (e.g., $x) to fix the issue, but as this faulty requirement percolates through the later stages it would cost more to fix the same issue: 3 times the cost to fix it during the design stage, 5 to 10 times the cost to fix it during the construction stage, 10 times the cost to fix it during the system testing stage of the project and 10 to 100 times the cost to fix it during the postrelease phase. Therefore, identifying and fixing errors at an early stage will cost less to the organization.

This signifies that capturing the requirements completely and correctly at the early stages of the project life cycle is essential to keep the project within the budget so that it is profitable. The only way to ensure this is to follow a disciplined requirements engineering process

Cost correct an Error		Stage detected				
		Requirements	Design	Construction	System Test	Post release
Stage introduced	Requirements	1X	3X	5-10X	10X	10-100X
	Design	-	1X	10X	15X	25-100X
	Construction	-	-	1X	10X	10-25X

Figure 2.2 Cost of Fixing Errors at Different Stages

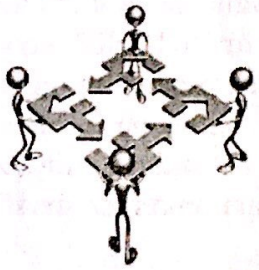
POINTS TO PONDER

The cost of fixing an error in the system increases exponentially with the stage of the project.

2.4 TYPES OF REQUIREMENTS

Requirements can be categorized into three main types: functional requirements, nonfunctional requirements, and interface specification.

Types of Requirements
- Functional Requirements
- Non Functional Requirements
- Interface Specification



2.4.1 Functional (User) Requirements

The set of requirements (Figure 2.3) that defines what the system will do or accomplish is called functional requirements. These requirements determine how the software will behave to meet users' needs.

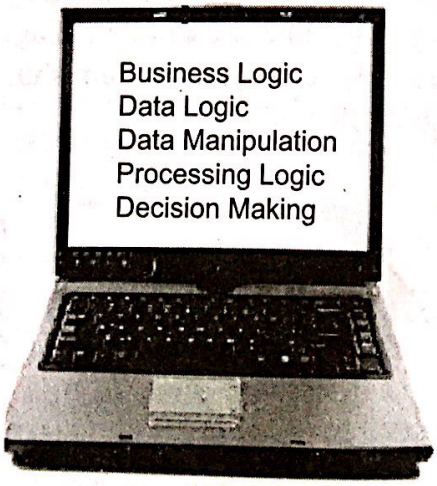


Figure 2.3 Functional Requirements Characteristics

The requirements may be for performing calculations, data manipulation, processing, logical decision-making, etc., which form the basis of business rules. Functional requirements are often captured in use cases. Functional requirements are the main drivers of the application architecture of a system. An example of functional requirement is a retail shop billing software having the functionalities of capturing commodity prices, calculating discounts, generating bills, printing invoices, etc.

2.4.2 Nonfunctional (System) Requirements

Functional requirements are supported by nonfunctional requirements (see Figure 2.4). The quality attributes and the design and architecture constraints that the system must have are called nonfunctional requirements (NFRs).

Some of the quality attributes of the system from an end user's perspective include performance, availability, usability and security and from a developer's perspective include reusability, testability, maintainability and portability.

NFRs as shown in Figure 2.4 are critical in most of the software projects than functional requirements. NFRs cater to the architectural needs of the overall system, whereas functional requirements cater to the design needs of the overall system.

Some examples of NFRs are:

"The system should be available 24 × 7" (Availability)

"The system should save or fetch 1000 records in 1 second" (Performance)

NFRs may be related to the product or the organization or to the external requirements of the system. Product-related NFRs include performance, availability, maintainability, portability, reliability, security, scalability, testability and usability, whereas organization-related NFRs include standards requirements and implementation requirements. External NFRs include legal requirements and ethical requirements.

Measuring NFRs We need to first define measurable criteria for each NFR and then realize the metrics by measuring it. For example, Availability can be first defined in terms of percentage and then define the duration within which we are going to measure it. To measure the availability, we can assume the availability of the system in the past 1 week to be 100% as the system was running continuously without any downtime (another related metrics). A Service Level Agreement is established with the customer for setting an agreed level of availability depending on the stability of the system. If the system is more stable without any downtime, we can fix the agreed limit of availability as 100%. If the system is not stable, then there will be frequent system shut downs and the target availability needs to be reduced accordingly (e.g., 95% or 90%).

