Software Engineering and Object-Oriented Analysis & Design

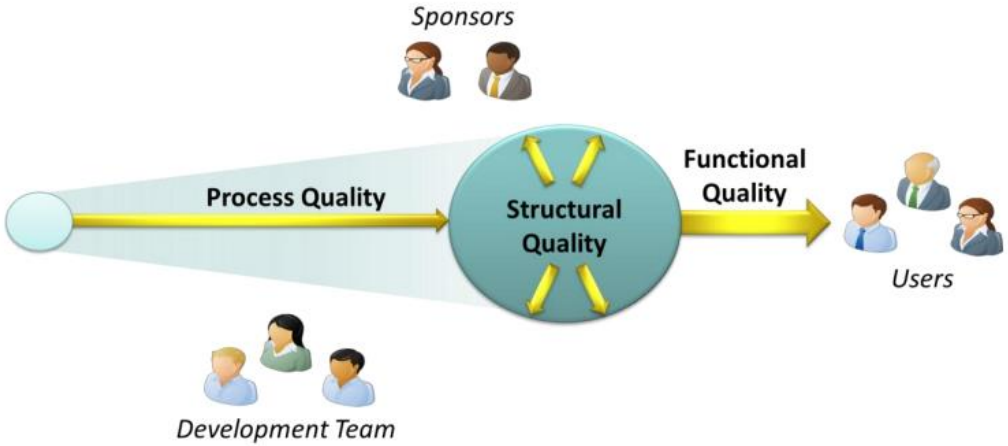
Software

- Software is a collection of computer programs, procedures and documentation that performs different tasks on a computer system.
- It plays the role of mediator between the user and computer hardware. Without software, a user can’t perform any task on a digital computer.

Characteristics of software

1. **Software does not wear out:**
   Different things like clothes, shoes, ornaments do wear out after some time. But, software once created never wears out.
2. **Software is not manufactured**:
   Software is not manufactured but is developed. So, it does not require any raw material for its development.
3. **Reusability**:
   As the software never wears out, we can reuse the existing code form the software in which it is already present. This reduced our work and also saves time and money.
4. **Flexibility**:
   We can make necessary changes in our software in the future according to the need of that time
5. **Maintainability**:
   Every software is maintainable. This means that if any errors or bugs appear in the software, then they can be fixed.
6. **Portability**:
   We can transfer our software from one platform to another that too with ease.
7. **Reliability**:
   Our software should work properly in each condition and provide the desired functionalities under every condition.

