



ebook

SOFTWARE ENGINEERING DSE

**SOFTWARE REQUIREMENT
ENGINEERING**

Lesson 03 – Software Requirement Engineering

Introduction to Software Requirements Engineering

Understanding the requirements of a problem is among the most difficult tasks that face by a Business Analysis and Software Engineer.

The problem is after completing the software,

- Is the system meet the requirements?
- Are the end users have a good understanding of the features and functions?

Requirement engineering establishes a solid base for design and construction. Without it, the resulting software has a probability of not meeting customer's needs.

The following images explain what will happen if you gather the requirements wrongly,

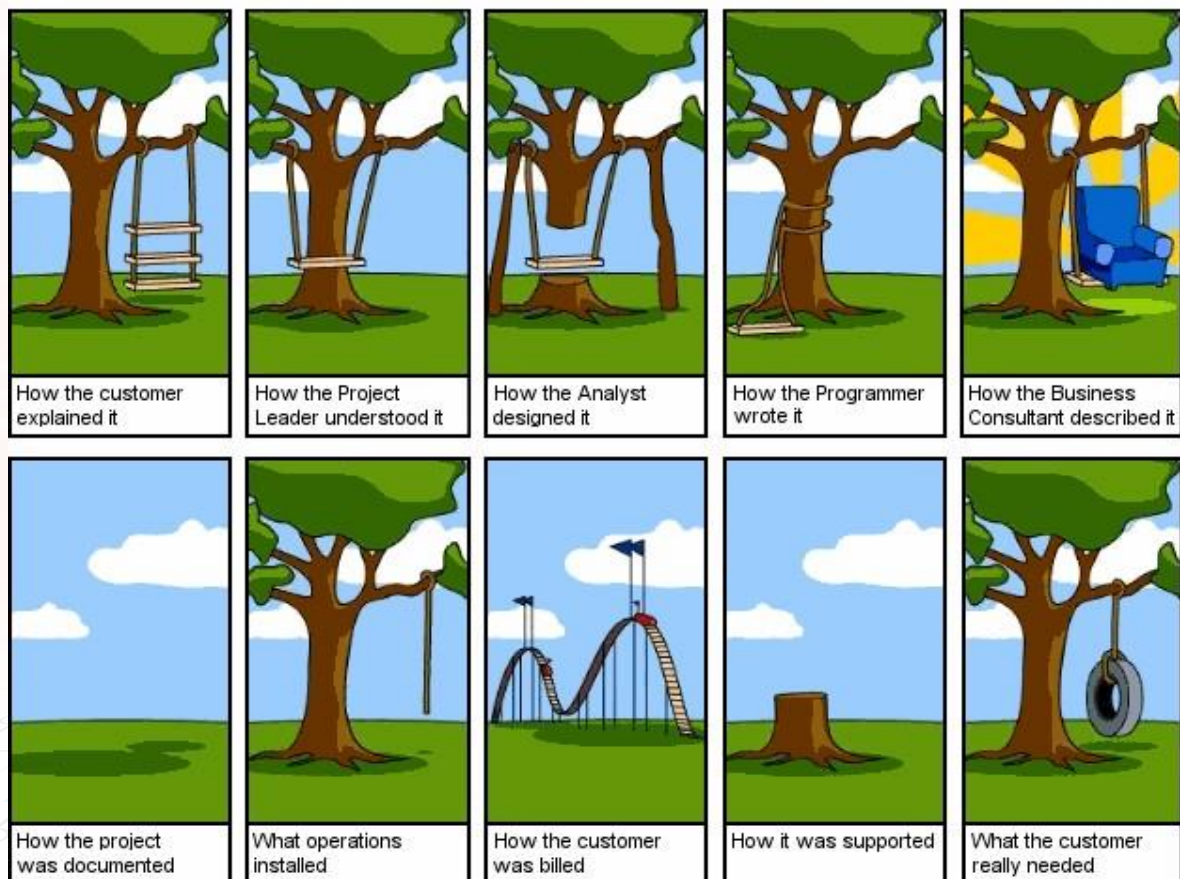


Figure 3.0.1 Wrong Requirement Gathering - 1

Software Engineering



Figure 3.0.2 Wrong Requirement Gathering - 2

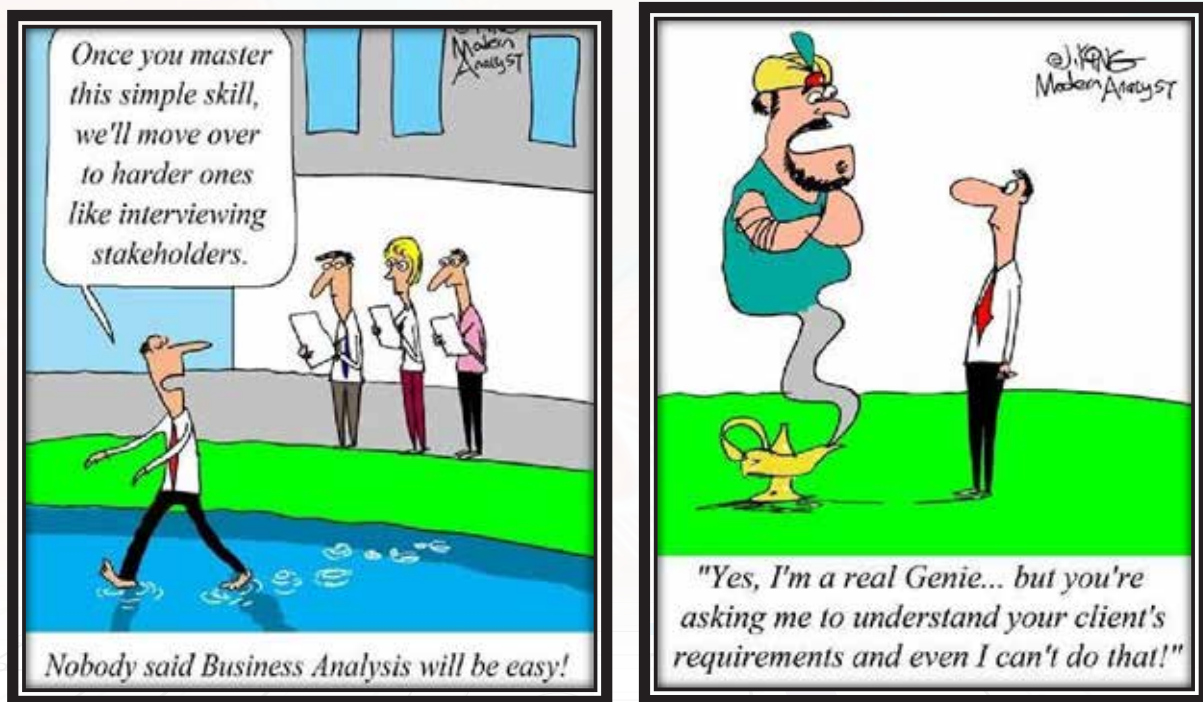


Figure 3.0.3 Wrong Requirement Gathering - 3