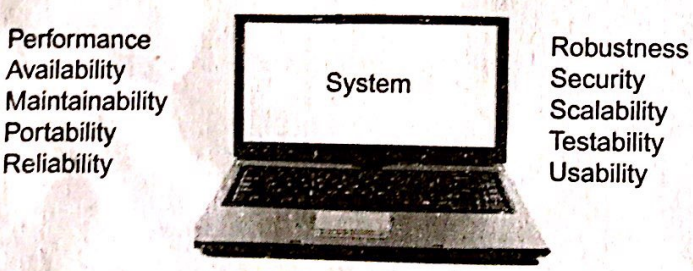


Figure 2.4 Nonfunctional Requirements Characteristics

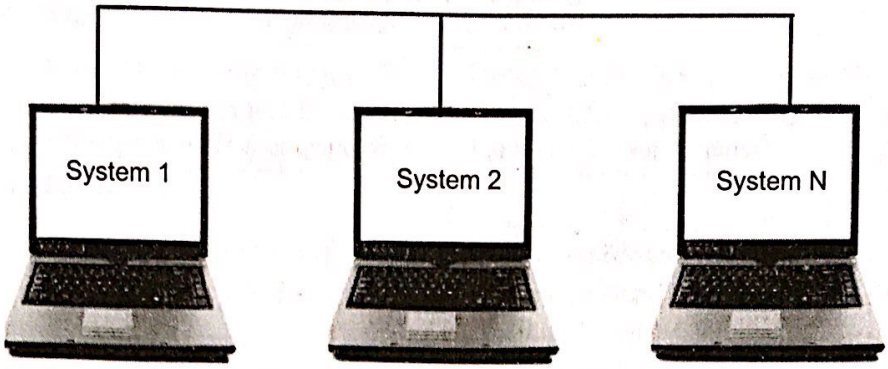


Figure 2.5 Interface Specification Characteristics

Approaches to NFRs NFRs can be approached in two ways- product-oriented and process-oriented approaches. As the names suggest, the product-oriented approach concentrates on the product and its associated NFR aspects, while the process-oriented approach concentrates on the process and its associated NFR aspects. Second, the NFRs can be approached qualitatively and quantitatively. The qualitative approach concentrates on non-numeric-related NFRs, whereas the quantitative approach focuses on the numeric aspects of the NFRs.

2.4.3 Interface Specification

Most of the software systems do not work alone and need to interact with other systems in order to work (Figure 2.5). They interact with many other systems to receive and send data, get help in processing logic, store information in other systems, etc. The interaction requirement of one system with another is defined in the interface specification.

Interface specification helps in building different systems in such a fashion that they can seamlessly interact with each other to produce the desired outcome. Let us consider our previous example of the retail shop billing system. It may interact with another system – an inventory system – to ensure that the warehouse keeps track of the stock of the material at hand. The protocol of how these two systems interact with each other are captured in the interface specification document.

POINTS TO PONDER

Functional requirements define what the system must do to fulfill the user's needs, nonfunctional requirements put forth the constraints on design and architecture, whereas interface requirements define how the system will interact with other external systems.

2.5 STEPS INVOLVED IN REQUIREMENTS ENGINEERING

Although the requirements engineering process may vary based on the application domain, a few generic steps are common across all types of software projects.

Requirements engineering includes requirements development and requirements management (Figures 2.6 and 2.7). The basic aims of the requirements engineering process are to provide a

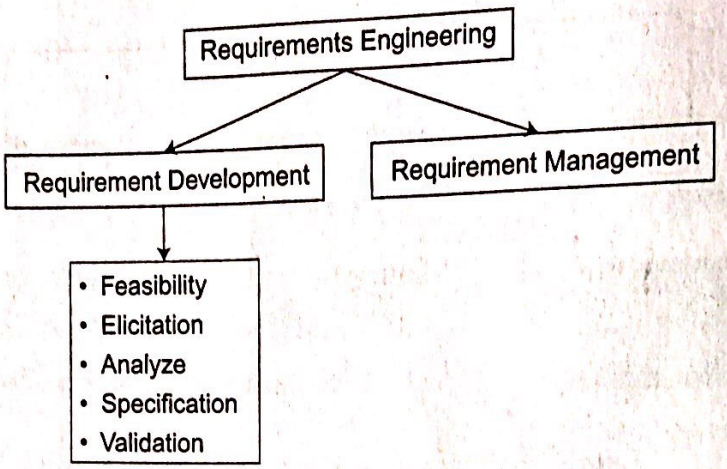


Figure 2.6 Requirements Engineering Steps

mechanism to understand the user's needs, analyze the need, assess the feasibility, specify and validate the requirements and proposed solution, and manage the requirements over the whole life cycle of the project, which are discussed in the following sections:

- Feasibility study
- Requirements elicitation
- Requirements analysis
- Specifying and validating requirements
- Requirements management

Figure 2.7 shows how each step of requirements engineering is related to each other.