Software quality attributes



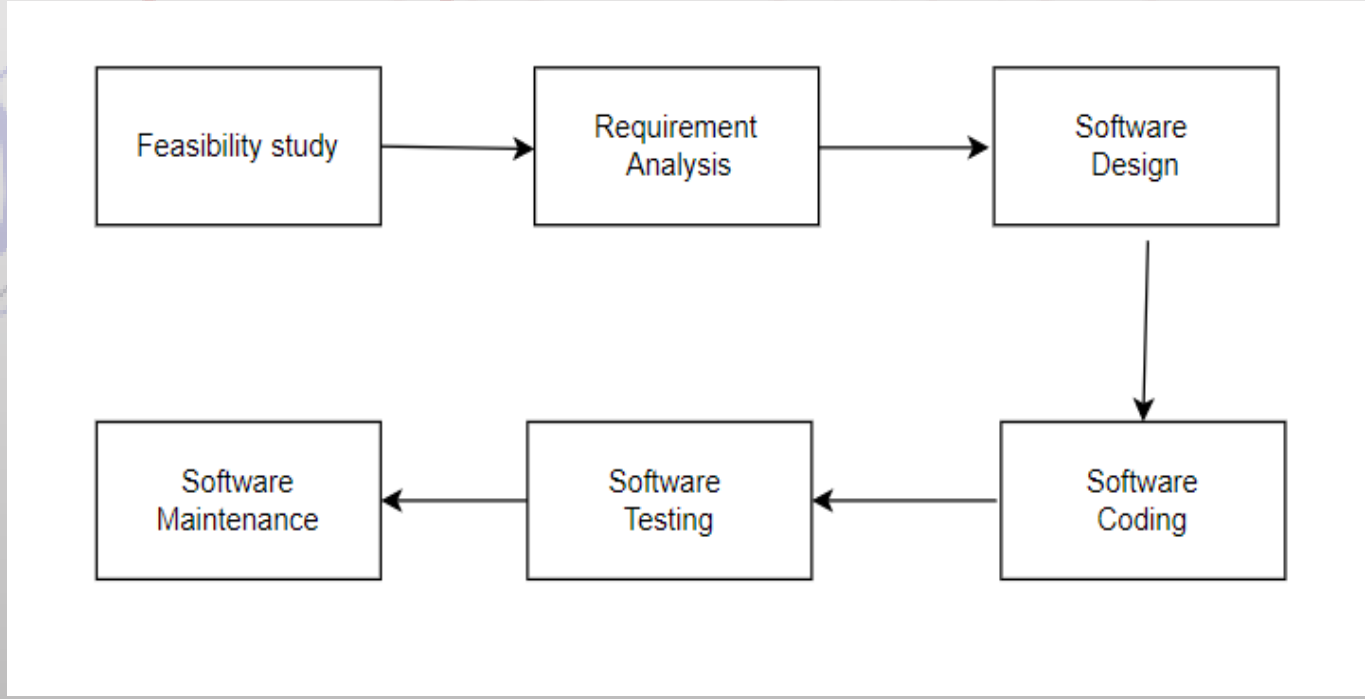
Functional quality means that the software correctly performs the tasks it's intended to do for its users.
Attributes of functional quality are :meeting the specified requirements, creating software that has few defects, good enough performance, ease of learning and ease of use.

Structural quality means that the code itself is well structured. Unlike functional quality, it is hard to test.
Attributes of structural quality are: code testability, code maintainability, code understandability, code efficiency, code security.

Process quality: The quality of the development process significantly affects the value received by users, development teams, and sponsors.
Attributes of process quality are: Meeting delivery dates, Meeting budgets, A repeatable development process that reliably delivers quality software.

Software Process Models/Software Development Lifecycle

- Software process model is an abstract representation of software process.
- Each process model represents a process from a particular perspective and thus provides only partial information about that process.