Software Engineering



Figure 3.0.4 Wrong Requirement Gathering - 4

Types of Requirements

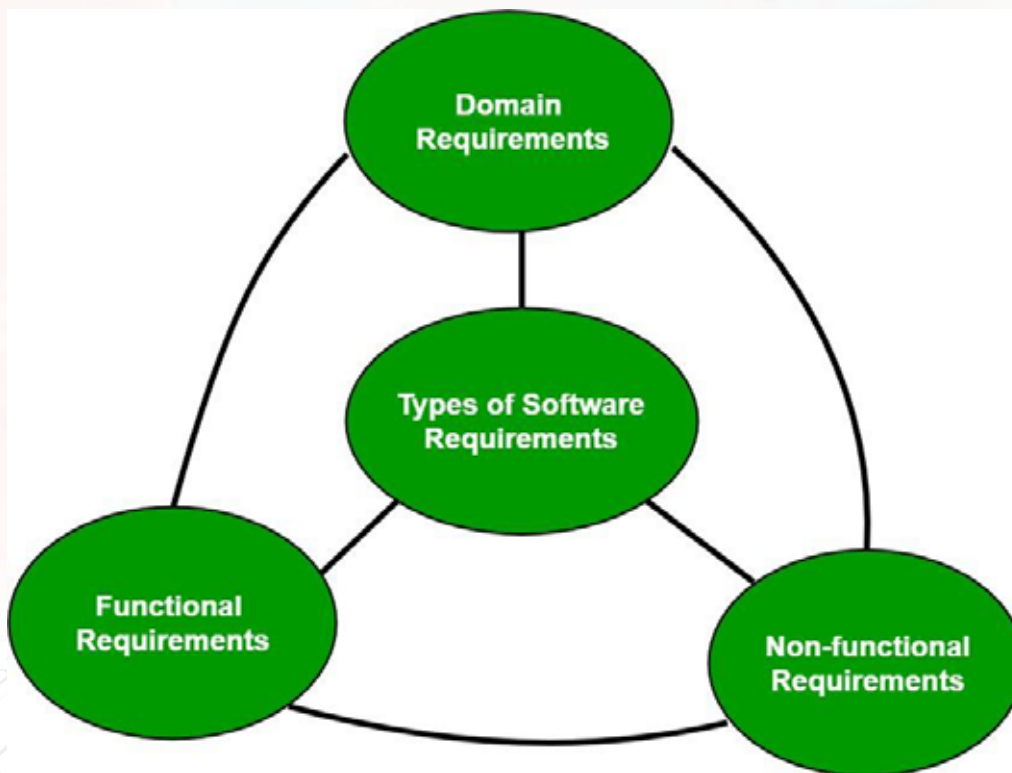


Figure 3.0.5 Types of Requirements

Software Engineering

Functional Requirements

- Functionality or services that the system is expected to provide.
- How the System should react to particular inputs and how the system should behave in particular situations.
- Functionality depends on the type of software, expected users and the type of system where the software is used.

E.g.: The user shall be able to search the relevant data which he needed.

Non-functional Requirements (Quality Requirements)

- Define system properties and constraints such as reliability, response time, interface, security, and storage requirements.
- Non-functional requirements may be categorized in to product requirements, organizational requirements and external requirements.

Non-functional requirements may be more critical than functional requirements.