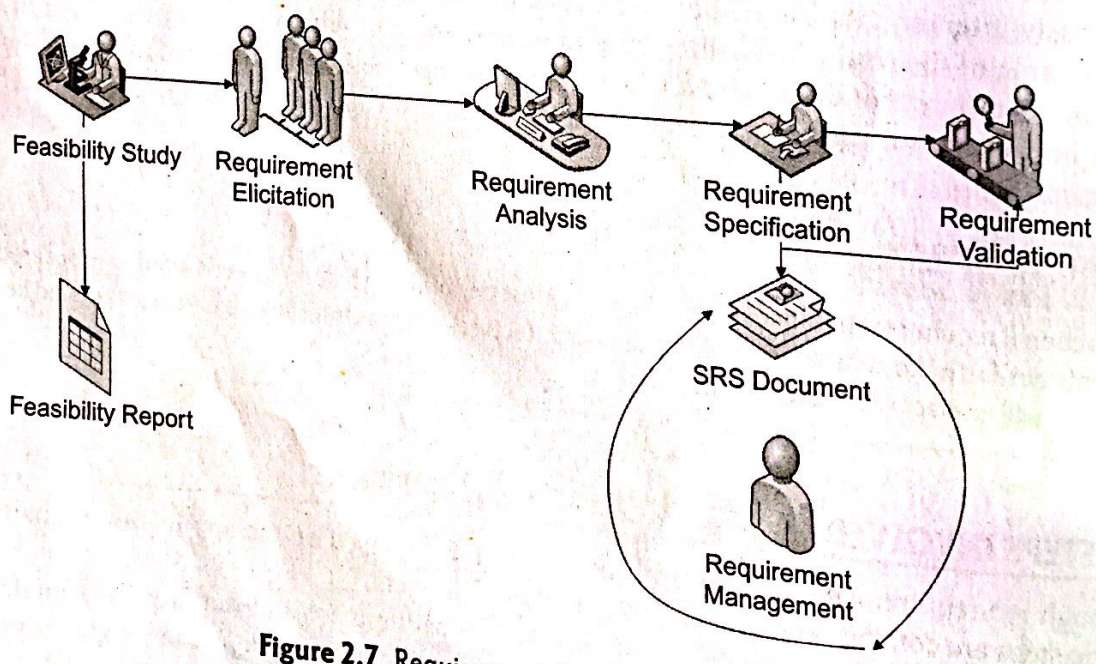


Figure 2.7 Requirements Engineering Process

2.5.1 Feasibility Study

Feasibility study is the first step of the requirements engineering process. The worthiness of the proposed software system should be determined before spending efforts on building the system.

A feasibility study (Figure 2.8) helps in deciding whether the proposed system is worthwhile and has the following characteristics:

- Whether the system contributes to organizational objectives
- Whether the system can be developed using the current available technology and within the specified time and budget
- Whether the system can easily be integrated with other surrounding systems as required by the overall architecture

Thus, feasibility can be classified into three broad categories (Figure 2.9): operational feasibility, technical feasibility and economic feasibility.

Operational feasibility: This checks the usability of the proposed software. If the operational scope is high, then the proposed system will be used more ensuring the acceptance from the sponsors of the project.

Technical feasibility: This checks whether the level of technology required for the development of the system is available with the software firm, including hardware resources, software development platforms and other software tools.

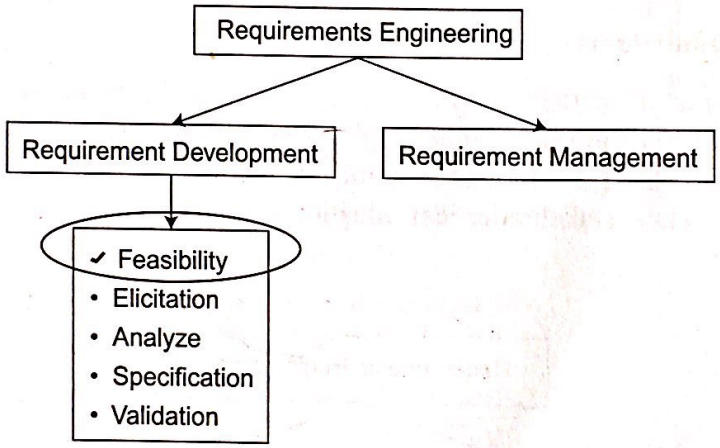


Figure 2.8 Requirements Feasibility

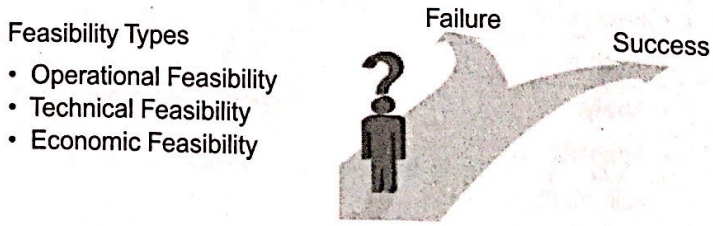


Figure 2.9 Feasibility Types

Economic feasibility: This checks whether there is scope for enough return by investing in this software system. All the costs involved in developing the system, including software licenses, hardware procurement and manpower cost, needs to be considered while doing this analysis. The cost-benefit ratio is derived as a result and based on the justification the stakeholders make a decision of going ahead with the proposed project.

POINTS TO PONDER

Before investing in any project, the sponsors will be keen to know whether the investment is worth and the project can succeed technically as well as economically. Thus, a feasibility study step provides an insight to all stakeholders at the very beginning.

2.5.2 Requirements Elicitation

Requirements elicitation is the second step of the requirements engineering process (Figure 2.10).

In this phase, the source for all the requirements are identified and then by using these sources, the user's needs and all the possible problem statements are identified. Although it looks simple, this phase is often iterative and at the end of each iteration the customer needs may become clearer and new needs may emerge. The stakeholders involved during this phase include end users, managers, development and test engineers, domain experts and business analysts.

