
Software
Engineering

Unit 8: Software Engineering

- Introduction to Software Engineering: Characteristics, Emergence of Software Engineering, Software Metrics & Models, Process & Product Metrics. Software Life Cycle Models: Waterfall, Prototype and Spiral Models and their Comparison.
- Software Project Management: Size Estimation- LOC and FP Metrics, Cost Estimation-Delphi and Basic COCOMO, Introduction to Halstead's Software Science, Staffing Level Estimation- Putnam's Model. Software Requirements Specification: SRS Documents, their Characteristics and Organization.

Unit 8: Software Engineering

- Software Design: Classification, Software Design Approaches, Function Oriented Software Design, Structured Analysis- Data flow Diagrams and Structured Design, Introduction to Object Oriented Design.
- Coding and Testing of Software: Unit Testing, Block Box Testing, White Box Testing, Debugging, Program Analysis Tools, System Testing. Software Reliability and Quality Assurance: Reliability Metric- Musa's Basic Model.
- Software Quality Assurance: ISO 9000 and SEI CMM and their Comparison. Software Maintenance: Maintenance Process Models and Reverse Engineering, Estimation of Maintenance Costs.

Risk Resoution:

- Risk resolution is the execution of the plan for dealing with each risk.
- The risk has triggered the project manager need to execute the action plan.

Outputs of risk resolution phase:

- Risk status.
- Acceptable risks.
- Reduced rework.
- Corrective actions.
- Problem prevention.

Risk Monitoring:

- Risk monitoring is the continually reassessing of risks as the project proceeds and conditions change.
- RMMM Plan:
- Risk Mitigation , Monitoring and management in the software project plan or the risk management steps are organized.

Requirement Engineering Process:

- **Introduction:**
- Requirement engineering is the sub-discipline of software engineering that is concerned with determine the goal, functions and constraints of software system.

Requirements:

- Requirements management is a systematic approach to eliciting organizing and documenting the requirements of the systems.

Types of Requirements:

- System Requirements.
- User Requirements.

System Requirements:

- System requirements set out the systems functions, services and operational constraints in detail.
- It may be part of the constract betwwen the system buyer and the software developer.

Types of system requirements:

- Functional requirements.
- Non-functional Requirements.
- Domain Requirements.

Functional Requirements:

- The customer should provide statement of service. It should be clear how the system should react to particular inputs and how a particular system.

Problem of Functional Requirements:

- User Intention.
- Developer Interpretation.
- Requirements completness and consistency:

Non-Functional Requirements:

- The system properties and constraints various properties of a system can be: realiability, response tiime, storage requirements.

Types of Non-Functional Requirements:

- Product Requirements.
- Organizational Requirements.
- External Requirements.

Domain Requirements:

- Requirements can be application domain of the system, reflecting, characteristics of the domain.

Problem of Domain Requements:

- Understandability.
- Implicitness.

User Requirements:

- User requirements are defined using natural language lables and diagrams because these are the representation that can be undestood by all users.

- Client Managers.
- System End Users.
- Client Engineers.
- Contract Managers.

Problem of User Requirements:

- Lack of Clarity.
- Requirements Confusion.
- Requirements Mixture.

Software Requirement Specification:

- Software Requirements document is the specification of the system.
- It is not a design document.
- Requirements document is called SRS.

Users of SRS:

- Users, Customer and marketing Personnel.
- Software Developers.
- Test Engineers.
- Project Managers.
- Maintenance Engineers.

Software Requirement Specification:

- Software Requirement Specification (SRS) Format as the name suggests, is a complete specification and description of requirements of the software that need to be fulfilled for the successful development of the software system. These requirements can be functional as well as non-functional depending upon the type of requirement. The interaction between different customers and contractors is done because it is necessary to fully understand the needs of customers.

- **Purpose of this Document –** At first, main aim of why this document is necessary and what's purpose of document is explained and described.
- **Scope of this document –** In this, overall working and main objective of document and what value it will provide to customer is described and explained. It also includes a description of development cost and time required.
- **Overview –** In this, description of product is explained. It's simply summary or overall review of product.

- **The important parts of the Software Requirements Specification (SRS) document are:**

- Functional requirements of the system
- Non-functional requirements of the system, and
- Goals of implementation

FUNCTIONAL vs NONFUNCTIONAL 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
Essentiality	They are mandatory	They are not mandatory but desirable
Origin type	Usually defined by the 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
Types	Authentication, authorization levels, data processing, reporting, etc.	Usability, reliability, scalability, performance, etc.

