



REQUIREMENT ENGINEERING

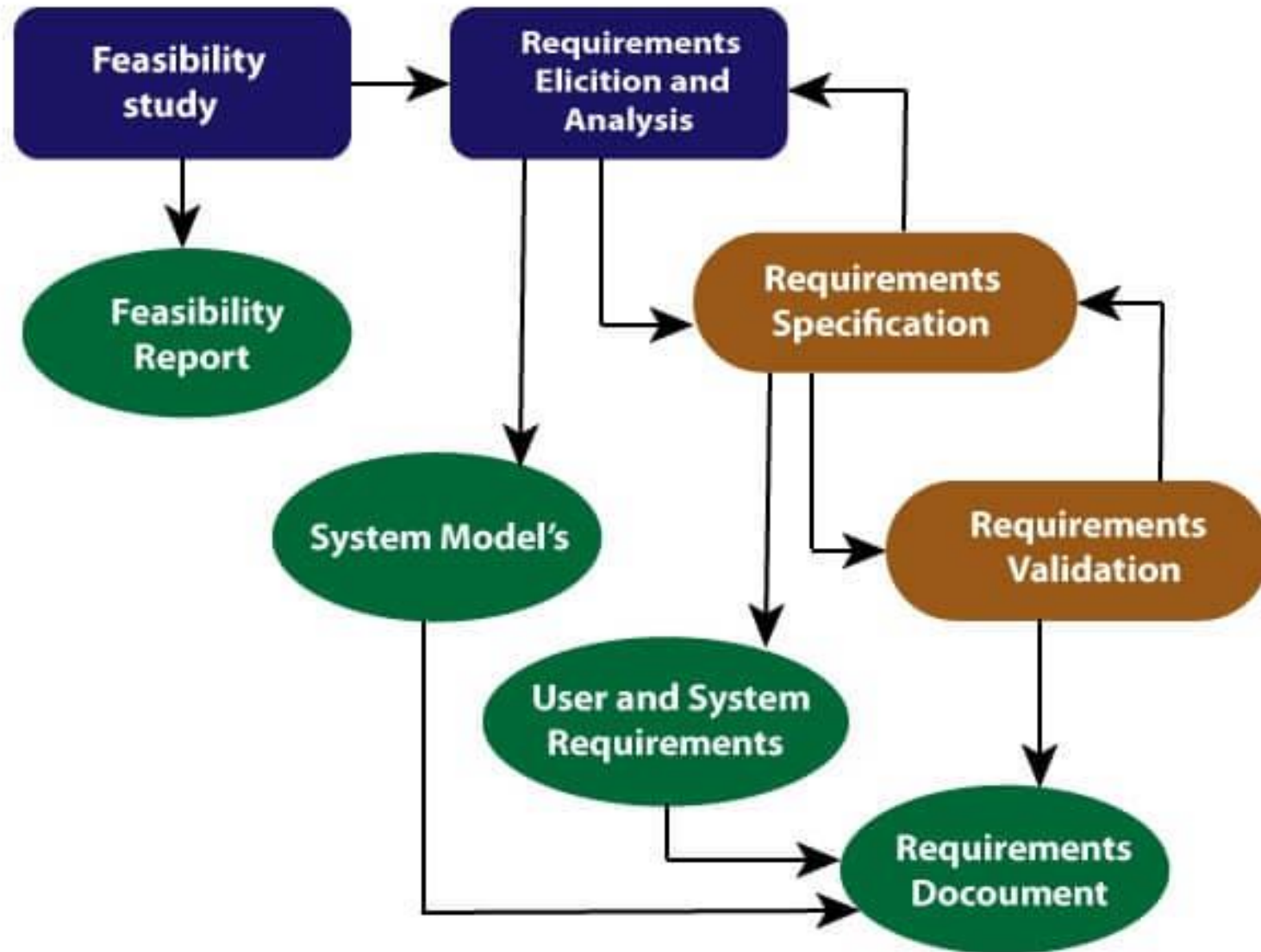
REQUIREMENT ENGINEERING

- **Requirements engineering (RE)** refers to the process of defining, documenting, and maintaining requirements in the engineering design process.
- Requirement engineering provides the appropriate mechanism to understand what the customer desires, analyzing the need, and assessing feasibility, negotiating a reasonable solution, specifying the solution clearly, validating the specifications and managing the requirements as they are transformed into a working system.
- Thus, requirement engineering is the disciplined application of proven principles, methods, tools, and notation to describe a proposed system's intended behavior and its associated constraints.

REQUIREMENT ENGINEERING PROCESS

It is a four-step process, which includes -

- Feasibility Study
- Requirement Elicitation and Analysis
- Software Requirement Specification
- Software Requirement Validation
- Software Requirement Management



Requirement Engineering Process

1. FEASIBILITY STUDY:

The objective behind the feasibility study is to create the reasons for developing the software that is acceptable to users, flexible to change and conformable to established standards.