Types of Non Functional Requirements

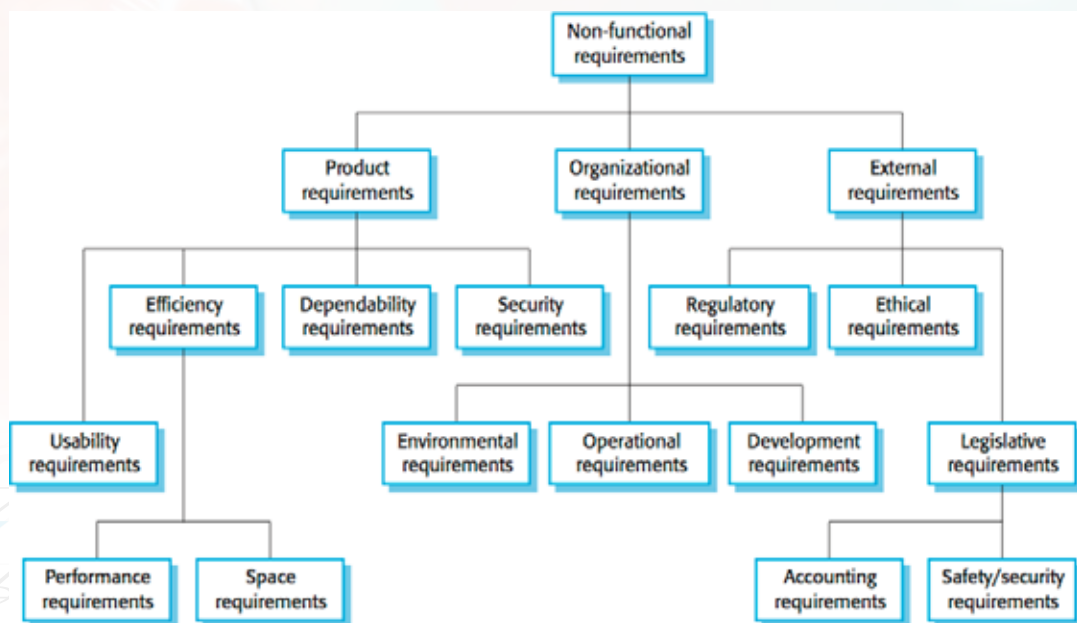


Figure 3.0.6 Types of non-functional Requirements

Software Engineering

Product Requirements

Requirements which specify that the delivered product must behave in a particular way.

- Performance Requirements
 - Describe the extent to which the software makes optimal use of resources.
- Reliability Requirements
 - Describe the acceptable failure rate of the software.
- Portability Requirements
 - Describe the ease with which the software can be transferred from one platform to another.
- Interface Requirements
 - Describe the user interface of the software.
- Usability Requirements
 - Describe the ease with which users are able to operate the software.
- Security Requirements
 - Describe the major security areas in the software.

Organizational Requirements

Requirements which are a consequence of organizational policies and procedures like process standards used and implementation requirements.