- **Solution requirements**

➢ Functional requirements define what a product must do and what its features and functions are.

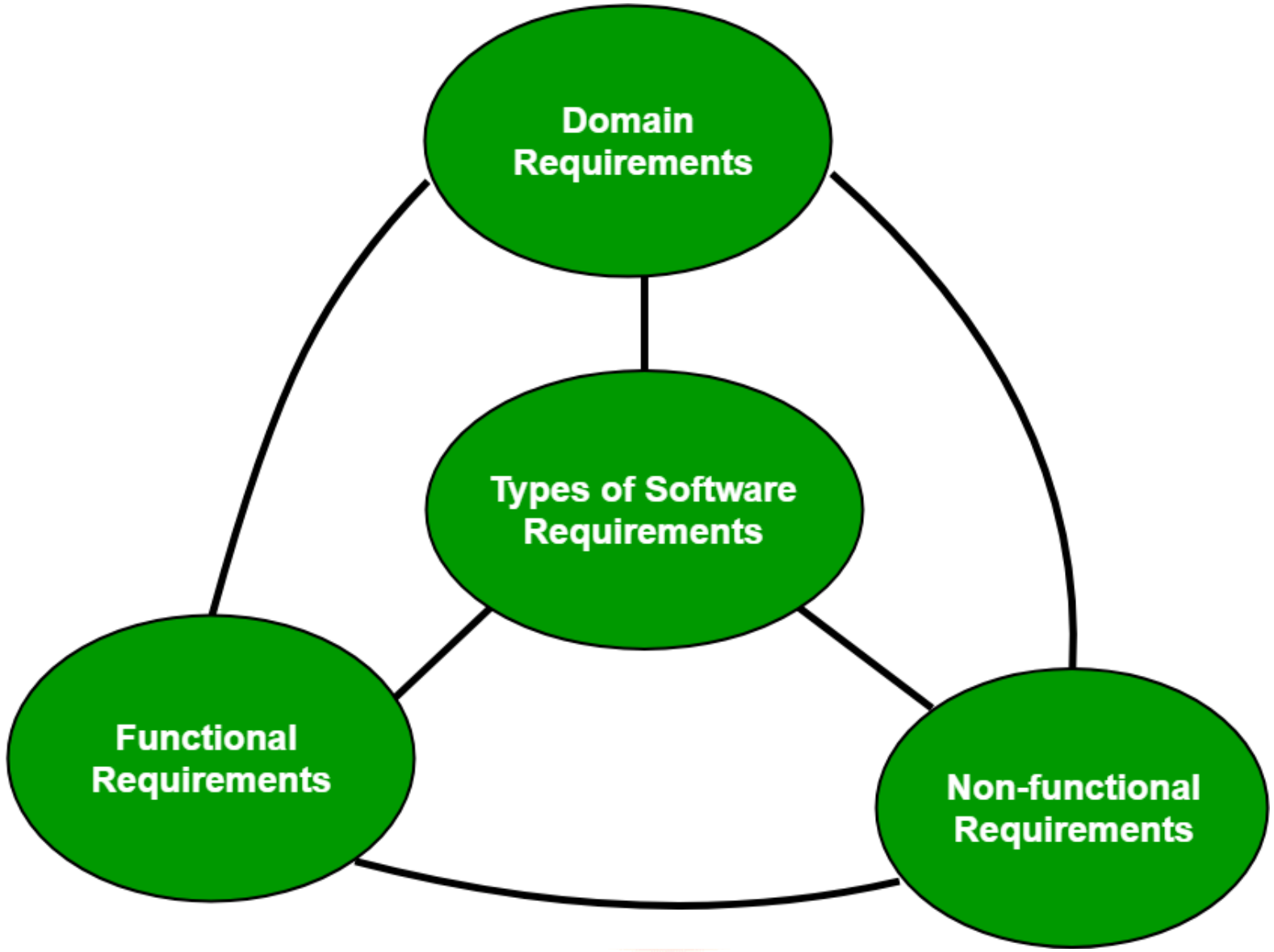
➢ Nonfunctional requirements describe the general properties of a system. They are also known as quality attributes.

• Goals of Implementation

- The goals of implementation part documents some general suggestions relating to development. These suggestions guide trade-off among style goals.

❖ The goals of the implementation section would possibly document problems like revisions to the system functionalities that will be needed within the future, new devices to be supported within the future, reusability problems, etc.