TYPES OF FEASIBILITY:

1. **Technical Feasibility** - Technical feasibility evaluates the current technologies, which are needed to accomplish customer requirements within the time and budget.
2. **Operational Feasibility** - Operational feasibility assesses the range in which the required software performs a series of levels to solve business problems and customer requirements.
3. **Economic Feasibility** - Economic feasibility decides whether the necessary software can generate financial profits for an organization.

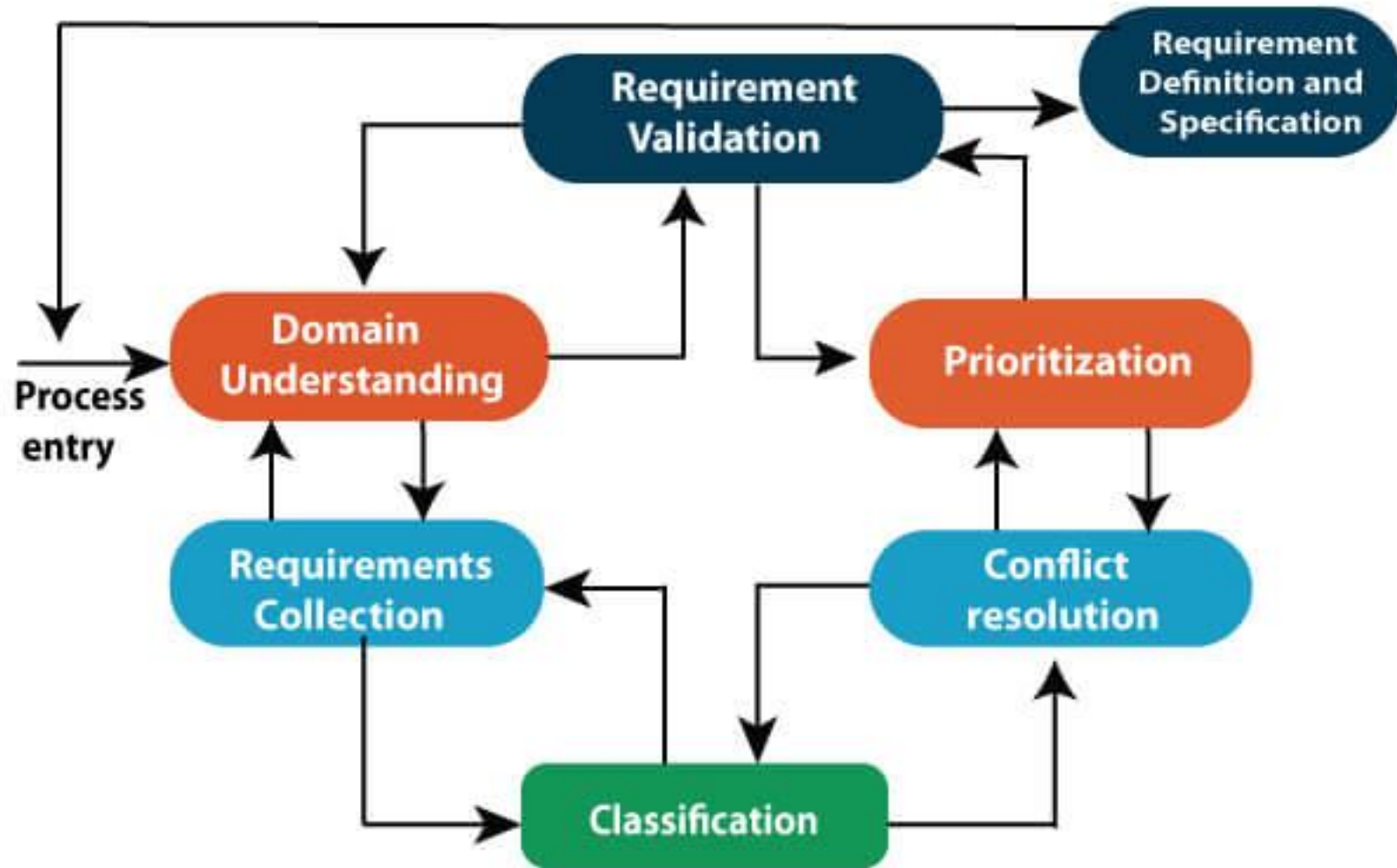
2. REQUIREMENT ELICITATION AND ANALYSIS:

- This is also known as the **gathering of requirements**.
- Here, requirements are identified with the help of customers and existing systems processes, if available.
- Analysis of requirements starts with requirement elicitation.
- The requirements are analyzed to identify inconsistencies, defects, omission, etc. We describe requirements in terms of relationships and also resolve conflicts if any.