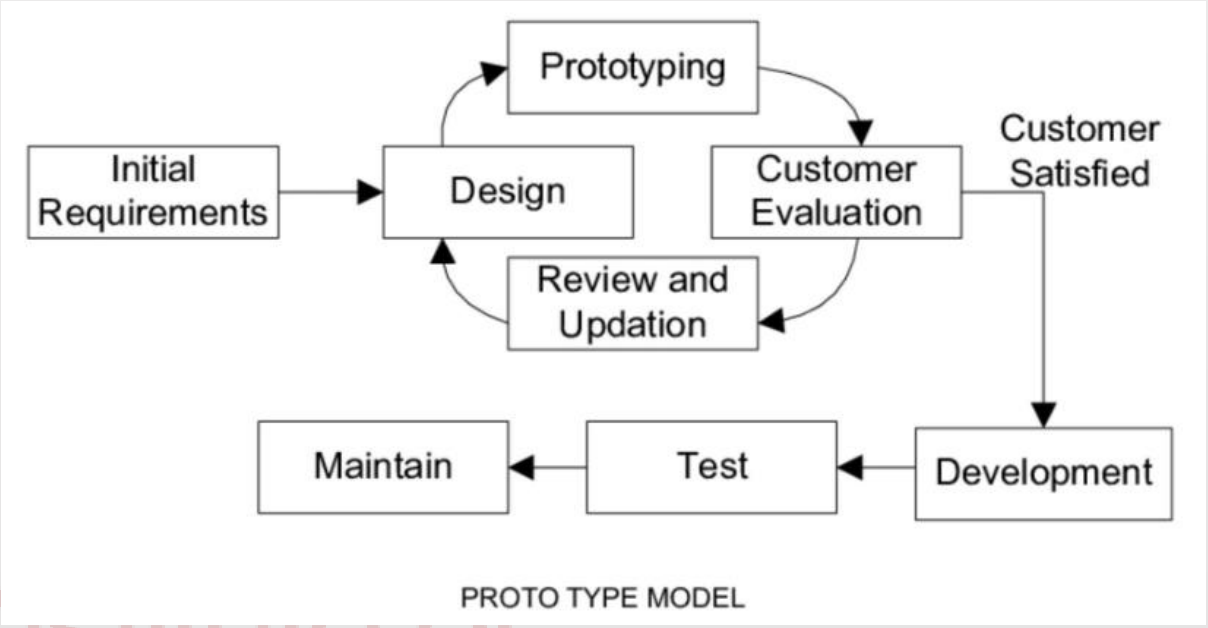
Prototype model

Advantages of Prototyping Model

- Users are actively involved in the development.
- Errors can be detected much earlier.
- Reduce Maintenance cost.
- Confusing or difficult functions can be identified
- Quicker user feedback is available leading to better solutions.
- It identifies the missing functionality easily. It also identifies the confusing or difficult functions.

Disadvantages of Prototyping Model

- The client involvement is more and it is not always considered by the developer.
- It is a slow process because it takes more time for development.
- Difficult to know how long the project will last.
- Many changes can disturb the rhythm of the development team.
- Poor documentation because the requirements of the customers are changing.

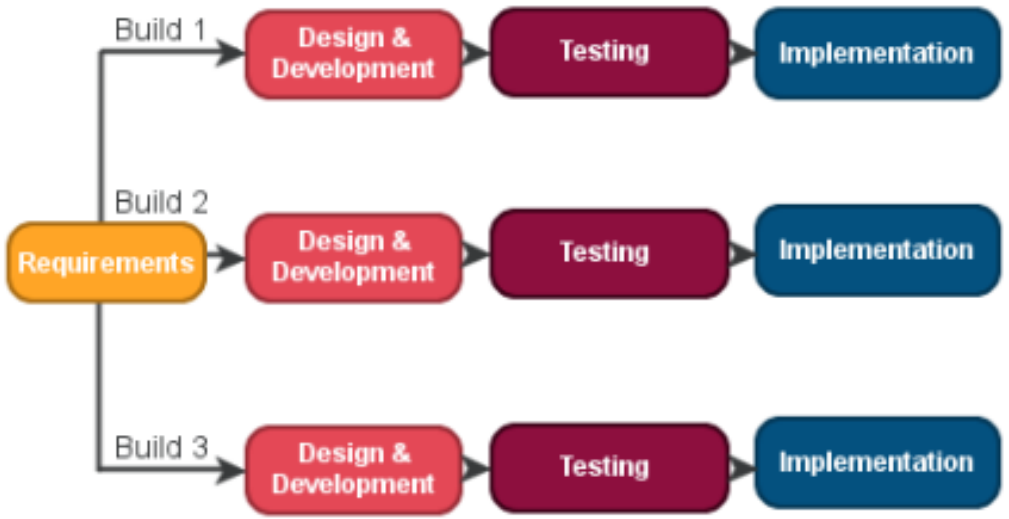


PROTO TYPE MODEL

When to use

- If user requirements and technical aspects are not understood.
- For development of User interface part of project.

Iterative Model



Advantages

- Parallel development can be planned.
- Testing and debugging during smaller iteration is easy.
- less time is spent on documenting and more time is given for designing.
- easily adaptable to the ever changing needs of the project as well as the client.

