QB MODULE 2

Q1

Benefits of Software Engineering

Efficient and Reliable Software – Outcome of SE is a software product that works correctly and meets user needs

Systematic Approach – Follows a structured and organized process for development.

Disciplined Development – Adheres to standards, methods, and guidelines, ensuring consistency.

Quantifiable Results – Software quality, performance, and attributes can be measured and improved.

Q2 Nature of Software

The nature of software has changed a lot over the years.

System software: System s/w is designed to provide a platform for other softwares. It is a interaction between h/w and application software. Eg. Android, MacOs, Windows.

Application software: It is a comp program designed to carry out a specific task as per user or business need. Eg. shopping app, banking app, messaging app, etc.

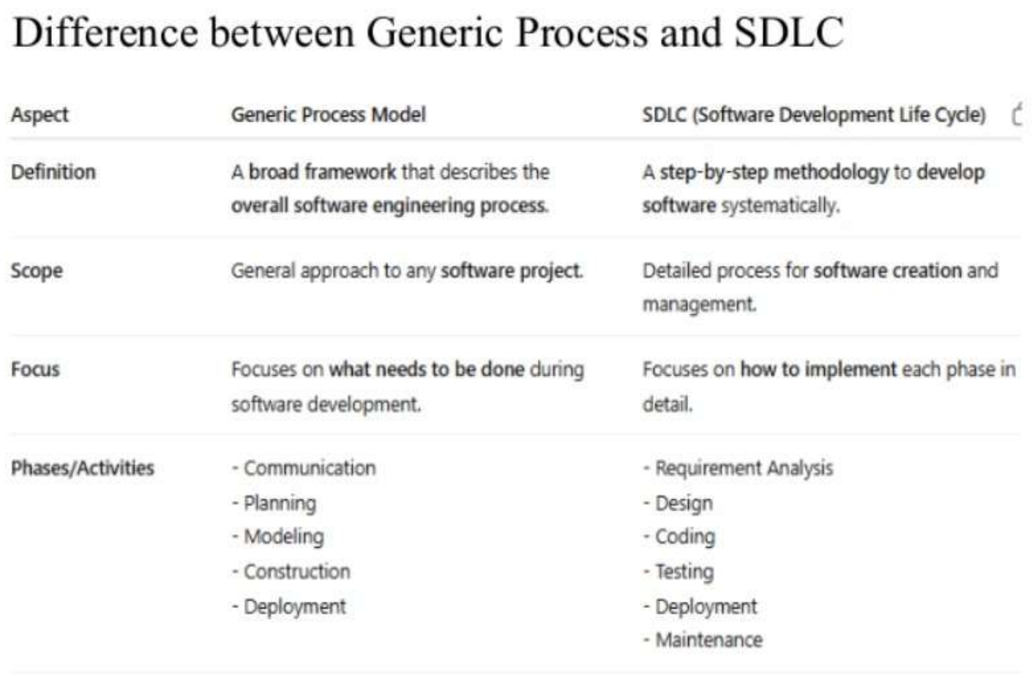
Real time software: These software are used to monitor, control and analyze real world events as they occur with real time data. It has very high accuracy, complex operations and data analysis. Eg. weather forecasting, stock market analysis, etc.

Embedded software: This type of software is placed in "Read-Only-Memory (ROM)" of the product and control the various functions of the product for the user. The product could be an aircraft, automobile, security system, signaling system, washing m/c, routers, switches, etc.

Artificial intelligence software: Artificial Intelligence software makes use of non numerical algorithms to solve complex problems. Eg. expert systems, artificial neural network, pattern recognition, alexa, siri, google cloud, etc.