2.5.2.1 Identifying Stakeholders

Before the requirements are gathered for a system, it is important to know the people who can contribute and help gather the requirements. If key stakeholders are not identified during the requirements elicitation phase, it can lead to nonidentification of important requirements causing rework at a later stage. Thus, stakeholder identification is the first step in starting the requirements elicitation.

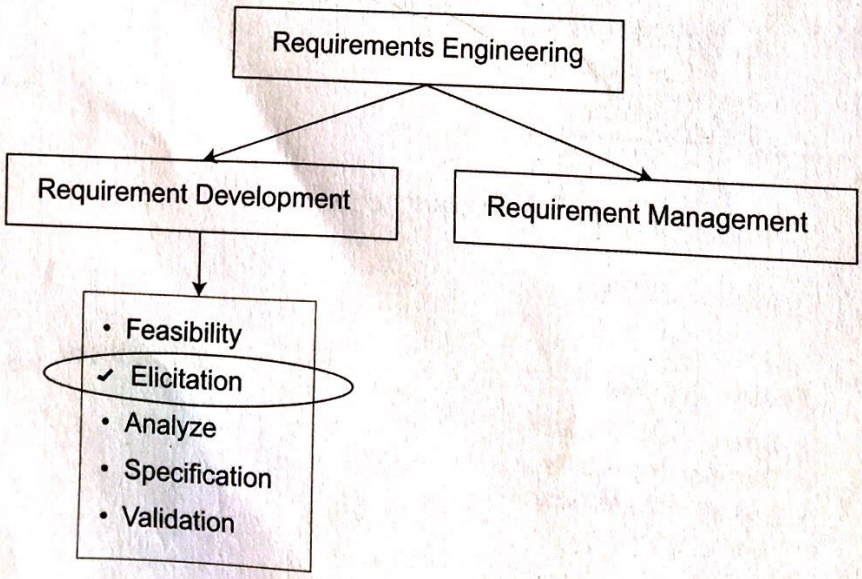


Figure 2.10 Requirements Elicitation

Stakeholders are people or entities (organizations) actively involved in the projects. They are affected by outcomes of the defined project.

Discussion Questions

Can you think of the important stakeholders for a medical insurance claim processing software development project?

1. _____
2. _____
3. _____
4. _____

2.5.2.2 Characteristics of Stakeholders

Stakeholder interests may be either positively or negatively impacted by the performance of the project. Stakeholders may have an influence on the project and its results. Thus, it is important for the project manager to identify all the stakeholders and their requirements (sometimes requirements may be implicit and not explicitly stated – the project manager should also try to understand such requirements). Stakeholders may have conflicting interests and objectives; therefore, managing them may not be easy. Involving stakeholders in the project phases improves the probability of success of the project.

Stakeholder involvement is always situation-specific; what works in one situation may not be appropriate in another. Ask a few simple questions to identify appropriate and required stakeholders. Although it is not intended to provide an exhaustive list of questions here, this will give you a fare idea of the type of questions appropriate for this purpose:

1. Who are the people likely to be affected/benefited?
2. Who is responsible for what we intend to do?
3. Who is likely to move for or against what is intended?
4. Who can make what is intended better?
5. Who can contribute to financial and technical resources?

POINTS TO PONDER

Properly identifying stakeholders and managing the stakeholder's expectation hold the key to success of any project. Maintaining a prioritized list of stakeholders according to their say in the project helps in avoiding the bottlenecks created by them at the later stages of the project.

2.5.2.3 Problems in Eliciting Requirements

There are several problems which can surface while eliciting requirements, which are discussed below (Figure 2.11).

- **Users not sure about the requirements:** This situation arises very often when the system to be developed is a completely new system and there is no corresponding manual system or any

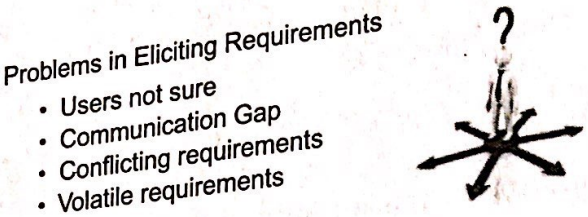


Figure 2.11 Problems in Eliciting Requirements

previous software system that can be analyzed to get the requirement. In this situation, the users will be looking for either similar systems developed elsewhere or mock-up screens, workflows, prototypes, etc. during the requirements gathering stage, which will aid their imagination in coming up with the requirements for the new system.