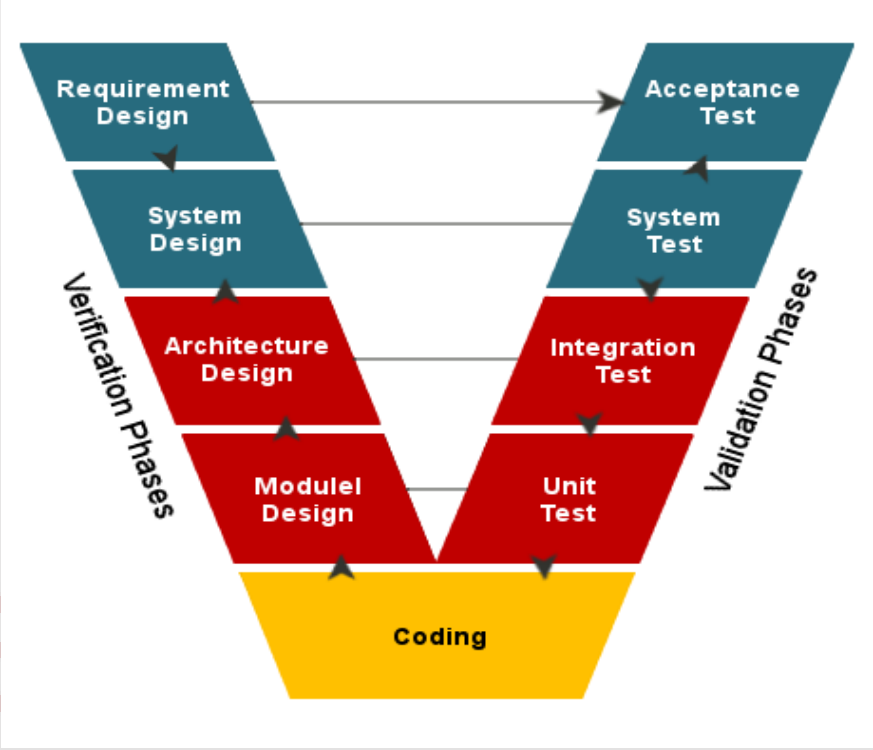
Disadvantages

- More resources may be required.
- More management attention is required.
- Not suitable for smaller projects.
- Project progress is highly dependent upon the risk analysis phase.
- Highly skilled resources are required for skill analysis.

When to use

- requirements of the complete system are clearly defined and understood.
- resources with needed skill sets are not available and are planned to be used on contract basis for specific iterations
- f there are some high risk features and goals, which might change in the future.

V- Model



Advantages

- Helps deliver quality product by testing product during every stage of development.
- Testing starts earlier which helps in avoiding defects in later stages of development.

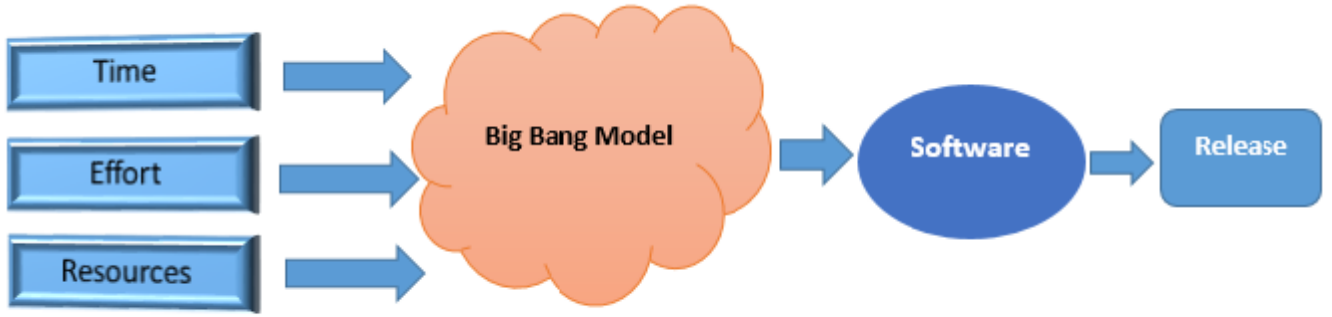
Disadvantages

- Needs to be updated in both test and requirement document when changes occur.
- Not suitable for projects when requirements are at moderate to high risk of changing.
- Very rigid and less flexible

When to Use

- When there are no undefined requirements.
- when sample technical resources are available with essential technical expertise.

