Software Requirements:

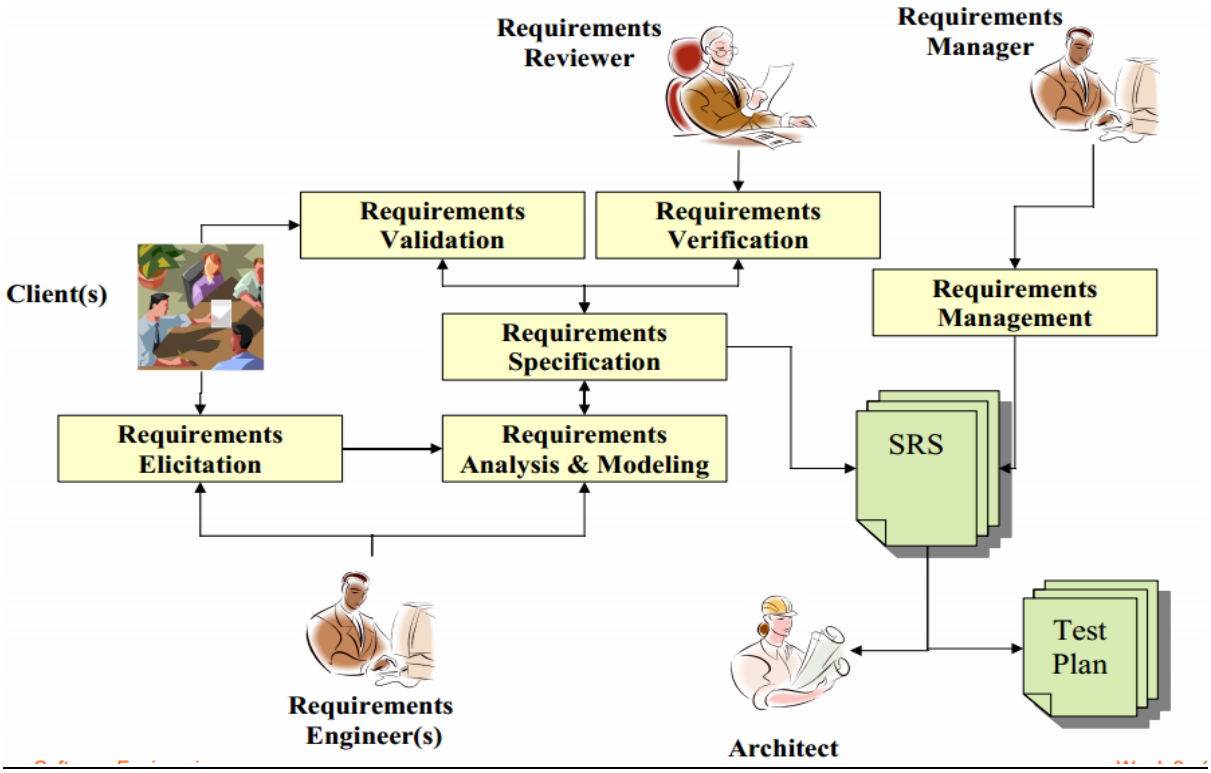
The software requirements are description of features and functionalities of the target system. The process of gather the software requirements from client, analyze and document them is known as requirement engineering.

The goal of requirement engineering is to develop and maintain sophisticated and descriptive document 'System Requirements Specification'.

Software Requirements Objectives:

1. To introduce the concepts of user and system requirements.
2. To describe functional and non-functional requirements.
3. To explain how software requirements may be organised in a requirements document.

Requirements Engineering Process:

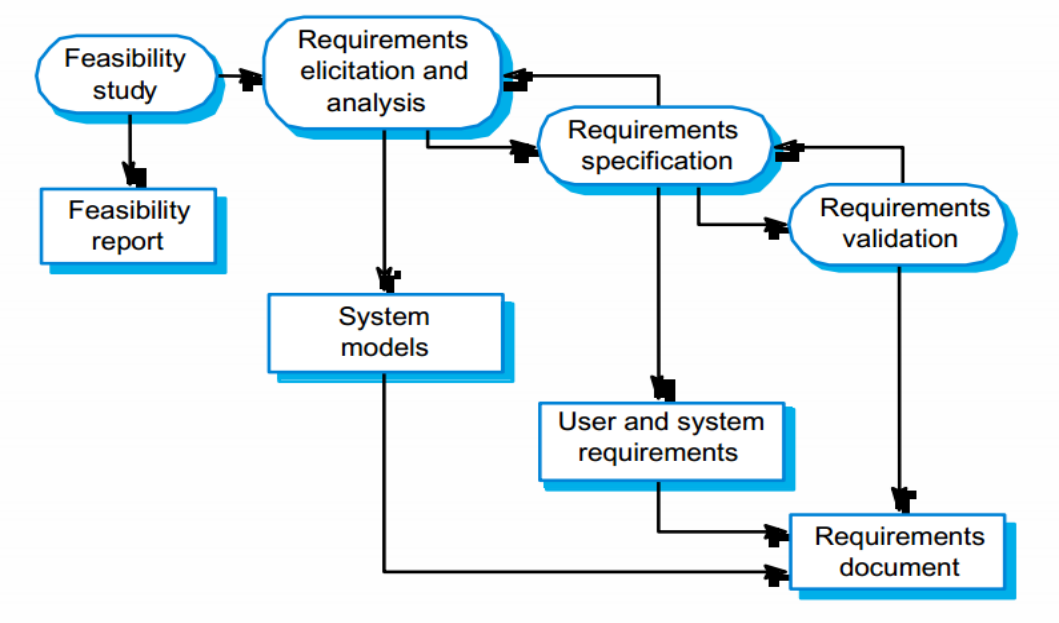


The differences between Verification and Validation:

They are Two key processes in software quality assurance.

Aspect	Verification	Validation
Definition	Ensures that the software is being built correctly (i.e., follows specifications, requirements, and design).	Ensures that the right software is being built (i.e., meets user needs and expectations).
Focus	Process-oriented (Are we building the product right?).	Product-oriented (Are we building the right product?).
Purpose	To confirm that the software conforms to predefined requirements, design documents, and standards.	To confirm that the final product meets customer expectations and business needs.
Timing	Done during development (before testing and deployment).	Done after development, usually during or after testing.

The Requirements Engineering Process:



Feasibility Study:

A feasibility study is the process of analyzing and evaluating a project or idea to determine its technical, financial, economic, legal, and operational viability. The goal of this study is to help decision-makers assess whether the project is worth investing in and pursuing.

Types of Feasibility Studies

- Technical Feasibility – Can the project be implemented from a technical perspective?
  Financial Feasibility – Is the project profitable? Is the budget sufficient?
  Economic Feasibility – How will the project impact the local or general economy?

- Legal Feasibility – Does the project comply with laws and regulations?
  Operational Feasibility – Can the project be practically implemented? Is there demand for it?

Types of Requirements:

1. User requirements.

   - Statements in natural language plus diagrams of the services the system provides and its operational constraints (the requirements of the system's customers or end-users).

   - Written for customers.

2. System requirements.

   - A structured document setting out detailed descriptions of the system services (are the requirements for the system as a whole).

   - Written as a contract between client and contractor.

User Requirements:

- Should describe functional and non-functional requirements so that they are understandable by system users who don't have detailed technical knowledge.

- User requirements are defined using natural language, tables and diagrams as these can be understood by all users.

System Requirements:

- They are more detailed specifications of user requirements.
- They are intended to be a basis for designing the system.
- May be used as part of the system contract.

The Taxonomy of Requirements:

Software system requirements are classified as:
1. Functional requirements.
2. Non-Functional requirements.

Functional Requirements: Statements of:
• services the system should provide,
• how the system should react to particular inputs and
• how the system should behave in particular situations.

They describe functionality or system services and depend on
• the type of software,
• expected users and
• the type of system where the software is used.

Note: Functional user requirements may be high-level statements of what the system should do but functional system requirements should describe the system services in detail.

Non-Functional Requirements:

Requirements not directly concerned with the specific functions delivered by the system. They may relate to emergent system properties. Constraints on the services or functions offered by the system such as: timing constraints and constraints on the development process.

They define:

- System properties, e.g.
  - Reliability.
  - Response time.
  - Storage requirements.
- System constraints, e.g.
  - I/O device capability.
  - System representations.

Non-functional requirements may be more critical than functional requirements. If these are not met, the system is useless.

Requirement Elicitation:

Requirements elicitation is concerned with where software requirements come from and how the software engineer can collect them. It is the first stage in building an understanding of the problem in the software which is required to solve. It is fundamentally a human activity, and is where the stakeholders are identified and relationships established between the development team and the customer.

Stakeholders :

stakeholders are individuals, groups, or organizations that have an interest in the development, implementation, and success of a software system. They can either influence the project or be affected by it.

Types of Stakeholders:

Internal Stakeholders – Within the development organization.

Project Managers (Oversee the project timeline and budget)

Developers (Write the code and build the system)

Testers (Ensure software quality and performance)

External Stakeholders – Outside the development team.