- **Communication gap:** Even if the end users and stakeholders are clear about the need for developing the new system, they may find it difficult to express it in a concrete manner due to their less exposure to the computing systems. They may state the requirements in an ambiguous or nontestable fashion. Sometimes the obvious basic requirements may be omitted and they concentrate on explaining the peripheral requirements. The system engineer needs to drive the stakeholders in the right direction throughout the requirements gathering stage so that they focus only on the most important requirements.
- **Conflicting requirements:** This problem increases with the number of stakeholders involved in the project. There may be conflicting needs and priorities for each stakeholder based on the business areas they are covering. For example, a stakeholder from the marketing team will try to provide requirements that will expedite the creation of new products, while the end users of the system will look for efficient screens that help easy data capture. During the analysis phase, the requirements analyst and the client should collaboratively discuss and resolve any conflicting requirements from multiple stakeholders.
- **Volatile requirements:** Requirements may get changed during the elicitation stage due to the entry of new stakeholders who have different perspectives of the system. Although this is helpful in clarifying the requirements better, it sometimes creates friction between the requirements engineer and the stakeholders due to continuous change in the scope.

Discussion Questions

Suppose you are the end user of a mainframe-based banking software that is going to be replaced by a new Java-based software. The requirements engineers from this new company have approached you to understand the requirements for the new system? How you will ensure that they have noted down all your requirements correctly?

1. _____
2. _____
3. _____
4. _____

2.5.2.4 Requirements Elicitation Techniques

Several techniques (Figure 2.12) can be employed for eliciting the requirements as listed below. Based on the project scope, domain, customer preparedness, etc., one or a combination of techniques needs to be used for gathering the requirements in an organized manner.

- Interviewing
- Focus groups
- Facilitated workshops
- Prototyping
- Questionnaires
- Brainstorming
- Direct observation
- Apprenticing

Interviewing Begin the interviews with the stakeholders who are believed to have complete understanding of the requirements. Face-to-face interactions with users through individual interviewing is the primary source of requirements (Figure 2.13).

