Software Engineering

Unit 2: Requirement Engineering Principle, Design and Architectural Engineering

Introduction

- As we are already aware that life cycle of software project starts with capturing the requirements of the project.
- Gathering, documenting, disseminating and managing the requirements is the key to success of any project.
- Gathering the requirements effectively and managing them efficiently is a complex task and needs to be handled in a systematic manner.
- Principles of requirements engineering 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.

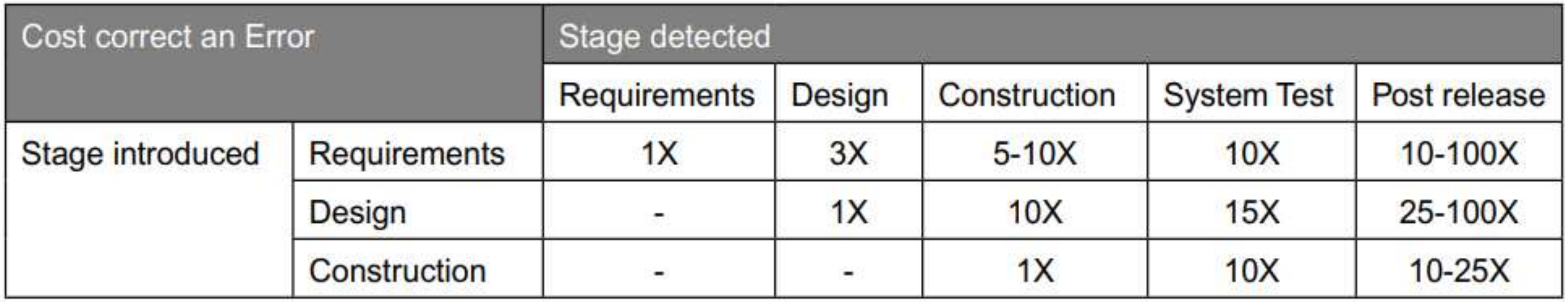
What is Requirement Engineering?

Requirement: The requirement is defined as “a condition or capability needed by a user to solve a problem or achieve an objective.

Requirements engineering: Requirements engineering is the discipline that involves establishing and documenting the requirements.

Importance of Requirement

- Requirements are the stepping stones to the success of any project.
- If software projects get started without properly understanding the user’s needs there will be lots of mismatch at the end between the final result delivered and the user’s expectations.
- A software project has to be completed within a specified time limit and budget.
- If requirements not gathered properly it results into rework, which is the primary cause of schedule and budget overruns.
- Below table shows the relative cost of repairing a defect at each stage of the software project life cycle.
- 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.



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

Types of Requirements

- Requirements can be categorized into three main types:
  1). Functional requirements, 2). Nonfunctional requirements, and 3). Interface specification.

1). Functional requirements:

- The set of requirements that defines what the system will do is called functional requirements.
- These requirements determine ***how the software will behave to meet users' needs.***
- 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.
- An example of functional requirement is a retail shop ***billing software having the functionalities of capturing products prices, calculating discounts, generating bills etc.***

2). Non-Functional requirements:

- Nonfunctional requirements stands in the support of the functional requirements.
- Some of the quality attributes of the system from an ***end user's perspective*** include ***performance, availability, usability and security.***
- Some of the quality attributes of the system from the ***developer's perspective*** include ***reusability, testability, maintainability and portability.***

Types of Requirements...

- Some of the characteristics of NFRs are:

  Performance, Availability, Maintainability

  Portability, Reliability, Security

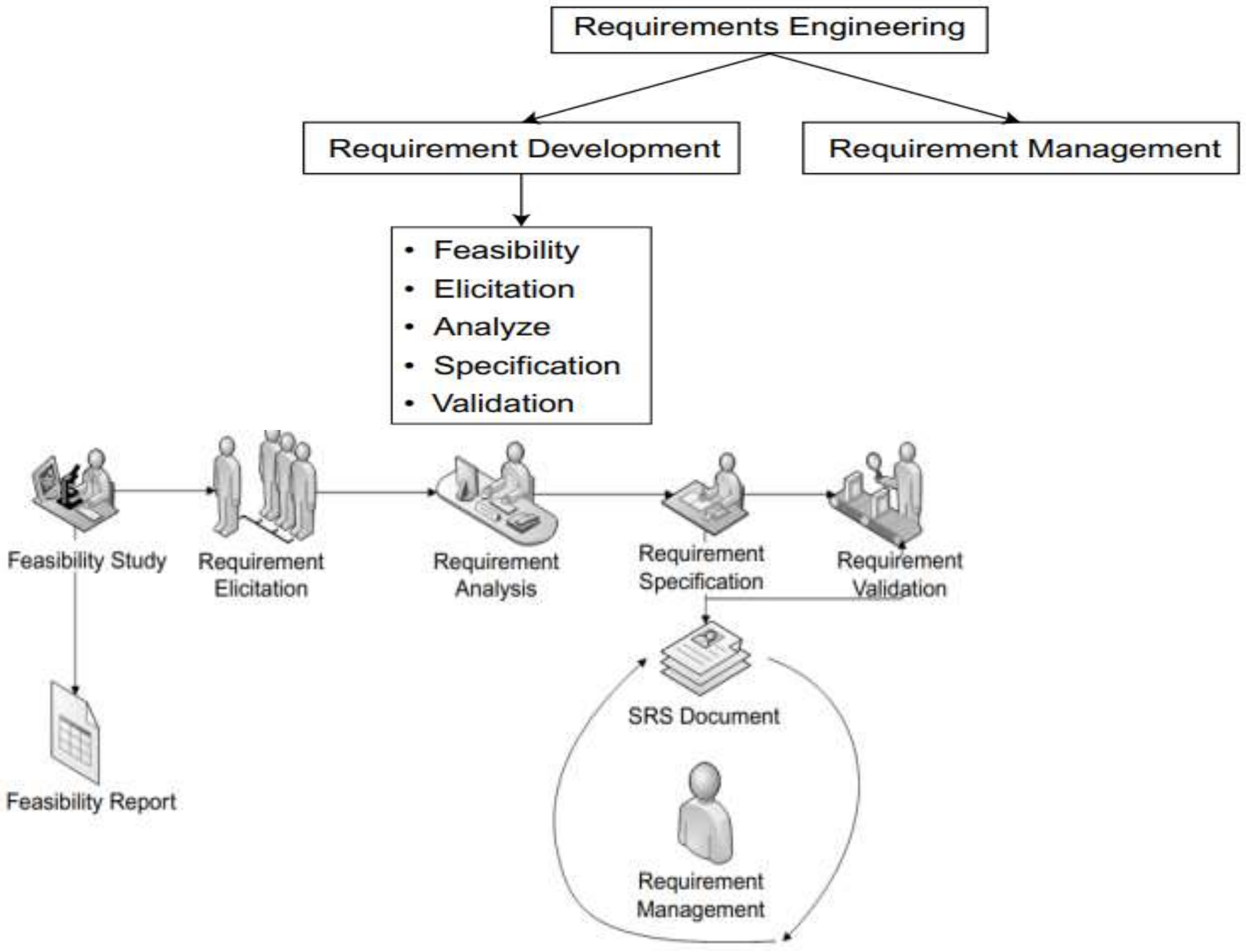
  Scalability, Testability, Usability

3). Interface Specification:

- Most of the software systems do not work alone and need to interact with other systems in order to work.
- 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.
- Let us consider 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.

Steps Involved In Requirements Engineering

- Requirements engineering includes requirements development and requirements management.



Steps Involved In Requirements Development: Feasibility Study

[A]. Requirement Development

1). Feasibility Study:

- Feasibility study is the first step of the requirements engineering process.
- A feasibility study helps in deciding whether the proposed (પ્રસ્તાવિત) system is worthwhile (યોગ્ય).
- Feasibility can be classified into three broad categories:

  i). operational feasibility, ii). technical feasibility and iii). economic feasibility.

i). Operational feasibility:

- This checks the usability (ઉપયોગીતા) 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.

ii). 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.

iii). 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.

Steps Involved In Requirements Development : Requirements Elicitation

2). Requirements Elicitation:

- In Requirement Elicitation, 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.
- 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.

i). 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 non-identification (બિન-ઓળખ) of important requirements causing (કારણભૂત) rework at a later stage.
- Thus, stakeholder identification is the first step in starting the requirements elicitation.
  - The stakeholder could be the person benefited/affected.
  - The stakeholder could be the person who is responsible for what we are making.
  - The stakeholder could be the person who may go against or in favour of what we are making.
  - The stakeholder could be the person who can make better what we are making.
  - The stakeholder could be the person who can contribute to the financial or technical resources.

Steps Involved In Requirements Development : Requirements Elicitation..

<u>ii). Problems in Eliciting Requirements</u>

a). 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 previous software system that can be analyzed to get the requirement.

b). Communication gap: Even if the end users and stakeholders are clear about the need for developing the new system, they may state the requirements in an unclear or non testable fashion.

Sometimes the important basic requirements may be omitted and they become engage in explaining the non-important requirements.

c). Conflicting requirements: This problem increases when the number of stakeholders is more.

There may be conflicting needs and priorities for each stakeholder based on the business areas they are covering.

d). Volatile requirements: Requirements may get changed due to the entry of new stakeholders who have different thinking of the system. It sometimes creates problem between the requirements engineer and the stakeholders due to continuous change in the scope.

Steps Involved In Requirements Development : Requirements Elicitation…

<u>**ii). Requirements Elicitation Techniques:**</u>

- 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.
- Available requirements elicitation techniques are:
- Interviewing • Focus groups • Facilitated workshops • Prototyping • Questionnaires • Brainstorming • Direct observation • Apprenticing

Interviewing:

- Interviewing is an important and effective way to gather and validate the requirements.
- Begin the interviews with the stakeholders who are believed to have complete understanding of the requirements.
- During the execution of the project, at any point in time, interviews may happen 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.
- There are several types of focus groups that facilitate group discussions, which are as below:

<u>***Two-way focus group***</u>: 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.