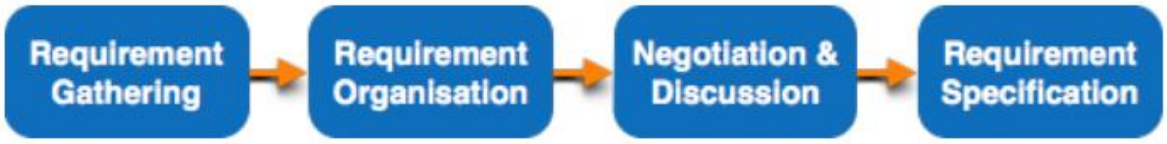
Customers (Buy or use the software)

Investors (Fund the project for profitability)

Third-party Vendors (Provide tools, APIs, or integrations)

Requirement Elicitation Process:

Requirement elicitation process can be depicted using the following diagram:



- Requirements gathering - The developers discuss with the client and end users and know their expectations from the software.
- Organizing Requirements - The developers prioritize and arrange the requirements in order of importance, urgency and convenience.
- Negotiation & discussion - If requirements are ambiguous or there are some conflicts in requirements of various stakeholders, if they are, it is then negotiated and discussed with stakeholders. Requirements may then be prioritized and reasonably compromised.
- Documentation - All formal & informal, functional and non-functional requirements are documented and made available for next phase processing.

Software Requirement Specification (SRS):

Also called software requirement document (SRD) which is a document created by system analyst after the requirements are collected from various stakeholders. SRS defines how the intended software will interact with hardware, external interfaces, speed of operation, response time of system, portability of software across various platforms, maintainability, speed of recovery after crashing, Security, Quality, Limitations etc.

The requirements received from client are written in natural language. It is the responsibility of system analyst to document the requirements in technical language so that they can be comprehended and useful by the software development team.

- SRS should come up with following features:
  - User Requirements are expressed in natural language.
  - Technical requirements are expressed in structured language, which is used inside the organization.
  - Design description should be written in Pseudo code.
  - Format of Forms and GUI screen prints.
  - Conditional and mathematical notations for DFDs etc.

A complete Software Requirement Specifications must be:

1. Clear
2. Correct
3. Consistent
4. Coherent
5. Comprehensible
6. Modifiable
7. Verifiable
8. Prioritized
9. Unambiguous
10. Traceable
11. Credible source

Software Requirement Validation:

After requirement specifications are developed, the requirements mentioned in this document are validated. User might ask for illegal, impractical solution or experts may interpret the requirements incorrectly. This results in huge increase in cost if not nipped in the bud.

The requirements may be validated to ensure that the software engineer has understood the requirements, and it is also important to verify that a requirements document conforms to company standards, and that it is understandable, consistent, and complete.

Requirements can be checked against following conditions:-

- If they can be practically implemented
- If they are valid and as per functionality and domain of software

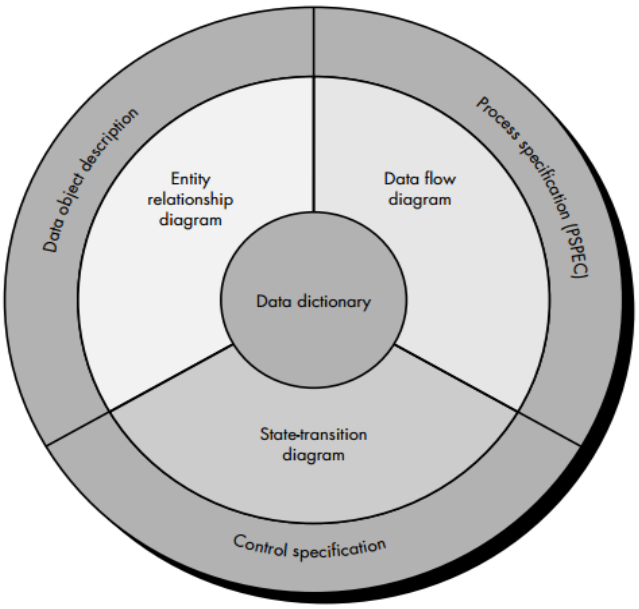
- If there are any ambiguities
- If they are complete
- If they can be demonstrated

The Elements of The Analysis Model:

The analysis model must achieve three primary objectives:

1. to describe what the customer requires
2. to establish a basis for the creation of a software design, and
3. to define a set of requirements that can be validated once the software is built.

To accomplish these objectives, the analysis model derived during structured analysis takes the form illustrated in next Figure.



The structure of the analysis model

The Elements of The Analysis Model are:

- *Data Dictionary*—a repository that contains descriptions of all data objects consumed or produced by the software.
- The *entity relation diagram* (ERD) depicts relationships between data objects. The ERD is the notation that is used to conduct the data modelling activity. The attributes of each data object noted in the ERD can be described using a data object description.
- The *data flow diagram* (DFD) serves two purposes: (1) to provide an indication of how data are transformed as they move through the system and (2) to depict the functions (and subfunctions) that transform the data flow. The DFD provides additional information that is used during the analysis of the information domain and

serves as a basis for the modelling of function. A description of each function presented in the DFD is contained in a *process specification* (PSPEC).

- The *state transition diagram* (STD) indicates how the system behaves as a consequence of external events. To accomplish this, the STD represents the various modes of behaviour (called *states*) of the system and the manner in which transitions are made from state to state. The STD serves as the basis for behavioural modelling. Additional information about the control aspects of the software is contained in the *control specification* (CSPEC).