Big Bang Model



Big Bang SDLC MODEL

Advantages of Big Bang Integration:

- No need of formal planning.
- Least planning required.
- Very limited resources are required.
- Developers have full flexibility in applying their ideas or opinions.

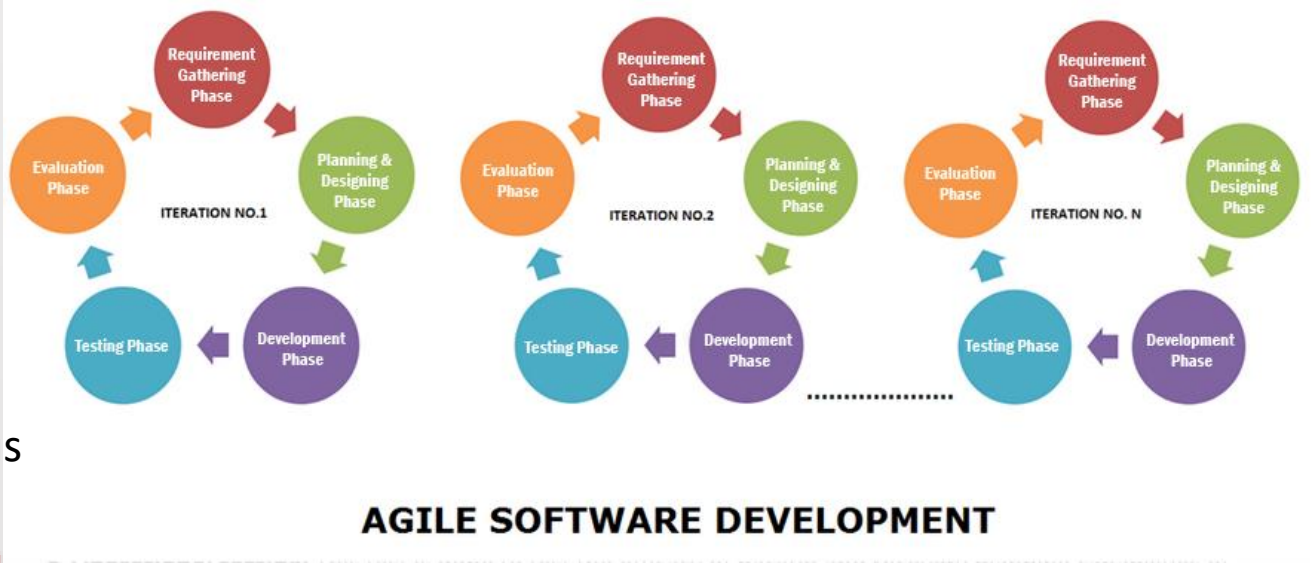
Disadvantages of Big Bang Integration:

- Not suitable for object oriented objects that are generally complex in nature.
- If requirements are not understood well, then it may prove to be quite expensive.
- Not suitable for projects that are lengthy in nature.
- Higher level of risk and complexity as no formal method is followed.

Introduction

- No certain technique used
- Started with just a basic amount of money and resources.
- Preferable when customer is not sure about his wants and the requirements are not analyzed that well

Agile Model



AGILE SOFTWARE DEVELOPMENT

Advantages of Agile model

- Customers are satisfied because of quick and continuous delivery of useful software.
- Regular delivery of working software.
- Face to face interaction between the customers, developers and testers

Disadvantages of Agile model

- Totally depends on customer interaction. If the customer is not clear with their requirements, the development team can go in the wrong direction.
- Documentation is less, so the transfer of technology to the new team members is challenging.

Introduction

- combination of incremental and iterative process models.
- focuses on the users satisfaction with quick delivery of working software product.
- breaks the product into individual iterations
- At the end of an iteration working product shows to the users.
- With every increment, features are incremented
- Shorter iteration varying from 2 weeks to 2 months.