<u>***Dual moderator focus group Model 1***</u>: 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.

Steps Involved In Requirements Development : Requirements Elicitation…

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 .

Client participant focus group: Client representatives participate in the discussion and will ensure ownership from the client on the decision taken. However, client participant focus groups help overcome the problem of project rejection at the acceptance test.

Mini focus group: Groups are restricted to four or five members rather than 6 to 10. This 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 can be used for collecting requirements quickly. A workshop is expensive because it involves many people, but it saves a large amount of time. Workshops are useful in situations where the requirements are focused on one area of business. In workshops, different alternatives are given to the customers so that they choose the appropriate options.

Steps Involved In Requirements Development : Requirements Elicitation…

Prototyping: 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***. It is actually the initial version of the system.

Questionnaires: Questionnaires can be used to collect input from ***multiple stakeholders quickly.*** 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 number of questions (***too many or less)***

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 a 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***.

Direct Observation: In many cases a new system is developed to replace an existing system. it is very useful to gather knowledge of the requirements by observing what is actually done by the user of the system.

Apprenticing: In apprenticing model the requirements engineer performs the tasks that the user of the system carries out. By which he tries to understand the task and find the requirements. The list of requirements will be increased as the need is raised.

Steps Involved In Requirements Development: Requirements Analysis

3). Requirement Analysis

- Requirements analysis is the ***third step*** of the requirements engineering process.
- The aim of the ***elicitation phase is to gather the requirements***, whereas in the analysis phase ***the requirements are understood in detail***.
- This is the phase that bridges the gap between ***requirements engineering*** and ***design.***
- Thus, the requirements analyst ***refines the requirements*** collected from the various stake holders.
- It acts as the ***input to the design***.
- In this phase, an ***analyst will start working on the solution of the requirements***.
- A system architecture, and database requirements, etc. are discussed with the customer to ensure that the system design is based on the customer's need.
- Various modeling techniques are used during the analysis phase, such as use-case diagram, E-R diagram, DFD, data dictionary, class diagram, etc.

Steps Involved In Requirements Development: Specifying Requirements

4). Specifying Requirements

- In this phase, all the knowledge captured in the form of requirements are gathered during these steps in a clear (ચોખ્ખુ), concise (સંક્ષિપ્ત) and unambiguous (સ્પષ્ટ) manner so that the design and development teams can use it going forward.
- While specifying the requirements, it is also important that it is ***validated with the customers*** and users to confirm if all the requirements are captured at this stage.
- ***Software Requirement Specification (SRS) is the outcome*** of this requirements specification and validation stage.

Specifying Requirements in the SRS Document

- SRS document becomes the main source for any requirements clarification and dissemination to all stakeholders.
- It may form the basis for the ***contractual agreement*** between the customer and the suppliers.
- Thus ***any change*** in this agreed understanding will be renegotiated (ફરીથી વાટાઘાટો) between the customer and the suppliers.
- This is the foundation document for the ***test engineers*** to develop their strategy on how to test the proposed system fully.
- The ***design*** and ***development*** team start their work based on this document.
- The ***resourcing, budgeting, scheduling and pricing*** for the project are based on the volume and complexity of the requirements captured in this document.
- A good SRS ensures ***future maintainability*** of the project.

Steps Involved In Requirements Development: Specifying Requirements..

Checklist for Writing a Good SRS Document

i). Correctness:

a) Will the requirements meet the customer's need?

b) Do the requirements capture how the system should transform, produce and provide the exact outcome expected from the system?

c) The correctness of requirements typically refers to the accuracy of the requirements and whether everything agreed is delivered.

ii). Completeness:

a) Are all the requirements captured which will form a system that fully provides a solution to the customer's problem?

b) Are each requirement detailed enough and supported by necessary diagrams, figures, data and use cases so that all the stakeholders get their necessary input from the requirement?

c) Are all functional, nonfunctional and interface requirements captured for the system?

iii). Consistency:

a) Are there requirements that conflict with each other? This may happen when multiple stakeholders have different views of the proposed system and provide requirements that create conflicts while grouped together.

b) Are the terminology, diagrams, tables and figures consistent and provide a consistent view of the proposed system?

Steps Involved In Requirements Development : Specifying Requirements..

iv). Verifiability

a) Are the requirements verifiable – that is, will the test engineers be able to check whether the requirement fulfills the customer's need in entirety before the product is shipped to the customer.

b) Are the requirements validated and verified by all the stakeholders before they are baseline for consumption of downstream processes?

v). Clarity

a) Is the requirements statement clear enough and can be interpreted only in one way?

b) Will the design and construction engineer be able to determine the single way of implementation from the requirements document?

vi). Priority

a) Prioritization of the requirements is important as it drives the plan for implementing these.

b)Priority of the requirements becomes important as it will provide a clear guideline of which requirements should be tackled first, and which requirements are less important.

vii). Modifiability

a) A requirement that has dependency on several other requirements should be structured in a fashion so that the impact of any modification to this requirement across other requirements can be easily traceable.