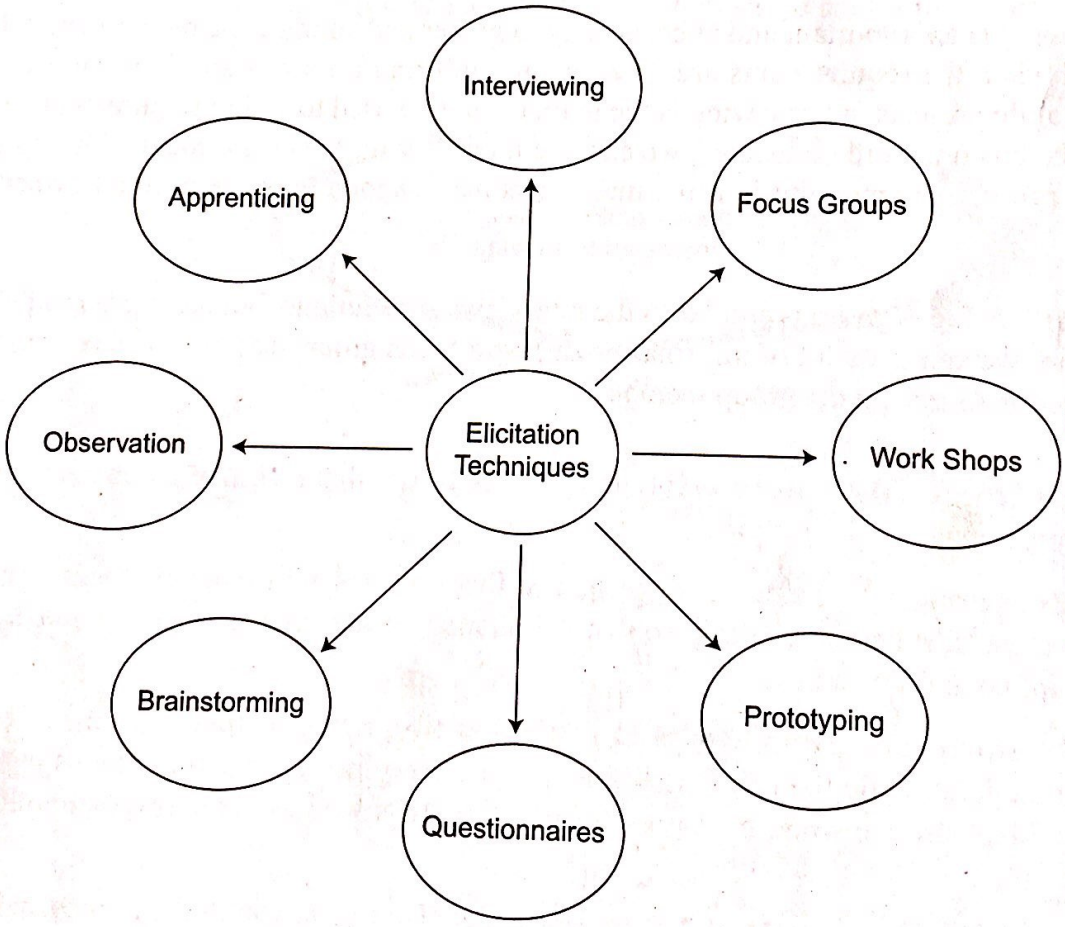


Figure 2.12 Requirements Elicitation Techniques

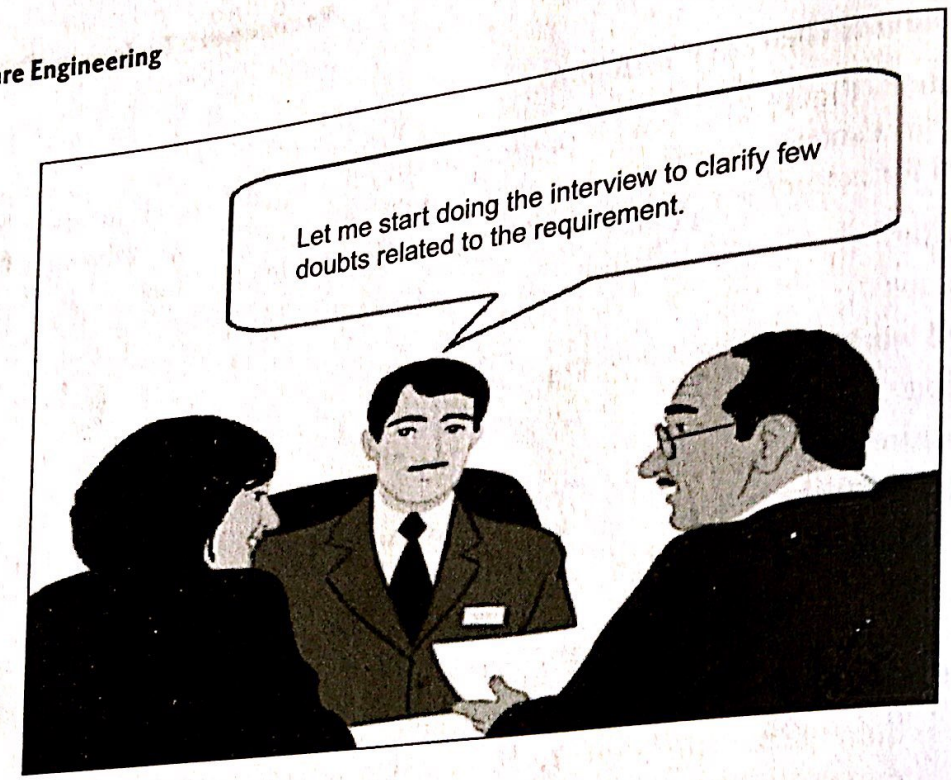


Figure 2.13 Interviewing Technique

Interviewing is an important and effective way to gather and validate the requirements. Interviewing is used when the requirements are detailed and differing opinions are likely or are sought. In distributed agile projects, interviewing is the main technique used to gather requirements. Even if the team and the customer are collocated, we can use interviewing to clarify doubts. During the execution of the project, at any point in time, interviews may happen with the product owner to clarify doubts.

Focus Groups A focus group is similar to the interviewing technique, but has a group of 6 to 10 people at a time. We can get a lot of information during a focus group session because of the enhanced level of discussion among the group members.

Types of Focus Groups There are several types of focus groups that facilitate group discussions, which are discussed below.

Two-way focus group: As the name suggests, two focus groups are used. One focus group will continuously watch the other focus group so that the interaction happens as per the predefined rules and proper discipline is followed.

Dual moderator focus group Model 1: Instead of two different focus groups as in the above model, two moderators do the job. One looks into the discipline and the other looks into the content flow. This will ensure the smooth progress of the session as well as the entire topic of discussion is covered.

Dual moderator focus group Model 2: In this model also two moderators are present but they take completely opposite stands, that is, if one says that something is possible, the other says why it is not possible. This is more helpful in finding the pros and cons of both views and will be helpful to make a final decision.

Respondent moderator focus group: Respondents are asked to act as the moderator temporarily and because of this they take ownership of the session. Hence, we can collect the requirements successfully and the outcome will be helpful (Figure 2.14).

Client participant focus group: Client representatives participate in the discussion and will ensure ownership from the client on the decision taken. Most of the projects fail at the last stage of the project life cycle, particularly in the User Acceptance Testing phase. The reason being the client may not take ownership as the end users have more ownership at that stage. However, client participant focus groups help overcome these kinds of problems.

Mini focus group: Groups are restricted to four or five members rather than 8 to 12 and is helpful to avoid confusion and manage and gather the requirements quickly.

Teleconference focus group: Telephone network is used to facilitate the discussion sessions.

Online focus group: Computers connected via the internet are used to facilitate the discussion sessions.