Computer Aided Software Engineering(CASE)

- CASE is the use of software tools to help in the development and maintenance of software.
- Are intended to provide Automated Support for software process activities.
- Software developers always looking for such CASE tools that help them in many different ways during the different development stages of software.
- They can understand the software and prepare a good end product that efficiently fulfill the user requirements.
- CASE tools provide the ways that can fulfill the requirements of software developers.
- These tools provide computerized setting to software developers to analyze a problem and then design its system model.

CASE Tools

Major categories of CASE tools are:

- **Diagram tools** : Flow Chart Maker tool, DFD, ER Diagram
- **Project Management tools**: Creative Pro office
- **Documentation tools**: DrExplain
- **Web Development tools**: Adobe Edge Inspect
- **Quality Assurance tools**: JMeter
- **Maintenance tools**: Bugzilla for defect tracking

CASE Tools

Advantages

- CASE tools improve quality and productivity of software.
- Produces system that more closely meet user needs and requirements.
- Produces system with excellent documentation.
- Tools are more effective for large scales systems. Produce more flexible system.
- CASE tools reduce the time for error correction and maintenance.

Disadvantages

- Very Complex
- Not easily maintainable
- Good quality CASE tools are very expensive.
- Require training of maintenance staff.

Functional and Non-functional Requirements

	Functional requirements	Nonfunctional requirements
Objective	Describe what the product does	Describe how the product works
End result	Define product features	Define product properties
Focus	Focus on user requirements	Focus on user expectations
Documentation	Captured in use case	Captured as a quality attribute
Essentiality	They are mandatory	They are not mandatory, but desirable
Origin type	Usually defined by user	Usually defined by developers or other tech experts
Testing	Component, API, UI testing, etc. Tested before nonfunctional testing	Performance, usability, security testing, etc. Tested after functional testing

User Requirements

- User requirements are statements in **natural language** along with corresponding **diagrams** (tables, forms) detailing the services provided by the system and operational constraints it must comply with.
- focus on the **user's needs**.
- It outlines the **activities a user can perform** with the system.
- It doesn't contain details of the system design and don't use technical expressions or formal notations.

System Requirements

- A structured document setting out detailed descriptions of the system services. **[Written as a contract between client and contractor]**
- It described as the expanded version of user requirements that engineers use at the beginning of a system's design.
- It serve as a blueprint for the developer to follow defining the parts of a system that are to be implemented.
- It can be written in natural language, and also possesses more structured forms and graphical notations

User and System Requirements Example

User requirement

- **UR1: A user of the work-based learning system shall be able to locate the person who is responsible for a document.**
- **UR2: A user of the work-based social networking system shall be able to control the amount of information about the user that the system presents to the other users like co-team member, group member and external for viewing.**

System requirement

- **SR1: The work-based learning system shall be able to retrieve the name, desk address, telephone number and email id fields of the record of a person responsible for the selected document.**
- **SR2: The work-based social networking system shall allow the user to present and hide fields from among the 48 data fields of the record of a person. The system presents these visible fields to the other users, based on three levels entitlements, namely co-team member, group member and external. The fields will be of view type only and not editable.**

Interface Specifications

- It is a formal description of how different software components or modules interact with each other.
- It defines the inputs and outputs of each component, as well as the expected behavior of each component when interacting with others.
- It serves as a contract between different components, ensuring that they can work together as intended.
- It also helps to reduce errors and misunderstandings during the development process by providing a clear and concise definition of the requirements for each component.

Types of Interface Specifications

- **Procedural specifications**: These are used to describe how different software components interact with each other through a set of defined methods and protocols. **[API specifications]**
- **User Interface (UI) specifications**: These are used to describe how users interact with a software system, including the design and behavior of graphical user interfaces (GUIs).
- **Protocol specifications**: These are used to describe how different software components communicate with each other over a network, including the format and structure of messages exchanged between them.

Requirement Elicitation

- It is a practice of collecting requirements of a system from the users, customers and other stakeholders.
- It is related to the various ways used to gain knowledge about the project domain and requirements.
- It deals with all the activities required in gathering the requirements of a system.
- The various sources of domain knowledge include customers, business manuals, and existing software of same type, standards and other stakeholders of the project.
- The developers and engineers work in close coordination with the customers and end-users to identify more about the problem to be solved and to bridge the gap between expectation and understanding.