Steps Involved In Requirements Development: Validating Requirements

- Validating requirements is an important step, as ***invalid requirements may cost more*** to rectify (સુધારવું) at later stages of the project life cycle.
- The main aim of requirements validation is to ensure that the customer needs are captured ***completely, clearly and consistently.***
- The validation step should provide ***enough confidence to all the stakeholders*** that the proposed system will provide all ***the features required*** and will be ***completed within the time*** and ***budget*** allocated for the project.
- Different stake holders who should be involved in the validation process are:

  1.Customer 2. End user 3. Domain expert 4. Architect 5. Construction and test engineer

  6. Third-party systems interacting with the proposed system
- There are different approaches taken for validating the requirements, which are discussed below.

<u>i). Walkthrough:</u>

This is an informal process where the author of the document presents the document to a group of people, and the observations or issues found by the group are reported and corrected after the walkthrough session.

Generally, ***minutes of the meeting are not maintained*** for the walkthrough sessions.

Steps Involved In Requirements Development: Validating Requirements..

ii). Review:

This may be either a formal or an informal process where the group of stakeholders will critically go through the SRS document and check the following:

a) The standards and guidelines available in the organization are followed in the document.

b) The completeness, consistency and other points mentioned in the Checklist for Writing a Good SRS Document are followed.

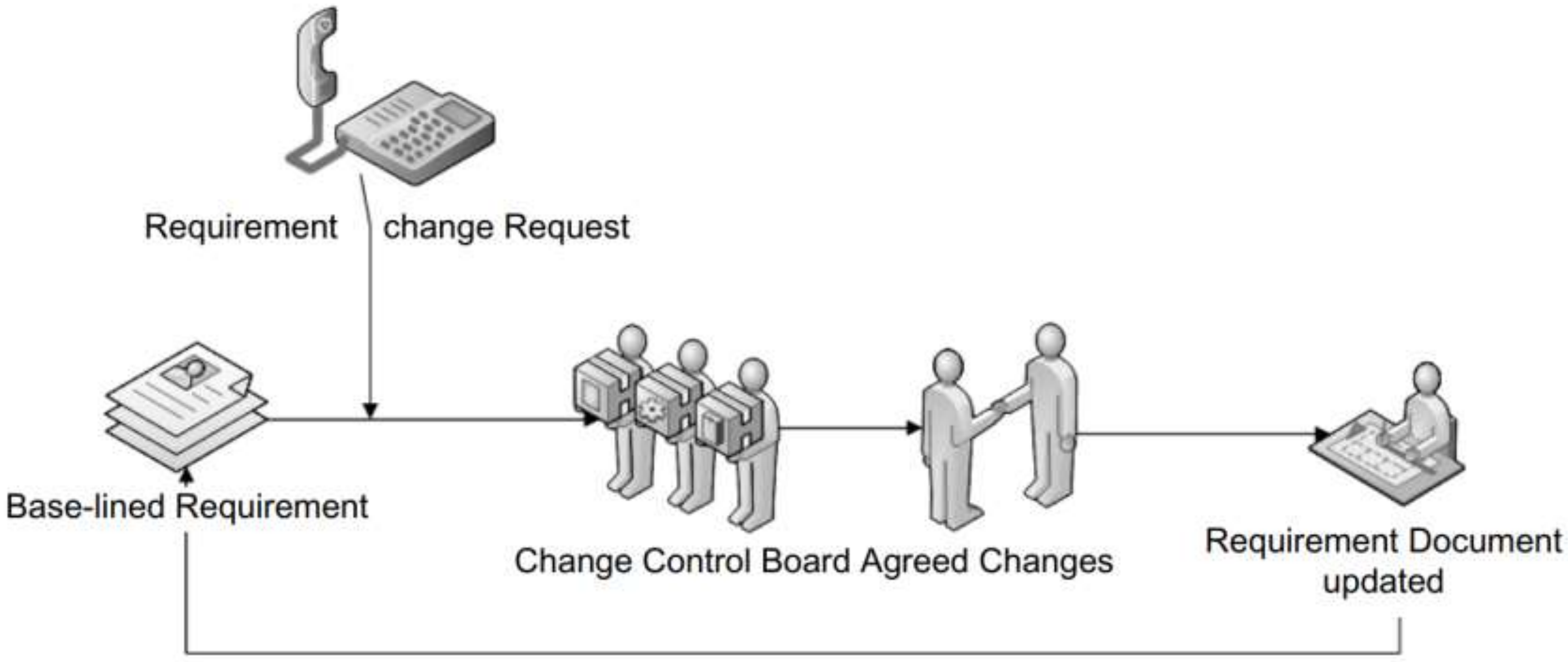
c) All the requirements could be traced back to their origin. This will ensure that only the valid requirements are captured and implemented and not just any wish list from the user groups.

In this process, the ***minutes of the review sessions are captured*** and all the findings are tracked until satisfactory closure.

iii). Inspection:

This is a more structured way of review where the non conformances found in the SRS document based on the points discussed above are listed as defects and tracked until the defects are closed satisfactorily.

Steps Involved In Requirements Management



Steps Involved In Requirements Management..