Facilitated Workshops Workshops (Figure 2.15) can be used for rapidly pulling together a good set of requirements. A workshop is a very quick and best way to gather requirements compared with all other techniques. A workshop is expensive because it involves many people, but it saves a large amount of time. Requirements are discussed at a high level. Workshops are useful in situations where

Figure 2.14 Focus Group Technique

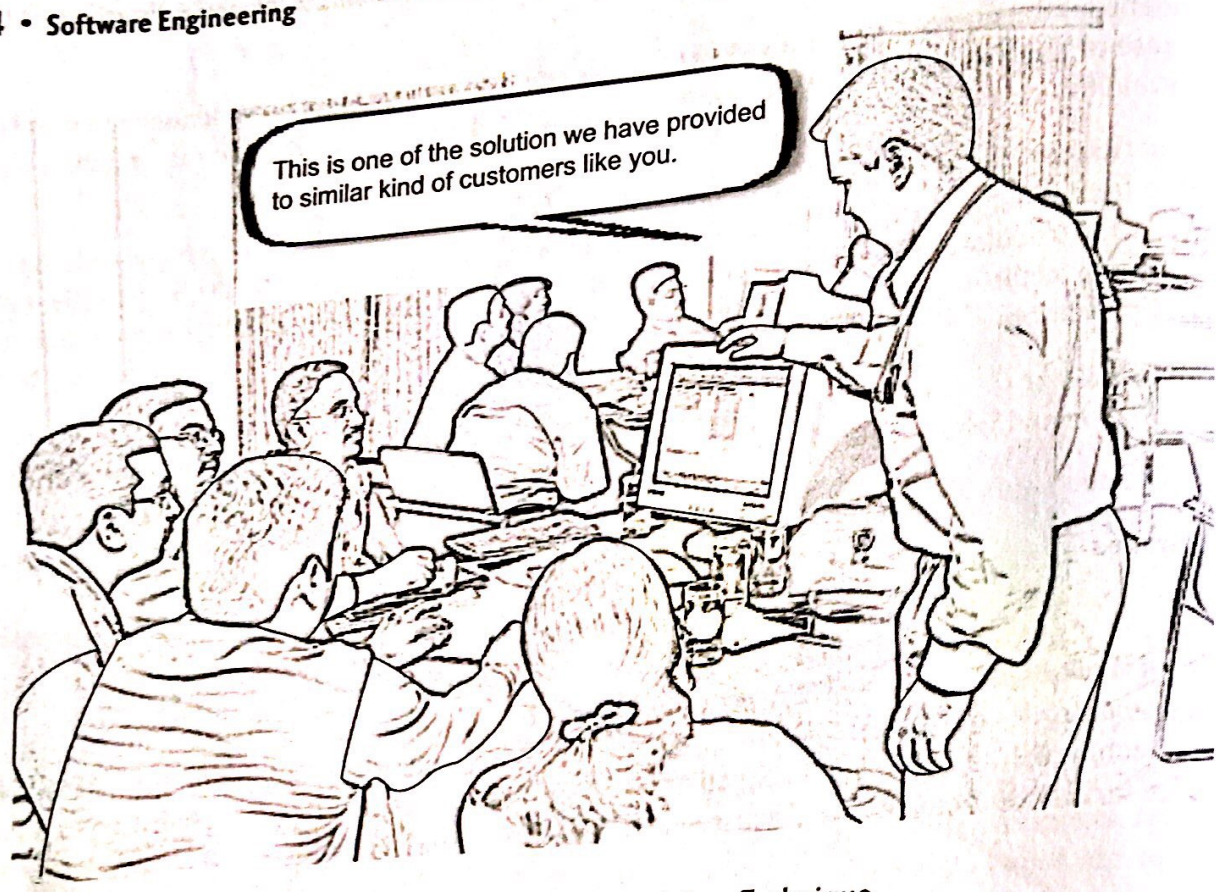


Figure 2.15 Facilitated Workshop Technique

the requirements are focused on one area of business in which the participants have knowledge and consensus is being sought. In workshops, different alternatives are given to the customers so that they choose the appropriate options.

Prototyping The word "prototype" is derived from the Greek word "prototypon." Prototypes and models are the best ways of presenting ideas to users. They give users a glimpse of what they might get. More requirements are likely to emerge when users are able to see what they will get as an outcome of their suggestion. This technique aims to get users to express their requirements. Prototyping is used to get feedback from users. Business logic may not be coded in prototyping and experimental systems are developed using prototyping. It is actually the initial version of the system.

POINTS TO PONDER

During the requirements phase, the prototype is generally document-based, where the user interface design, use cases, overall look and feel, navigation, etc. are discussed with the end users. The scope of a working prototype normally comes at the construction phase.

Types of Prototypes As discussed in Chapter 1, prototypes are classified into two broad categories, namely, evolutionary prototype and throw away prototype. The former type is reused and the changes are implemented in that particular prototype. The new changes and corrections are updated in the

existing prototype until the final list of requirements is obtained, without wasting the used resources. The latter type, as the name suggests, destroys the wrong prototypes. Throw away prototypes are proven to be wrong and thus the new prototypes are developed in the subsequent stages. The resources used in the previous prototypes are either freed or rejected.

Figure 2.16 shows the different types of prototypes, which are discussed below.

Proof-of-principle prototype (bread board): Only the intended design is tested and visual appearance is not in the scope of this prototype. For understanding purposes, a few working codes are part of this prototype.

Form study prototype: Only visual appearance is considered and the functionalities are not considered.

Visual prototype: It simulates the appearance, color, fonts and surface textures of the intended product, but it does not represent of the final function(s) of the final product.

Functional prototype: It is also called a working prototype, which means it simulates the actual functionality of the intended work.