- Delivery Requirements
 - Specify when the software and its documentation are to be delivered to the user.
- Implementation Requirements
 - Describe requirements such as programming language and design method.
- Standards Requirements
 - Describe the process standards to be used during software development.

Software Engineering

External Requirements

Requirements which arise from factors which are external to the system and its development process.

- Interoperability requirements:

Define the way in which different computer based systems will interact with each other in one or more organizations.

- Ethical requirements:

Specify the rules and regulations of the software so that they are acceptable to users.

- Legislative requirements:

Ensure that the software operates within the legal jurisdiction.

Domain Requirements

Requirements come from the application domain of the system and that reflect characteristics of that domain. They may be new functional or non-functional requirements in their own right and constrain existing requirements or set out how particular computation must be carried out.

E.g.: The requirements of the user such as copyright restrictions and security mechanism for the files and documents used in the system are also domain requirements.

Requirement Engineering Process

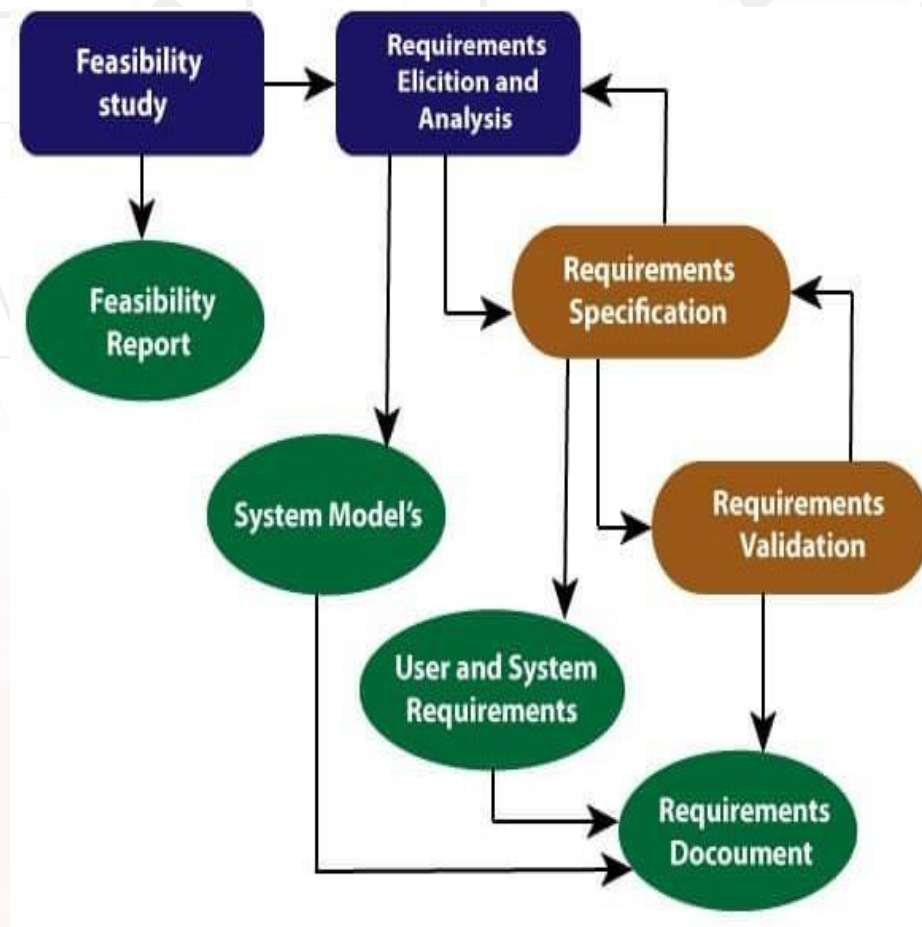
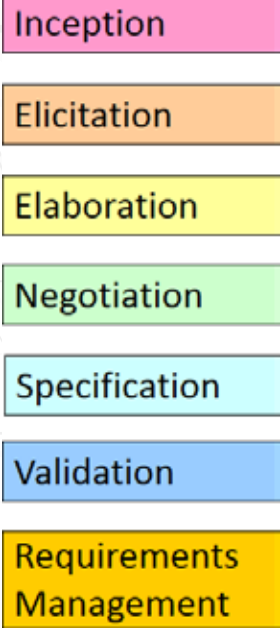


Figure 3.0.7 Requirement Engineering Process

Requirement Engineering Tasks inside the Process

- Some of these tasks may occur in parallel and all are adapted to the needs of the project
- All strive to define what the customer wants
- All serve to establish a solid foundation for the design and construction of the software

Software Engineering

 Inception

Elicitation

Elaboration

Negotiation

Specification

Validation

Requirements
Management

1. Inception (Feasibility Study)

- At project inception you establish a basic understanding of the problem, the people who want a solution, the nature of the solution that is desired.
- When initiating requirements engineering task, the requirements engineer needs to...
 - Identify the stakeholders (Users, Owners, Administrators)
 - Recognize multiple viewpoints
- The output of this phase should be a **Feasibility Study Report** that should contain adequate comments and recommendations for management about whether or not the project should be undertaken.