- The requirements management deals with how the ***existing requirements*** can be stored and tracked and how the ***changes to these requirements*** and also the ***introduction of new requirements*** can be handled during the requirements phase as well as throughout the life cycle of the project.

Requirements Management Plan

- During requirements engineering, the plan for managing the requirements should mention how to manage the following:

1). Requirements identification

2). Requirements change

3). Requirements status tracking

1). Requirements Identification: It is the first step in requirements management. This creates the baseline set of requirements on which the implementation team starts working.

2). Requirements Change: Requirement can change if the customer feels so. The requirement may also change because of changing requirement of the business and the market. The priority may also change.

Steps Involved In Requirements Management..

Change Control Process:

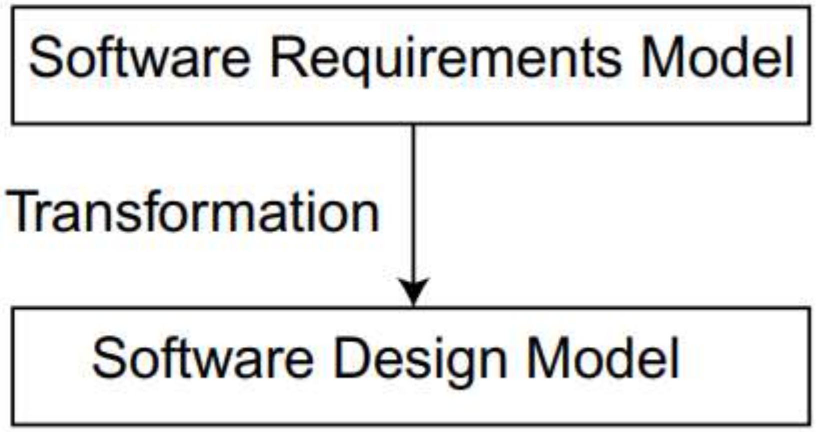
- Once the initial requirements are agreed upon with different stakeholders, it is important that any change in these requirements are controlled well.
- Changes in requirements are almost always related to change in ***project effort, timeline and cost.***
- By introducing a Change Control Board, all the stakeholders try to ensure that everyone gets ***good transparency*** on the changes agreed upon.
- A change control board is a group of people comprising ***members from each stakeholder group*** who are directly linked with the project outcome.
- This group sets forth the steps to be followed whenever a change or a new requirement crops up after the initial set of requirements are signed off by everyone.

3). Requirements Status Tracking After the requirements gathering phase, the set of requirements get baseline, which can be modified after the change control board approves a requirement for implementation.

It is important to ***track the status of all these requirements*** throughout the life cycle of the project, that is, until the ***requirements are available to the customer.***

Design Process & Concepts

- During the design process, the ***software requirements model***, which was prepared for the requirements is ***converted into suitable and appropriate design models*** that describe the architecture and other design components.
- Each design product is ***verified and validated*** before moving to the next phase of software development.
- The design process encompasses a sequence of activities that slowly reduces the level of abstraction. ***It means the requirements are made clear in detail.***



Basic issues in Software Design

- If the requirements model is wrongly represented then the design model also will go wrong.
- Mapping each functional requirement into design representation is difficult.
- Mapping all the requirements into a single design model is a tedious job.
- Modeling a design that enables easy coding as well as catering to all the requirements is difficult.
- Requirements keep changing and evolve over a period of time, but changing the design accordingly is difficult.
- Balancing simplicity and efficiency is difficult.
- Balancing clarity and code safety is difficult.
- Balancing the robust design and the system response time is difficult.
- Following standards in design and simultaneously catering to and matching the requirements is difficult.

Characteristics of a Good Design

- For good quality software to be produced, the software design must also be of good quality.
- Now, the matter of concern is how the quality of good software design is measured?
- This is done by observing certain factors in software design. These factors are:
    1. Correctness
    2. Understandability
    3. Efficiency
    4. Maintainability

<u>1). Correctness (ચોકસાઈ)</u>

- First of all, the design of any software is evaluated for its correctness.
- The evaluators check the software for every kind of input and action and observe the results that the software will produce according to the proposed design.
- If the results are correct for every input, the design is accepted and is considered that the software produced according to this design will function correctly.

<u>2). Understandability (સમજણ)</u>

- The software design should be understandable so that the developers do not find any difficulty to understand it.
- Good software design should be self- explanatory.
- This is because there are hundreds and thousands of developers that develop different modules of the software, and it would be very time consuming to explain each design to each developer.
- So, if the design is easy and self- explanatory, it would be easy for the developers to implement it and build the same software that is represented in the design.