Questionnaires Questionnaires can be used to collect input from multiple stakeholders quickly, which can be consolidated to create a list of requirements. As the stakeholders targeted for the questionnaire need not be physically present with the requirements elicitation team, this process can be run with a large number of participants in quite an inexpensive fashion.

However, the questionnaires have their own limitations such as designing an exhaustive questionnaire becomes difficult especially if the complexity of the system is high. As the stakeholders are not in direct contact with the elicitation team, there may be ambiguity in questions that lead to erroneous responses. Moreover, the response rate may be low and may need a lot of follow-up for getting sufficient input through this technique.

Brainstorming Brainstorming is a powerful technique using which a large number of requirements or ideas can be generated in a short period of time.

- In these sessions, members meet face-to-face and rely on both verbal and nonverbal interactions to communicate with each other.
- The main purpose of the brainstorming technique is to get as many ideas as possible to obtain different views of the requirements, thereby helping to capture better requirements.

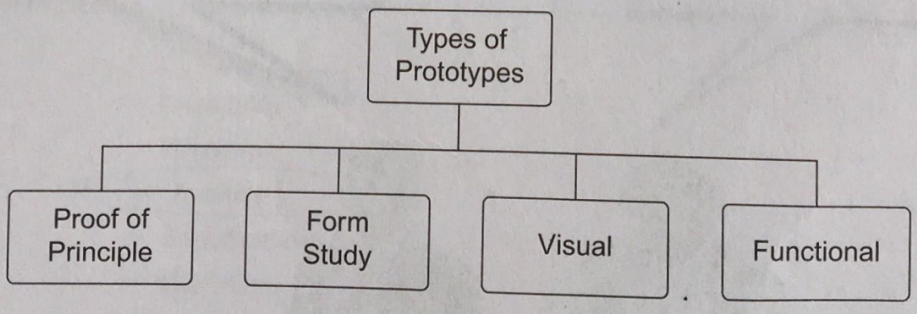


Figure 2.16 Types of Prototypes

In a typical brainstorming session, about six people are involved and the group leader states the problem in a clear and understandable manner so that it is understood by all the participants involved. Members are allowed to suggest as many alternatives as they can in a given period of time. No criticism is allowed. Alternative ideas generated are also recorded and are used for later analyses and discussions. Brainstorming, however, is merely a process used to generate ideas and may not be effective for decision-making purposes.

Discussion Questions

Do you think the brainstorming sessions can help in a situation where the end users do not have a clear idea of the features they want from the software to be developed?

1. _____
2. _____
3. _____
4. _____

Direct Observation and Apprenticing In many cases a new system is developed to replace an existing system, in which either a lot of manual intervention is involved or it is not efficient and suitable for doing the job. In such scenarios, it is very useful to gather knowledge of the requirements by observing (Figure 2.17) what is actually done by the user of the system. This is far more powerful than going through the documents of the capability of the existing system as it provides an opportunity to capture the real objective behind creating the new system.

A better technique will be the apprenticing model where the requirements engineer performs the tasks that the user of the system carries out. Although this requires the understanding of the business

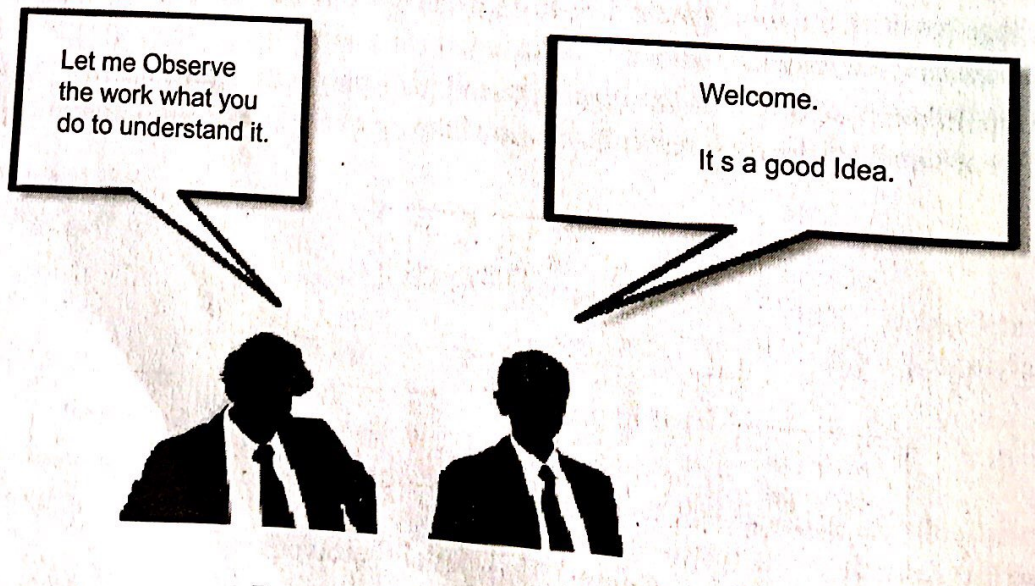


Figure 2.17 Observation Technique