PROBLEMS OF ELICITATION AND ANALYSIS

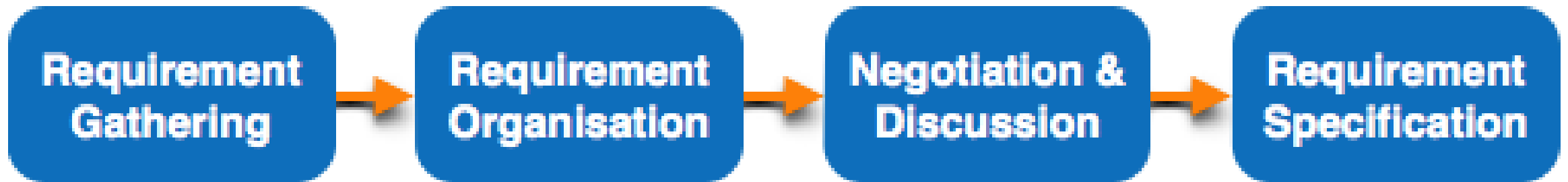
- Getting all, and only, the right people involved.
- Stakeholders often don't know what they want
- Stakeholders express requirements in their terms.
- Stakeholders may have conflicting requirements.
- Requirement change during the analysis process.
- Organizational and political factors may influence system requirements.

Elicitation and Analysis Process



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.

The requirements come from various stakeholders. To remove the ambiguity and conflicts, they are discussed for clarity and correctness. Unrealistic requirements are compromised reasonably.

Documentation - All formal & informal, functional and non-functional requirements are documented and made available for next phase processing.

REQUIREMENT ELICITATION TECHNIQUES

Interviews

Surveys

Questionnaires

Task analysis

Domain Analysis

Brainstorming

Prototyping

Observation

INTERVIEWS

Interviews are strong medium to collect requirements. Organization may conduct several types of interviews such as:

Structured (closed) interviews, where every single information to gather is decided in advance, they follow pattern and matter of discussion firmly.

Non-structured (open) interviews, where information to gather is not decided in advance, more flexible and less biased.

Oral interviews


Written interviews

One-to-one interviews which are held between two persons across the table.

Group interviews which are held between groups of participants. They help to uncover any missing requirement as numerous people are involved.


3. SOFTWARE REQUIREMENT SPECIFICATION:


1. Software requirement specification is a kind of document which is created by a software analyst after the requirements collected from the various sources - the requirement received by the customer written in ordinary language.
2. It is the job of the analyst to write the requirement in technical language so that they can be understood and beneficial by the development team.



The models used at this stage include ER diagrams, data flow diagrams (DFDs), function decomposition diagrams (FDDs), data dictionaries, etc.

- **Data Flow Diagrams:** Data Flow Diagrams (DFDs) are used widely for modeling the requirements. DFD shows the flow of data through a system. The system may be a company, an organization, a set of procedures, a computer hardware system, a software system, or any combination of the preceding. The DFD is also known as a data flow graph or bubble chart.

- 
- **Data Dictionaries:** Data Dictionaries are simply repositories to store information about all data items defined in DFDs. At the requirements stage, the data dictionary should at least define customer data items, to ensure that the customer and developers use the same definition and terminologies.



Entity-Relationship Diagrams: Another tool for requirement specification is the entity-relationship diagram, often called an "***E-R diagram***." It is a detailed logical representation of the data for the organization and uses three main constructs i.e. data entities, relationships, and their associated attributes.

4. SOFTWARE REQUIREMENT VALIDATION:

After requirement specifications developed, the requirements discussed in this document are validated. The user might demand illegal, impossible solution or experts may misinterpret the needs. Requirements can be the check against the following conditions -

- If they can practically implement
- If they are correct and as per the functionality and specially of software
- If there are any ambiguities
- If they are full
- If they can describe

REQUIREMENTS VALIDATION TECHNIQUES

Requirements reviews/inspections: systematic manual analysis of the requirements.

Prototyping: Using an executable model of the system to check requirements.

Test-case generation: Developing tests for requirements to check testability.

Automated consistency analysis: checking for the consistency of structured requirements descriptions.

SOFTWARE REQUIREMENT MANAGEMENT:

Requirement management is the process of managing changing requirements during the requirements engineering process and system development.

New requirements emerge during the process as business needs a change, and a better understanding of the system is developed.


The priority of requirements from different viewpoints changes during development process.

The business and technical environment of the system changes during the development.

PREREQUISITE OF SOFTWARE REQUIREMENTS

Collection of software requirements is the basis of the entire software development project. Hence they should be clear, correct, and well-defined.