Methods of Requirement Elicitation

- **Interviews:** Interviewers should be open minded willing to listen to stakeholders.
- **Questionnaire:** A document with pre-defined set of objective questions and respective options is handed over to all stakeholders to answer, which are collected and compiled.
- **Survey:** Organization may conduct surveys among various stakeholders by querying about their expectation and requirements from the upcoming system.
- **Domain Analysis:** Every software falls into some domain category. The expert people in the particular domain can be a great help to analyze general and specific requirements.
- **Observation:** observe the actual working of the existing similar type of installed systems
- **Brain Storming:** a spontaneous group debate to produce ideas and ways of solving problems.
- **Prototyping:** is building user interface without adding detail functionality for user to interpret the features of intended software products. It helps giving better idea of requirements.

Benefits of Requirement Elicitation

- Establishes the precise scope of work and the budget
- Avoids confusion during development
- Adds business value
- Reveals hidden and assumed requirements
- Allows for developing only relevant functionality:

Requirement Validation

- Checks that the right product is being built.
- Ensures that the software being developed (or changed) will satisfy its stakeholders.
- Checks that the software requirement specification against stakeholders goals and requirements.
- Certifies that the required document is an acceptable description of the system to be implemented.
- Checks a document for:
  - Completeness and consistency
  - Conformance to standards
  - Requirements conflicts
  - Technical errors
  - Ambiguous requirements

Requirement Validation Techniques

- **Requirements reviews**
- **Prototyping:** Using an executable model of the system to check requirements.
- **Test-case generation:** Developing tests for requirements to check stability
- **Walkthrough**: Involves a group of experts reviewing the requirements document and walking through it line by line.
- **User feedback**
- **Inspection**: Involves reviewing the requirements document with a group of experts, looking for errors, inconsistencies, and missing information.

SRS(Software Requirement Specification)

- It describes **WHAT** the system should do without describing **how** the software will do it.
- It is a document that captures complete description about how the system is expected to perform.
- It should include both a definition of user requirements and a specification of the system requirements.
- SRS document is known as black-box specification:
  - System is considered as a black box whose internal details are not known.
  - Only its visible external (i. e. input/output) behavior is documented.

Purpose of an SRS

- SRS precisely defines the software product that will be built.
- SRS used to know all the requirements for the software development and thus that will help in designing the software.
- It provides feedback to the customer. For example :
- communication between the Customer, Analyst, system developers, maintainers
- contract between Purchaser and Supplier
- firm foundation for the design phase
- support system testing activities
- support project management and control
- controlling the evolution of the system

SRS Features

- 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.

What should SRS address?

- **Functionality:** What is the software supposed to do?
- **External interfaces:** How does the software interact with people, the system's hardware and software?
- **Performance:** What is the speed, availability, response time, recovery time of various software functions?
- **Attributes:** What are the portability, correctness, maintainability, security etc. considerations?
- **Design constraints imposed on an implementation:** Are there any required standards in effect, implementation language, policies for database integrity, resource limits, operating environments?

What is not included in SRS?

- **Project requirements**: cost, delivery schedules, staffing, reporting procedures
- **Design solutions**: partitioning of SW into modules, choosing data structures
- **Product assurance plans**: Quality Assurance procedures, Configuration Management procedures, Verification & Validation procedures

SRS Format

Table of Contents

Requirement Management

- It is a process of eliciting, documenting, organizing, and controlling changes to the requirements of a software system throughout its lifecycle.
- It involves continuous evaluation and refinement of requirements to ensure that they meet needs of stakeholders, are feasible to implement, and are consistent with the system's overall goals and objectives.
- Ensure that all requirements are clearly defined, unambiguous, and measurable, so that they can be effectively communicated to all stakeholders.
- Trace requirements throughout the software development process to ensure that they are met and to provide visibility into the progress of the project.