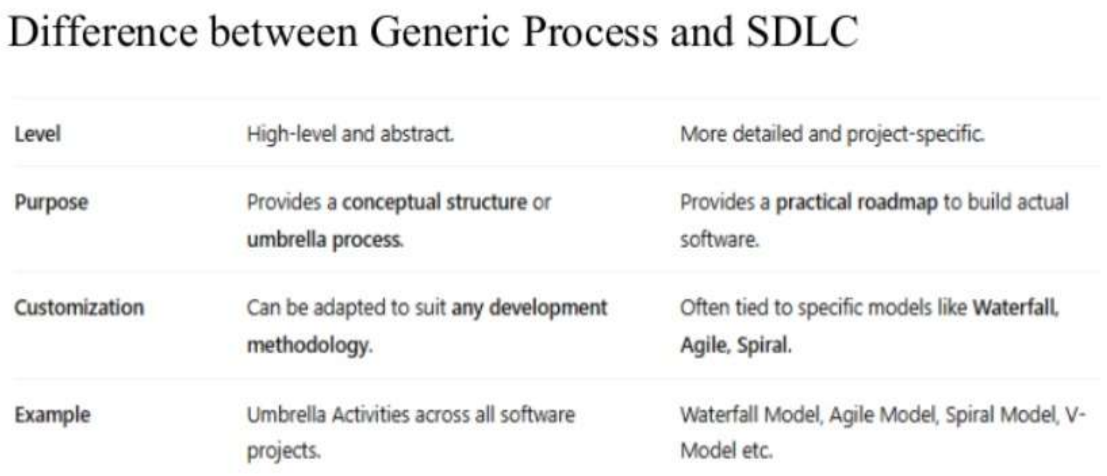
Web based software: The software related to web applications come under this category which uses client-server architecture. Eg. Gmail, Yahoo, etc.

Difference between Generic and SDLC



Difference between Generic Process and SDLC

Aspect	Generic Process Model	SDLC (Software Development Life Cycle)
Definition	A **broad framework** that describes the **overall software engineering process.**	A **step-by-step methodology** to **develop software** systematically.
Scope	General approach to any **software project.**	Detailed process for **software creation** and management.
Focus	Focuses on **what needs to be done** during software development.	Focuses on **how to implement** each phase in detail.
Phases/Activities	- Communication - Planning - Modeling - Construction - Deployment	- Requirement Analysis - Design - Coding - Testing - Deployment - Maintenance



Difference between Generic Process and SDLC

Level	High-level and abstract.	More detailed and project-specific.
Purpose	Provides a **conceptual structure** or **umbrella process.**	Provides a **practical roadmap** to build actual software.
Customization	Can be adapted to suit **any development methodology.**	Often tied to specific models like **Waterfall, Agile, Spiral.**
Example	Umbrella Activities across all software projects.	Waterfall Model, Agile Model, Spiral Model, V-Model etc.

Q3.Explain the characteristics of software

Answer:

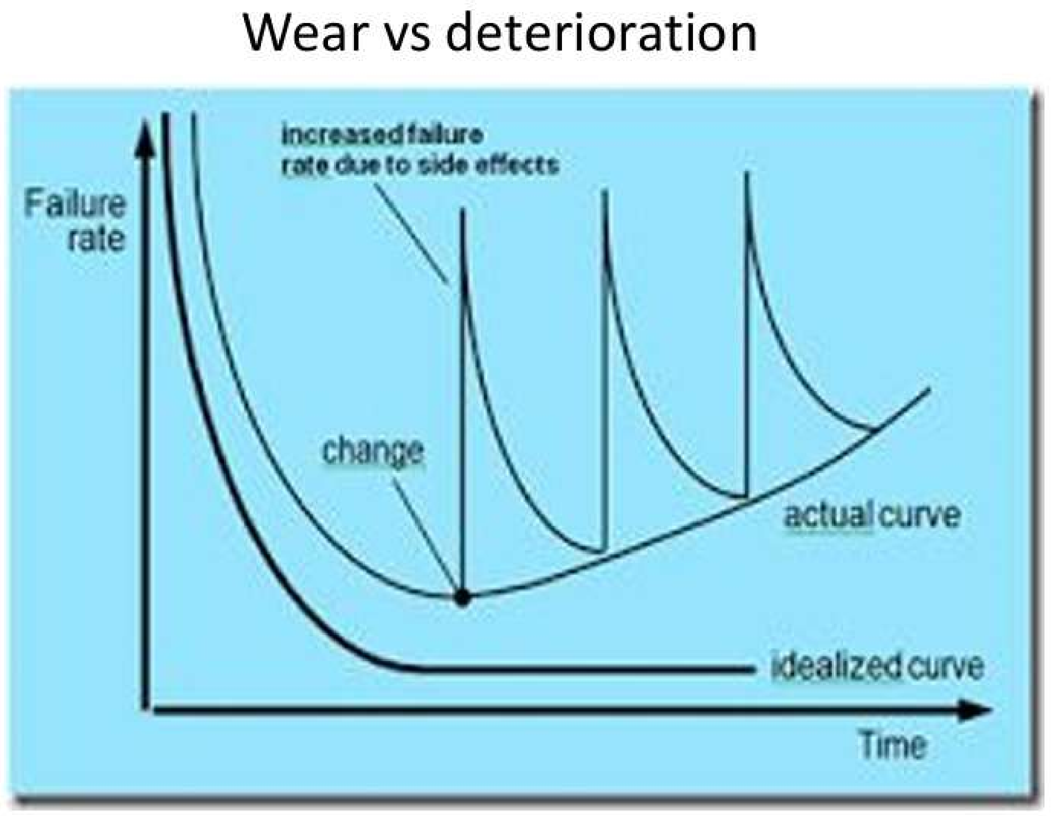
Software is instructions (computer programs) that when executed provide desired features, function, and performance.

Software consists of data structures that enable the programs to adequately manipulate information.

Software contains documentation that describes the operation and use of the programs.

Unlike hardware, software does not wear out, but it can deteriorate with time when modifications, enhancements, and fixes are added.

It is intangible, custom-built, and evolves over time.



(Page 11)

Q4. Describe the Generic Process Model

Answer:

A generic process framework for software engineering defines five framework activities—

Communication: Heavy communication with customers and stakeholders; requirement gathering is done. It determines the system objectives, software requirements and features.

Planning: Estimation of project resources, schedule, and tracking. It includes technical tasks, work schedule, risks, and required resources.

Modeling: Requirement analysis and design of the project like algorithms, flowcharts, etc. If changes are required, they are implemented here.

Construction: Code generation and testing. Implementing design details using programming languages and testing to check structured logic and desired output.

Deployment: Delivering the product to customers for evaluation and feedback. Based on feedback, modifications are done.

Additionally, a set of umbrella activities such as project tracking, risk management, quality assurance, configuration management, and technical reviews are applied throughout the process.

(Page 15–18)

Q5 Explain Software Development Life Cycle (SDLC) in detail

Answer:

SDLC represents the process of developing quality software as per the customer requirements.

Phases of SDLC:

Requirement Analysis & Planning: Conducted by product manager or business analyst. Involves understanding customer needs, creating SRS, identifying risks, and planning schedule/cost.

Design: Conducted by project managers and UI/UX designers. Converts requirements into high-level design (architecture, DB design, modules) and low-level design (inputs, outputs, interface details). Documented in DDS (Design Document Specification).

Implementation: Done by developers using appropriate languages (Java, Python, etc.) and databases (SQL, Oracle, etc.).

Testing: Performed by QA team (integration, black box, white box, regression, performance testing). Ensures the system meets customer requirements.

Deployment & Maintenance: Final software is deployed in the customer environment. Issues and updates are handled in maintenance phase.

(Page 19–22)

Q6. Explain Waterfall Model with advantages and disadvantages

Answer:

The Waterfall Model is a linear sequential development model (proposed by Winston Royce). It was the first SDLC model. Each phase must be completed before moving to the next.

When to use: When requirements are clear and constant, project is small, environment is stable, and resources are trained.