Characteristics of a Good Design…

3). Efficiency (કાર્યક્ષમતા)

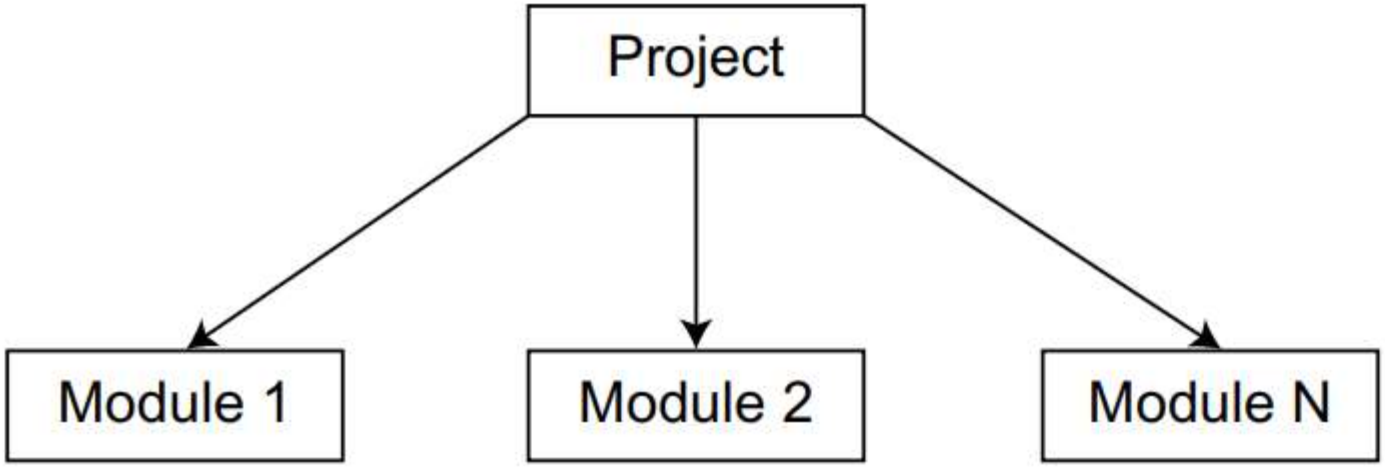
- The software design must be efficient.
- The efficiency of the software can be estimated from the design phase itself, because if the design is describing software that is not efficient and useful, then the developed software would also stand on the same level of efficiency.
- Hence, for efficient and good quality software to be developed, care must be taken in the designing phase itself.

4). Maintainability (જાળવણીક્ષમતા)

- The software design must be in such a way that modifications can be easily made in it.
- This is because every software needs time to time modifications and maintenance.
- So, the design of the software must also be able to bear such changes.
- It should not be the case that after making some modifications the other features of the software start misbehaving.
- Any change made in the software design must not affect the other available features, and if the features are getting affected, then they must be handled properly.

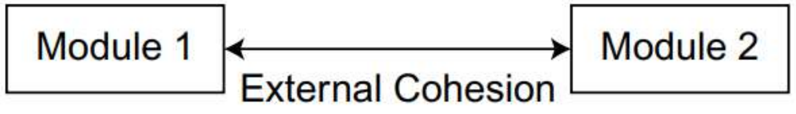
Modularity

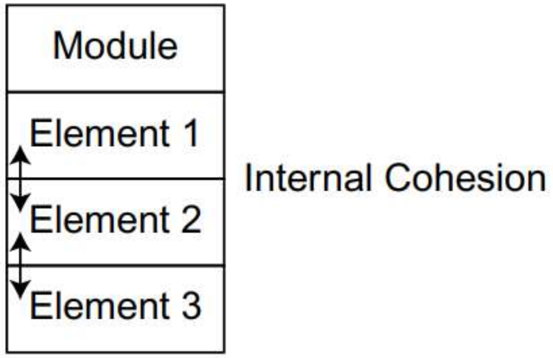
- The possibility of ***dividing a single project into smaller units called modules*** is termed modularity of the project.
- Modularity helps in designing the system in a ***better way and ensures that each module communicates and does the specific task*** assigned to it.
- It also helps in ***reusability and maintainability***.
- The two main characteristics that need to be considered while designing the modularity of projects are (i). cohesion and (ii). coupling.



(i). Cohesion

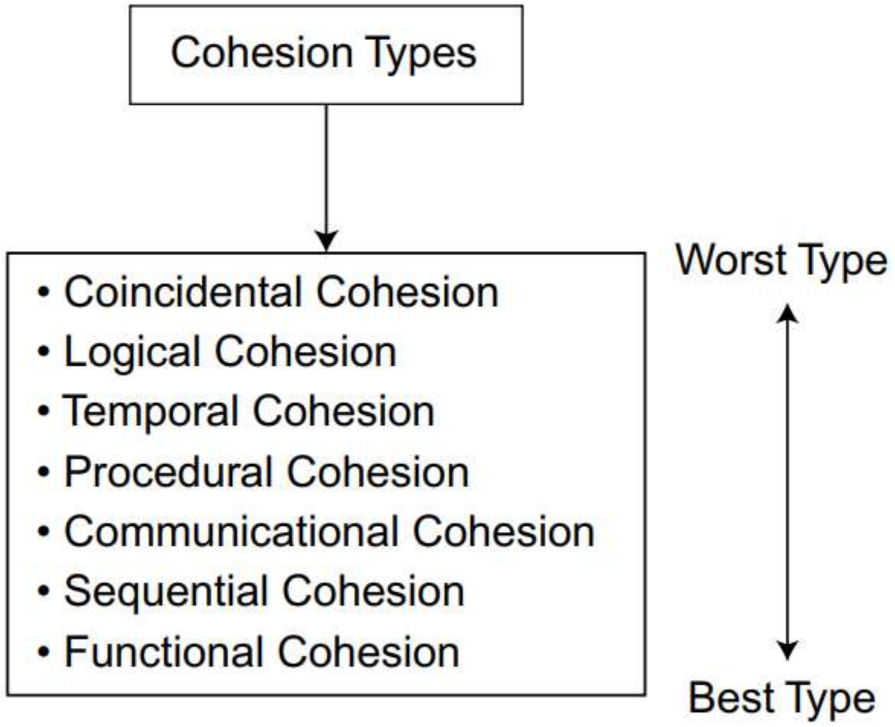
- Cohesion refers to ***"how strongly" one module is related to the other***, which helps to group similar items together.
- Cohesion measures the semantic strength ***(high and low cohesion***) of relationships between modules within a functional unit.
- ***Internal cohesion of a module is the strength of the elements within the modules***, whereas ***external cohesion of a module is the strength of relationships between modules.***