Requirement Management

- It is also referred to as the activity of managing changing requirements during the development of the software system.
- It consists of:
  - Eliciting
  - Organizing
  - Documenting the requirement of the system and
  - Establishing and maintaining agreement between the customer and the project team on the changing requirements of the system

Advantages of Requirement Management

- Better control of complex projects
- Improved software quality
- Reduced project costs and delays
- Improved team communication
- Easing compliance with standards and regulations

Requirement Management Process

- Requirement management planning is an essential first stage in the requirement management process.
- Requirement management is very expensive and for each project, the planning stage establishes the level of requirements management detail which is required.

Advantages of Requirement Management Process

- Better control of complex projects
- Improved software quality
- Reduced project costs and delays
- Improved team communication
- Easing compliance with standards and regulations

Requirement Management Process Steps

1. **Requirement Identification:**

- Each requirement must be uniquely identified so that it can be cross referred by other requirement and so that it may be used in traceability assessments.

2. Change Management Process:

- This is the set of activities which assess the impact and cost of changes.

Requirement Management Process Steps

3. Traceability Policies:

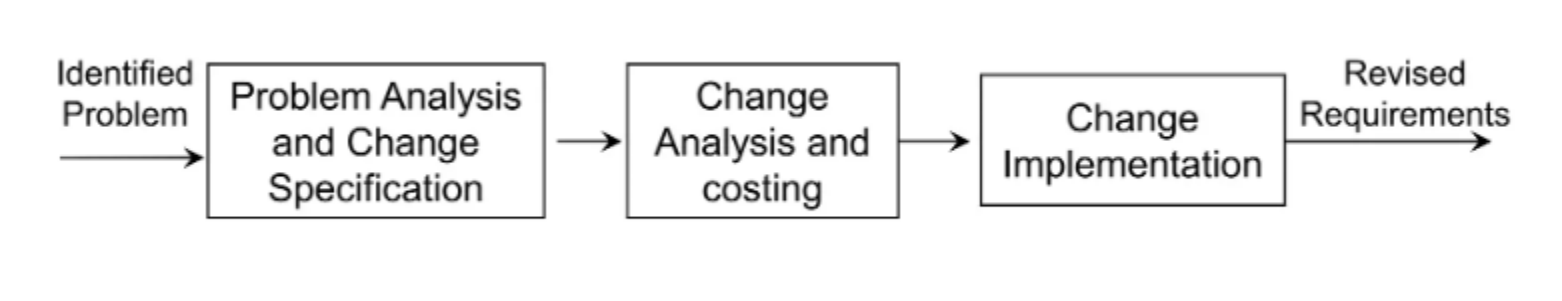
- These policies define the relationship between requirements and system design that should be recorded and how these records should be maintained.

4. Case Tool Support:

- Case tool support is required for:
  (a) Requirement storage
  (b) Change Management
  (c) Traceability Management

Requirement Change Management

- It is used when there is request or proposal for a change in requirements.
- The advantage of this process is that the changes to the proposals are managed consistently and in a controlled manner.
- An efficient requirements change management process undergoes a number of stages for changes to the requirements stated below:
  - ❑**Problem Analysis and Change Specification**
  - ❑**Change Analysis and Costing**
  - ❑**Change Implementation**



Requirement Change Management

1. **Problem Analysis and Change Specification:**

- The entire process begins with identification of problems to the requirements.
- The problem or proposal is analyzed to verify whether the change is valid. The outcome of the analysis is provided to change requester and more specific requirements change proposal is then made.

Requirement Change Management

2. Change Analysis and Costing

- The effect of a change requested on the requirement is assessed according to traceability information.
- The cost for this can be estimated on the basis of modification made to the design and implementation.
- After the analysis is over, a decision is made whether changes are to be made.

Requirement Change Management

3. Change Implementation

- Finally, the changes are made to the requirements document, system design and implementation.
- The requirements document is organized in such a manner so that changes to it can be made without extensive rewriting.
- Minimizing the external references and making document sections modular achieves changeability in the document. By doing this, individual sections can be changed and replaced without affecting other parts of the document.


THANK
YOU