2. Elicitation

- If the feasibility report is positive towards undertaking the project, next phase starts with gathering requirements from the user.
- Ask from the Customer, Users and Others
 - What are the objectives for the system or products are?
 - What is to be accomplished?

Software Engineering

- How the system or products fits into the needs of the business?
- How the system or product is to be used on a day-to-day basis?

Elicitation Techniques (Requirements Gathering Techniques)

- Brainstorming
- Record Review (Document Analysis)
- Focus Group
- Questionnaires (Survey)
- Observations
- Interview
- Collaborative Requirement Gathering (Joint Application Development)
- Quality Function Deployment
- Prototyping

Brainstorming

Brainstorming can be utilized in requirements gathering to gather a good number of ideas from a group of people. Usually brainstorming is used in identifying all possible solutions to problems and simplifies the detail of opportunities.

Record Review (Document Analysis)

Reviewing the current process and documentation can help the analyst understand the business, or system, and its current situation. Existing documentation will provide the analyst the titles and names of stakeholders who are involved with the system.

This will help the analyst formulate questions for interviews or questionnaires to ask of stakeholders, in order to gain additional requirements.

Focus Group

A focus group is a gathering of people who are customers or user representatives for a product to gain its feedback. The feedback can be collected about opportunities, needs, and problems to determine requirements or it can be collected to refine and validate the already elicited requirements.

Software Engineering

Questionnaires (Survey)

When gathering information from many people: too many to interview with time constraints and less budget: a questionnaire survey can be used. The survey insists the users to choose from the given options agree / disagree or rate something.

Observations

The observation covers the study of users in its natural habitat. By watching users, a process flow, pain points, awkward steps and opportunities can be determined by an analyst for improvement.

Interview

Interviews of users and stakeholders are important in creating wonderful software. Without knowing the expectations and goal of the stakeholders and users, you are highly unlikely to satiate them. You also have to understand the perspective of every interviewee, in order to properly address and weigh their inputs. **Listening is the most important thing here.**

Collaborative Requirement Gathering (Joint Application Development)

Popularly known as JAD or joint application design, these workshops can be efficient for gathering requirements. The requirements workshops are more organized and structured than a brainstorming session where the involved parties get together to document requirements.

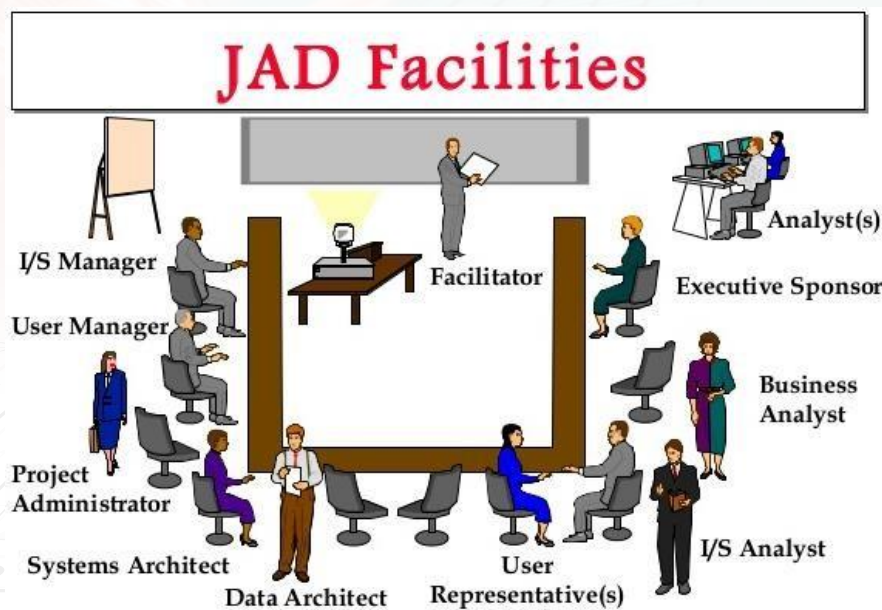


Figure 3.0.8 Joint Application Development

Software Engineering

Prototyping

Prototyping can be very helpful at gathering feedback. Low fidelity prototypes make a good listening tool. Many a times, people are not able to articulate a specific need in the abstract. They can swiftly review whether a design approach would satisfy the need. Prototypes are very effectively done with fast sketches of storyboards and interfaces.