(i). Cohesion...

- There are several types of cohesion, which are discussed as below:
  i). Coincidental cohesion ii). Logical cohesion iii). Temporal cohesion
  iv). Procedural cohesion v). Communicational cohesion vi). Sequential cohesion
  vii). Functional cohesion



(i). Cohesion...

i). Coincidental cohesion:

- As the name indicates, in this type the ***cohesion (grouping) occurs coincidently.***
- Elements or modules are grouped together for different purposes and there is no common reason for grouping.
- The only common thing is that they are grouped together.

ii). Logical cohesion

- In this type, as the name indicates, ***cohesion (grouping) occurs logically based on similarity and not based on functionality***.
- Elements or modules are grouped together based on common logic.
- Logical grouping may be based on the functionality, nature, and behavior of the elements/modules.

iii). Temporal cohesion

- In this type, ***cohesion (grouping) occurs during run time at a particular time of program execution***.
- Modules that are being called during the exception handling procedure can be grouped together.
- Modules that are being executed at time T, (M1, M9, M16 etc.) can be grouped together.

(i). Cohesion...

<u>iv). Procedural cohesion</u>

- In this type, cohesion (grouping) is used to execute a certain procedure.
- ***Modules that are used for a procedure can be grouped together.***
- Note that the same module can be ***grouped together with another module for executing another procedure.***

<u>v). Communicational cohesion</u>

- In this type, ***cohesion (grouping) occurs because part of the module shares same data*** (acts on same data) for communication purposes.
- For example, modules M1, M2, and M3 access common data (same database).

<u>vi). Sequential cohesion</u>

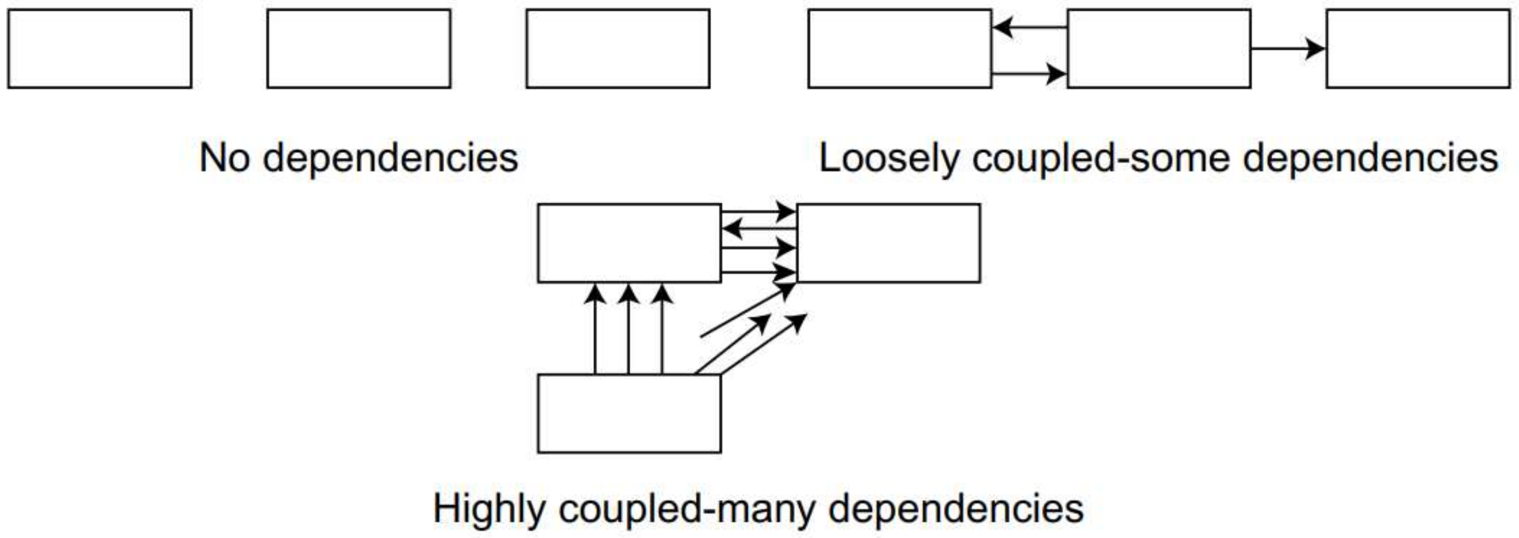
- In this type, ***cohesion (grouping) occurs as the output of one module is an input of another module***. This will look like an ***assembly line sequence***.