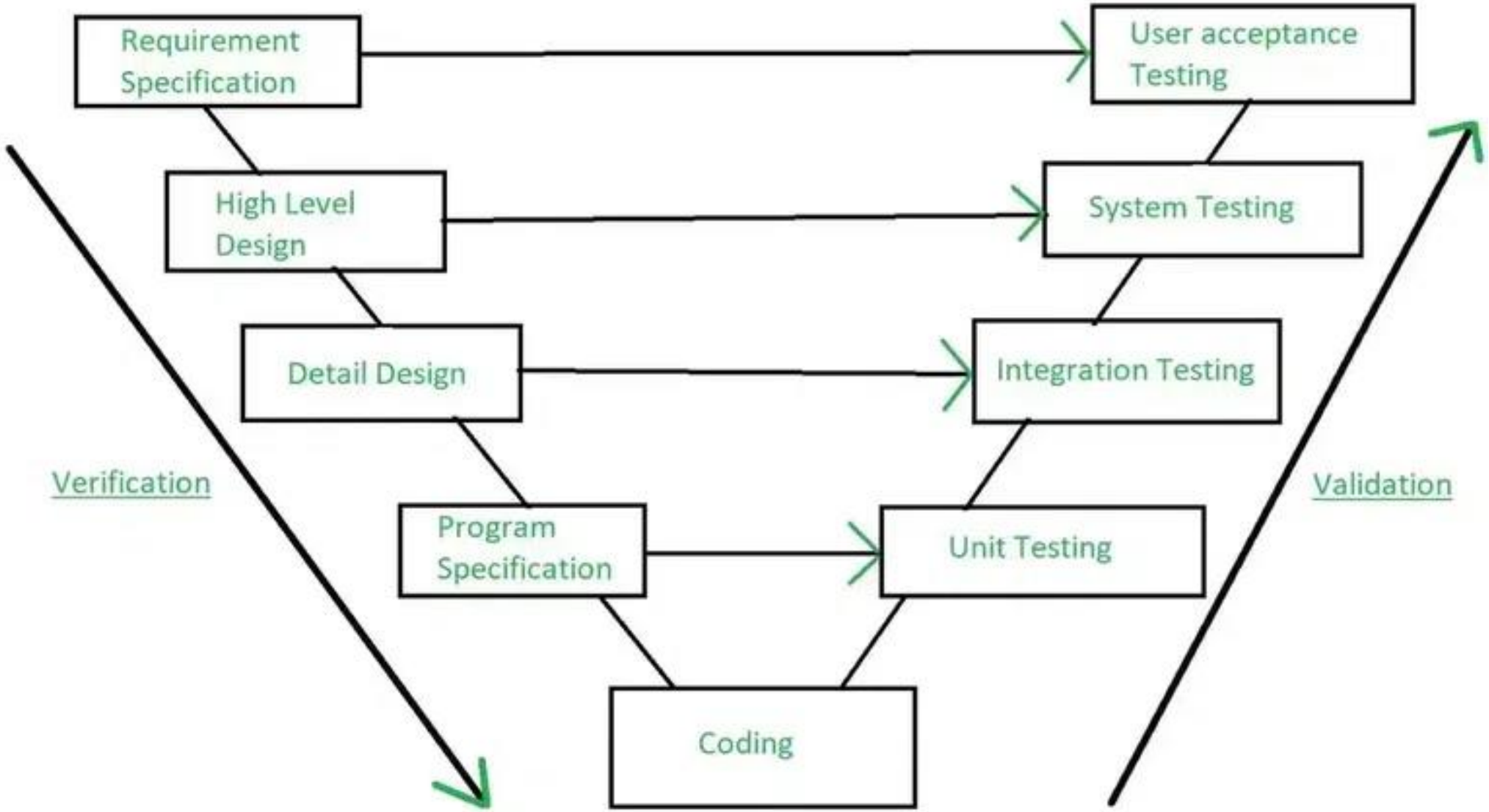
❖ These are the things that the developers would possibly detain their mind throughout development in order that the developed system may meet some aspects that don’t seem to be needed straightaway



- **Verification and Validation in Software Engineering**

- Verification and Validation is the process of investigating whether a software system satisfies specifications and standards and fulfills the required purpose. Barry Boehm described verification and validation as the following:

- Verification: Are we building the product, right?
- Validation: Are we building the right product?



- **COCOMO Model**

- Cocomo (Constructive Cost Model) is a regression model based on LOC, i.e., the number of Lines of Code.