A complete Software Requirement Specifications should be:

- 
- Clear
 - Correct
 - Consistent
 - Coherent
 - Comprehensible
 - Modifiable
 - Verifiable
 - Prioritized
 - Unambiguous
 - Traceable
 - Credible source

SOFTWARE REQUIREMENTS:

Largely software requirements must be categorized into two categories:

Functional Requirements: Functional requirements define a function that a system or system element must be qualified to perform and must be documented in different forms. The functional requirements are describing the behavior of the system as it correlates to the system's functionality.



Non-functional Requirements: This can be the necessities that specify the criteria that can be used to decide the operation instead of specific behaviors of the system.

Non-functional requirements are divided into two main categories:

- **Execution qualities** like security and usability, which are observable at run time.
- **Evolution qualities** like testability, maintainability, extensibility, and scalability that embodied in the static structure of the software system.

REQUIREMENTS VALIDATION TECHNIQUES

Requirements reviews/inspections: systematic manual analysis of the requirements.

Prototyping: Using an executable model of the system to check requirements.

Test-case generation: Developing tests for requirements to check testability.

Automated consistency analysis: checking for the consistency of structured requirements descriptions.

PREREQUISITE OF SOFTWARE REQUIREMENTS

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

SOFTWARE REQUIREMENTS

Software Requirements: Largely software requirements must be categorized into two categories:

Functional Requirements: Functional requirements define a function that a system or system element must be qualified to perform and must be documented in different forms. The functional requirements are describing the behavior of the system as it correlates to the system's functionality.

Non-functional Requirements: This can be the necessities that specify the criteria that can be used to decide the operation instead of specific behaviors of the system. Non-functional requirements are divided into two main categories:

- **Execution qualities** like security and usability, which are observable at run time.
- **Evolution qualities** like testability, maintainability, extensibility, and scalability that embodied in the static structure of the software system.

FUNCTIONAL REQUIREMENTS

Requirements, which are related to functional aspect of software fall into this category.

They define functions and functionality within and from the software system.

Examples -

Search option given to user to search from various invoices.

User should be able to mail any report to management.

Users can be divided into groups and groups can be given separate rights.

Should comply business rules and administrative functions.

Software is developed keeping downward compatibility intact.

NON-FUNCTIONAL REQUIREMENTS

Requirements, which are not related to functional aspect of software, fall into this category. They are implicit or expected characteristics of software, which users make assumption of.

Non-functional requirements include -

Security

Logging

Storage

Configuration

Performance

Cost

Interoperability

Flexibility

Disaster recovery

Accessibility

NON-FUNCTIONAL REQUIREMENTS

Requirements are categorized logically as

Must Have : Software cannot be said operational without them.

Should have : Enhancing the functionality of software.

Could have : Software can still properly function with these requirements.

Wish list : These requirements do not map to any objectives of software.

While developing software, 'Must have' must be implemented, 'Should have' is a matter of debate with stakeholders and negation, whereas 'could have' and 'wish list' can be kept for software updates.

USER INTERFACE REQUIREMENTS

easy to operate

quick in response

effectively handling operational errors

providing simple yet consistent user interface

USER INTERFACE REQUIREMENTS

Content presentation

Easy Navigation

Simple interface

Responsive

Consistent UI elements

Feedback mechanism

Default settings

Purposeful layout

Strategical use of color and texture.

Provide help information

User centric approach

Group based view settings.

SOFTWARE SYSTEM ANALYST

System analyst in an IT organization is a person, who analyzes the requirement of proposed system and ensures that requirements are conceived and documented properly & correctly.

Role of an analyst starts during Software Analysis Phase of SDLC.

It is the responsibility of analyst to make sure that the developed software meets the requirements of the client.

System Analysts have the following responsibilities:

- Analyzing and understanding requirements of intended software
- Understanding how the project will contribute in the organization objectives
- Identify sources of requirement
- Validation of requirement
- Develop and implement requirement management plan
- Documentation of business, technical, process and product requirements
- Coordination with clients to prioritize requirements and remove and ambiguity
- Finalizing acceptance criteria with client and other stakeholders

Table of Contents for a SRS Document

1. Introduction

- 1.1 Purpose
- 1.2 Document Conventions
- 1.3 Intended Audience and Reading Suggestions
- 1.4 Project Scope
- 1.5 References

2. Overall Description

- 2.1 Product Perspective
- 2.2 Product Features
- 2.3 User Classes and Characteristics
- 2.4 Operating Environment
- 2.5 Design and Implementation Constraints
- 2.6 Assumptions and Dependencies

3. System Features

- 3.1 Functional Requirements

4. External Interface Requirements

- 4.1 User Interfaces
- 4.2 Hardware Interfaces
- 4.3 Software Interfaces
- 4.4 Communications Interfaces

5. Nonfunctional Requirements

- 5.1 Performance Requirements
- 5.2 Safety Requirements
- 5.3 Security Requirements
- 5.4 Software Quality Attributes