3. Elaboration

- During elaboration, the software engineer takes the information obtained during inception and elicitation and begins to expand and refine it.
- Elaboration focuses on developing a refined technical model of software functions, features, and constraints.

4. Negotiation

- During negotiation, the software engineer reconciles the conflicts between what the customer wants and what can be achieved given limited business resources.
- Requirements are ranked (i.e., prioritized) by the customers, users, and other stakeholders.

5. Specification

- The Software Requirement Specification (SRS) is the final work product produced by the Business Analysis or requirements engineer.
- It describes the function and performance of a computer-based system and the constraints that will govern its development.
- It formalizes the informational, functional, and behavioral requirements of the proposed software in both a graphical and textual format.
- It is not a design document. As far as possible, **it should set of WHAT the system should do** rather than HOW it should do.

Software Engineering

Software Requirements Specification



Figure 3.0.9 Software Requirements Specification

Software Requirement Specification (SRS) A Template (Karl Wigers)

1. Introduction

- 1.1 Purpose of the requirements document
- 1.2 Scope of the product
- 1.3 Definitions, acronyms and abbreviations
- 1.4 References
- 1.5 Overview of the remainder of the document

2. General Description

- 2.1 Product Perspective
- 2.2 Product functions

Software Engineering

2.3 User Classes and Characteristics

2.4 Operating Environment

2.5 Design and Implementation Constraints

2.6 User Documentation

2.7 Assumptions and Dependencies

3. System Features (Specific requirements)

3.1 System Feature 1

3.2 System Feature 2

3.3 System Feature 3 (and so on)

4. External Interface Requirements

4.1 User Interface

4.2 Hardware Interfaces

4.3 Software Interfaces

4.4 Communications Interfaces

5. Other Nonfunctional Requirements

5.1 Performance Requirements

5.2 Safety Requirements

5.3 Security Requirements

5.4 Software Quality Attributes

6. Other Requirements

Appendix A: Glossary

Appendix B: Analysis Models

Software Engineering

Writing an SRS in Microsoft Word vs. Helix ALM

You can write your software requirement specification in Microsoft Word. But if a requirement changes, your SRS can fall easily out-of-date. Plus, there can be versioning issues with requirements documents in Word. You can save time — and ensure accuracy — by writing an SRS in **Helix ALM**.

Helix ALM, formerly called **TestTrack**, is application lifecycle management



HelixALM

(ALM) software developed by Perforce. The software allows developers to manage requirements, defects, issues and testing during software development.

6. Validation

- During validation, the work products produced as a result of requirements engineering are assessed for quality.
- The specification is examined to ensure that,
 - All the software requirements have been stated unambiguously.
 - All the inconsistencies, omissions, and errors have been detected and corrected.
 - All the requirements should be consistent with all the other requirements i.e no two requirements should conflict with each other.
 - All the requirements should be practically achievable.

7. Requirements Management

- Requirement management is the process of analyzing, documenting, tracking, prioritizing and agreeing on the requirement and controlling the communication to relevant stakeholders.
- Each requirement is assigned a unique identifier.
- The requirements are then placed into one or more traceability tables.
- These tables may be stored in a database that relate features, sources, dependencies, subsystems, and interfaces to the requirements.

Software Engineering

What is Requirement Traceability Matrix?

- Requirement Traceability Matrix (RTM) is a document that maps and traces user requirement with test cases.
- It captures all requirements proposed by the client.
- The main purpose of Requirement Traceability Matrix is to validate that all requirements are checked via test cases such that no functionality is unchecked during Software testing.

Req No	Req Desc	Testcase ID	Status
123	Login to the application	TC01,TC02,TC03	TC01-Pass TC02-Pass
345	Ticket Creation	TC04,TC05,TC06, TC07,TC08,TC09 TC010	TC04-Pass TC05-Pass TC06-Pass TC06-Fail TC07-No Run
456	Search Ticket	TC011,TC012, TC013,TC014	TC011-Pass TC012-Fail TC013-Pass TC014-No Run

Figure 3.0.10 Requirement Traceability Matrix