- What is the Cocomo Model?
- The Cocomo Model is a procedural cost estimate model for software projects and is often used as a process of reliably predicting the various parameters associated with making a project such as size, effort, cost, time, and quality. It was proposed by Barry Boehm in 1981 and is based on the study of 63 projects, which makes it one of the best-documented models.

- **COCOMO Model**

- The key parameters that define the quality of any software products, which are also an outcome of the Cocomo are primarily Effort and schedule:

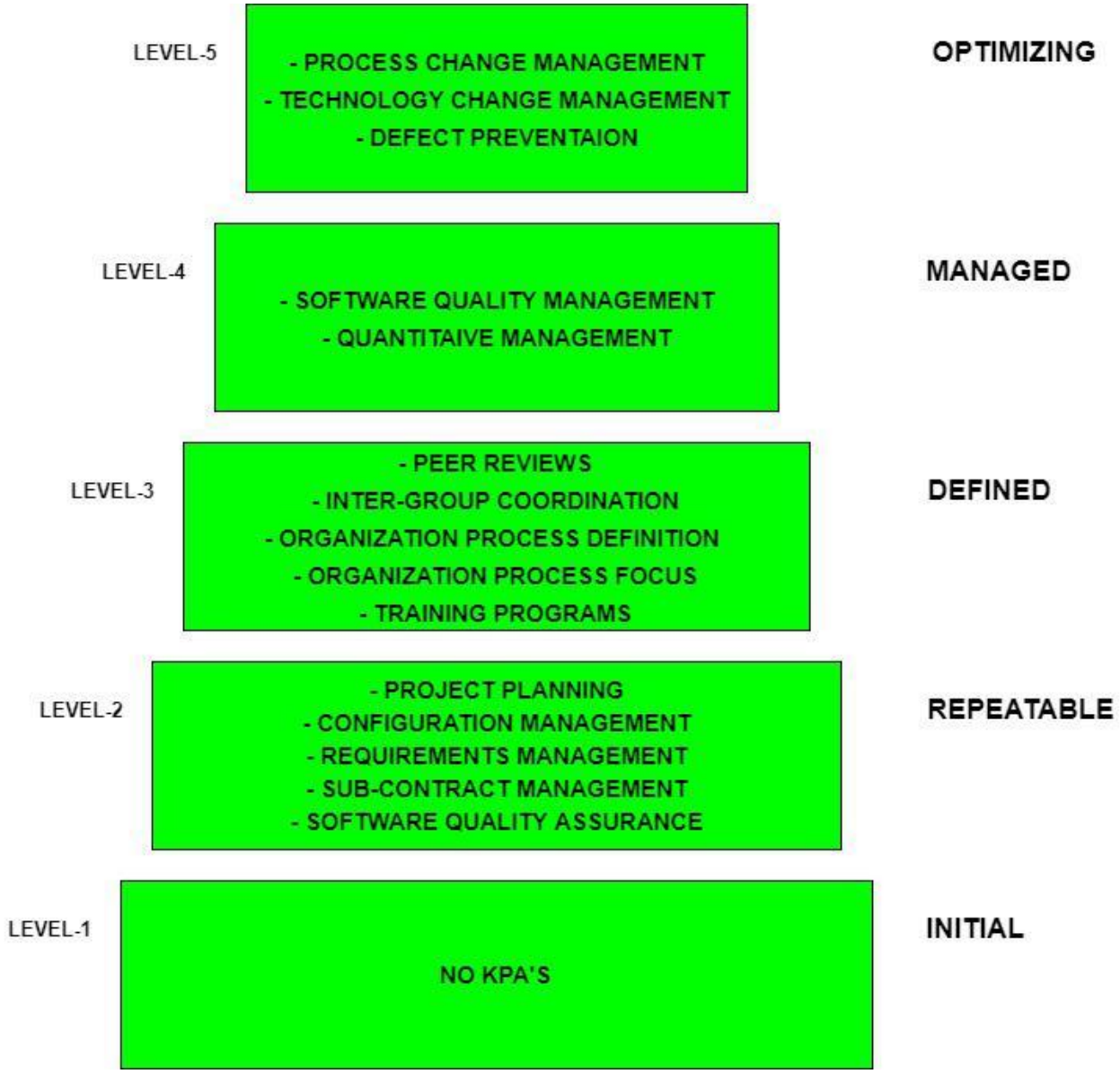
- **Effort:** Amount of labor that will be required to complete a task. It is measured in person-months units.