<u>vii). Functional cohesion</u>

- In this type, ***cohesion (grouping) occurs as all the functions contribute to a single task of the module***.
- ***In order to execute a single task, all modules contribute***.

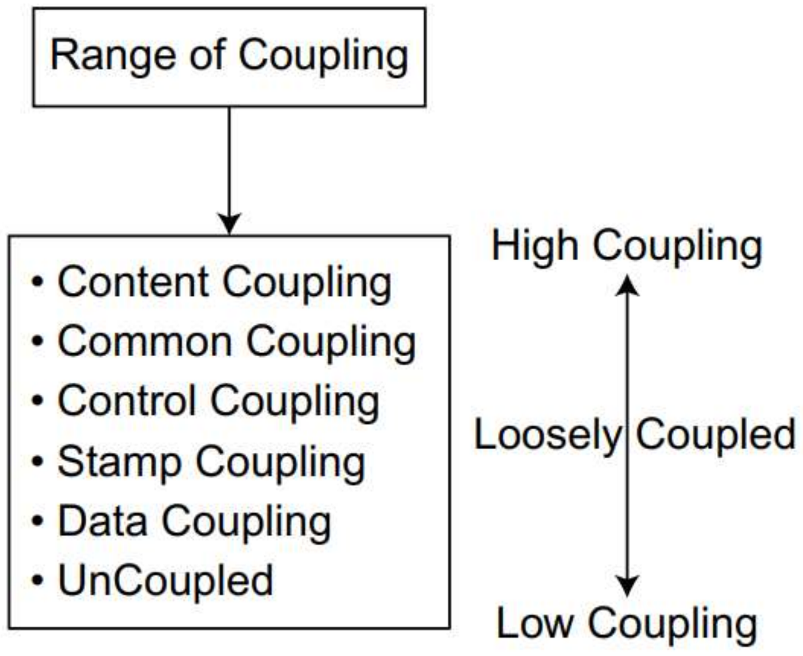
(ii). Coupling

- Coupling is a ***measure of "how tightly" two entities or modules are related to each other***.
- Coupling ***measures the strength of the relationships between entities or modules***.
- ***Minimize coupling and maximize cohesion is the principle of design***.
- This means that a module should be ***as independent as possible***.
- It is very easy to measure coupling both quantitatively and qualitatively.


No dependencies
Loosely coupled-some dependencies
Highly coupled-many dependencies

(ii). Coupling...

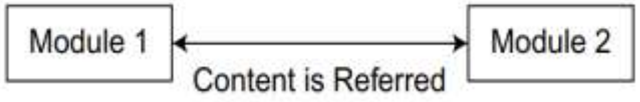
- There are different types of coupling as listed below:
  (i). Content coupling (ii)Common coupling (iii). Control coupling
  (iv). Stamp coupling (v). Data coupling (vi). Uncoupled



(ii). Coupling...

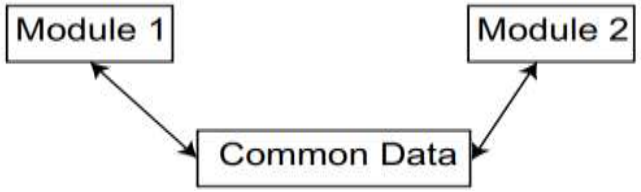
(i). Content Coupling:

- In content coupling, one component refers the content of another component and here the coupling is considered to be higher.
- For example, Module 1 accesses the data of Module 2 and changes it.



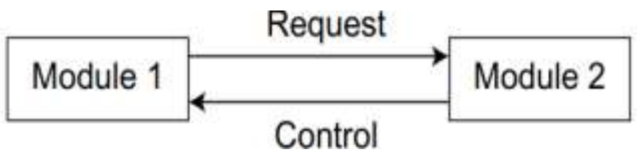
(ii). Common coupling:

- In common coupling, one or more modules share the same data which is common.
- This is an example of bad design and should be avoided.



(iii). Control coupling:

- In control coupling, a module passes its control component to another module.
- Depending on the control being passed, it is considered as good or bad.



(ii). Coupling...

(iv). Stamp coupling:

- In stamp coupling, a module passes a data structure to a module, which does not have access to the entire data structure.
- The module sends the data structure to another module to understand it.
- The module does not have any power and just acts as a stamp and is therefore called stamp coupling

Module 1
.Data Structure
Module 2
.Data Structure

(v). Data coupling:

- Two modules are data coupled, if there are homogeneous data items or structures.

(vi). Uncoupled:

- It means there is no coupling between the modules.