Phases: Requirement Analysis → Design → Development → Testing → Deployment & Maintenance.

Advantages:

Simple and easy to understand.

Works well for smaller projects.

Requirements are well understood and fixed.

Process and results are well documented.

Disadvantages:

High risk and uncertainty.

Very difficult to move back to previous phases.

Not suitable for complex and object-oriented projects.

Client feedback cannot be included during development.

Testing comes late in the cycle.

(Page 23–27)

Q7. Explain Incremental Model with advantages and disadvantages

Answer:

The Incremental model divides requirements into multiple standalone modules. Each module goes through requirement, design, implementation, and testing phases. Every release adds functions until the complete system is achieved.

When to use: When requirements are known upfront, when quick delivery is needed, when project has a long schedule, or when web/product-based companies develop applications.

Advantages:

Client gets important functionality early.

Errors are easier to detect.

Easier to test and debug.

Risk is handled in each iteration.

Flexible and less expensive to change requirements.

Disadvantages:

Well-defined module interfaces are needed.

Proper planning and skilled teams are required.

Each iteration is rigid and does not overlap.

(Page 30–34)

Q8. Explain Spiral Model with advantages and disadvantages

Answer:

The Spiral Model was proposed by Boehm in 1986. It is also called a Meta Model, combining Waterfall, Iterative, and Prototyping models. It is risk-driven and suited for large, complex, and high-risk projects.

Phases:

Planning (requirement gathering, cost, and schedule).

Risk Analysis (identify potential risks, mitigation strategies, and prototypes).

Engineering & Execution (coding, testing, deployment).

Evaluation (customer feedback, next iteration).

Advantages:

Strong focus on risk analysis.

Useful for complex and critical projects.

Allows use of prototypes.

Customer feedback is included.

Changing requirements can be accommodated.

Disadvantages:

Needs highly skilled teams.

Expensive model.

Not suitable for small projects.

May become infinite with repeated spirals.

(Page 40–43)

Q9. Explain RAD Model with advantages and disadvantages

Answer:Rapid Application Development (RAD) was proposed by IBM in the 1980s. It focuses on quick prototyping, iterative development, high user involvement, and parallel development.

Phases: Business Modeling → Data Modeling → Process Modeling → Application Modeling → Testing & Turnover.

Advantages:

Reduces cycle time of projects.

Customer feedback is quickly included.

Produces better quality in shorter time.

Flexible and adapts to changes.

Disadvantages:

Requires highly skilled developers.

Higher cost.

Not suitable for small projects.

Needs close cooperation between users and developers.

(Page 36–39)

Q 10. Compare RAD and Waterfall Model

Answer:

Approach: Waterfall is linear and sequential, RAD is iterative and parallel.

Flexibility: Waterfall is rigid; RAD is adaptable.

Customer Involvement: Waterfall has little; RAD has high user involvement.

Speed: Waterfall is slow; RAD is fast due to prototyping.

Risk Handling: Waterfall is high risk if requirements change; RAD adapts easily.

(Page 45)

Q 11. Compare Spiral with other models

Answer:

Spiral vs. Waterfall: Spiral is iterative with risk analysis; Waterfall is linear with no risk focus.

Spiral vs. Incremental: Spiral emphasizes risk management; Incremental emphasizes partial builds.

Spiral vs. RAD: Spiral is risk-driven; RAD is speed-driven with prototypes.

(Page 45–47)

Q12. Explain advantages of Waterfall and Spiral Model with examples

Answer:

Waterfall Advantages:

Simple, structured, and works when requirements are fixed.

Good for projects where documentation and compliance are required.

Examples: Government projects, Banking systems, Healthcare software, Telecom systems, Education software, Embedded systems.

Spiral Advantages:

Handles risks early.

Suitable for large, complex, critical projects.

Examples: Aerospace & defense industry, Healthcare systems, Microsoft Windows, IBM WebSphere, Avionics and satellite communication systems.

(Page 27 & 42–43)