- **Schedule:** This simply means the amount of time required for the completion of the job, which is, of course, proportional to the effort put in. It is measured in the units of time such as weeks, and months

- **Capability Maturity Model (CMM)**

- Capability Maturity Model (CMM) was developed by the Software Engineering Institute (SEI) at Carnegie Mellon University in 1987. It is not a software process model.

- It is a framework that is used to analyze the approach and techniques followed by any organization to develop software products. It also provides guidelines to enhance further the maturity of the process used to develop those software products.



Aspects	Capability Maturity Model (CMM)	Capability Maturity Model Integration (CMMI)
Scope	Primarily focused on software engineering processes.	Expands to various disciplines like systems engineering, hardware development, etc.
Maturity Levels	Had a five-level maturity model (Level 1 to Level 5).	Initially had a staged representation; it introduced continuous representation later.
Flexibility	More rigid structure with predefined practices.	Offers flexibility to tailor process areas to organizational needs.
Adoption and Popularity	Gained popularity in the software development industry.	Gained wider adoption across industries due to broader applicability.

- **Software Quality Assurance – Software Engineering**

- Software Quality Assurance (SQA) is simply a way to assure quality in the software. It is the set of activities which ensure processes, procedures as well as standards are suitable for the project and implemented correctly.

- Software Quality Assurance is a process which works parallel to development of software. It focuses on improving the process of development of software so that problems can be prevented before they become a major issue. Software Quality Assurance is a kind of Umbrella activity that is applied throughout the software process.

- **Elements Of Software Quality Assurance:**

- **Standards:** The IEEE, ISO, and other standards organizations have produced a broad array of software engineering standards and related documents. The job of SQA is to ensure that standards that have been adopted are followed, and all work products conform to them.

- **Reviews and audits:** Technical reviews are a quality control activity performed by software engineers for software engineers. Their intent is to uncover errors. Audits are a type of review performed by SQA personnel (people employed in an organization) with the intent of ensuring that quality guidelines are being followed for software engineering work.

- **Elements Of Software Quality Assurance:**

- **Testing:** Software testing is a quality control function that has one primary goal—to find errors.

- **Error/defect collection and analysis:** SQA collects and analyzes error and defect data to better understand how errors are introduced and what software engineering activities are best suited to eliminating them.

- **Change management:** SQA ensures that adequate change management practices have been instituted. Security management: SQA ensures that appropriate process and technology are used to achieve software security.

- **Elements Of Software Quality Assurance:**

- **Safety:** SQA may be responsible for assessing the impact of software failure and for initiating those steps required to reduce risk.

- **Risk management:** The SQA organization ensures that risk management activities are properly conducted and that risk-related contingency plans have been established

Software is defined as _____

a) set of programs, documentation & configuration of data

b) set of programs

c) documentation and configuration of data

d) None of the mentioned

What does SDLC stands for?

a) System Design Life Cycle

b) Software Design Life Cycle

c) Software Development Life Cycle

d) System Development Life cycle

Which of the following document contains the user system requirements?

a) SRD

b) DDD

c) SDD

d) SRS

_____ specification is also known as SRS document.

a) white-box

b) grey-box

c) black-box

d) none of the mentioned

Which one of the following is a functional requirement ?

a) Maintainability

b) Portability

c) Robustness

d) None of the mentioned

Acceptance testing is also known as

a) Grey box testing

b) White box testing

c) Alpha Testing

d) Beta testing

Which of the following is non-functional testing?

a) Black box testing

b) Performance testing

c) Unit testing

d) None of the mentioned

Which of the following property does not correspond to a good Software Requirements Specification (SRS) ?

a) Verifiable

b) Ambiguous

c) Complete

d) Traceable

RAD stands for

a) Relative Application Development

b) Rapid Application Development

c) Rapid Application Document

d) None of the mentioned

What is the first step in the software development lifecycle?

A. System Design
B. Coding
C. System Testing
D. Preliminary Investigation and Analysis

What is the full form of the “COCOMO” model?

a) Cost Constructive Estimation Model

b) Constructive Cost Estimation Model

c) Constructive Case Estimation Model

d) Constructive Cost Estimating Model

Arrange the following activities to form a general software engineering process model.

I. Manufacture

II. Maintain

III. Test

IV. Install

V. Design

VI. Specification

A. 6, 5, 1, 3, 4, 2

B. 1, 2, 4, 3, 6, 5

C. 6, 1, 4, 2, 3, 5

D. 1, 6, 5, 2, 3, 4