Question 1: What is/are the characteristics of a system?

A. Integration

B. Interdependence

C. Interaction

D. All of the above

Question 2: Orderly grouping of independent components linked together

according to a plan to achieve a specific objective is called .

A. Interconnection

B.Relationship

C.Data

D. System

Question 3: Decision support systems (DSS) are essential for .

A. Providing statutory information.

B. The day-to-day operation of an organization.

C. Ensuring the organization remains profitable.

D. Top-level strategic decision-making

Question 4: is a computerized information system used to support

decision- making in an organization.

A. DSS

B. MSS

C. OSS

D. HSS

Question 5: In data-flow diagrams (DFD) external entities are represented via

A. Rectangle

B. Ellipse

C. Diamond-shaped box

D. Circle

Question 6: SDLC stands for

A. System Development Life-cycle

B. Structure Development Life-cycle

C. System Design Life-cycle

D. Structure Design Life-cycle

Question 7: Which of the following principles of project management defines

and controls the functions that are to be included in the system?

A. Project quality management

B. Project time management

C. Project cost management

D. Project scope management

Question 8: A context diagram

A. Describes the context of a system

B. is a DFD that gives an overview of the system

C. is a detailed description of a system

D. is not used in drawing a detailed DFD

Question 9: HIPO stands for

A. Hierarchy input plus output

B. Hierarchy plus input process output

C. Hierarchy input process output

D. Hierarchy input-output process

Question 10: What is/are the advantages of system flowcharts?

A. Effective communication

B. Effective analysis

C. Queasier group of relationships

D. All of the above

Question 11: DDS stands for

A.Data Dictionary Systems

B. Data Data Systems

C. Data Digital Systems

D. Digital Data Service

Question 12: A DFD is normally leveled as

A. Good idea in designing

B. Recommended by experts

C. Easier to read and understand the number of smaller DFDs than one large

DFD

D. East to do

Question 13: What is the tabular method for describing the logic of the decisions to be taken?

A. Decision Tree

B. Decision Tables

C. Decision Data

D. Decision Method

Question 14: MDP stands for

A. Master Design Program

B. Master Development Plan

C. Master Database Plan

D. Mandatory Database Program

Question 15: Which property is possessed by a physical DFD?

A. It does not concern itself with material flow.

B. It has no means of showing material flow.

C. It can show only stored material.

D. It can show the flow of material

Question 16: The decision-making model was proposed by

A. Herbert A Simon

B. Recon Michal

C. Harry Goode

D. Lampart Lock

Question 17: is a sort of blueprint of the System Development Effort.

A. DMP

B. MPD

C. DPM

D. MDP

Question 18: Data stored in a data flow diagram represents

A. a disk store

B. a sequential file

C. a random access memory

D. a repository of data

Question 19: includes a review of existing procedures and information flow.

A. Feasibility Study

B. System Analysis

C. System Design

D. Feasibility Analysis

Question 20: A system that is a part of the larger system is known as

A.System Unit

B.Subsystem

C. System Element

D. None of the above

Question 21: During a system audit, the system performance is compared to

A.Competing systems

B. Newer systems

C. Design specifications

D. Similar systems

Question 22: Which is not considered a tool in the system design phase

A. Pie Chart

B. Data-Flow Diagram

C. System Flowchart

D. Decision Table

Question 23: The term DFD is/are referred to as

A. Physical DFD

B. Logical DFD

C. Practical DFD

D. Both Physical DFD and Logical DFD

Question 24: A feasibility study is used to determine the proposed systems.

A. Resource requirements

B. Availability of hardware and software

C. Costs and benefits

D. All of the above

Question 25: The design phase is followed by

A. Feasibility study

B. Initial Investigation

C. Implementation

D. Analysis

Question 26: What technique is used to express the requirements of the system

in graphical form?

A. SFD

B. DFD

C. AFD

D. Context Diagram

Question 27: Arrow in DFD represents

A. Data flow

B. Data store

C. Process transforming data flow

D. Source/Destination of Data

Question 28: Cost-benefit analysis is performed during which phase

A. Analysis phase

B. Feasibility study phase

C. Implementation Phase

D. Design Phase

Question 29: What are pilgrimage basic concepts of ER diagram?

A.Entity

B. Relationships

C. Attributes

D. All of the above

Question 30: is a realistic approach to the development of large-scale software products because the software evolves as the process progresses.

A. Iterative Model

B. Spiral Model

C. V Model

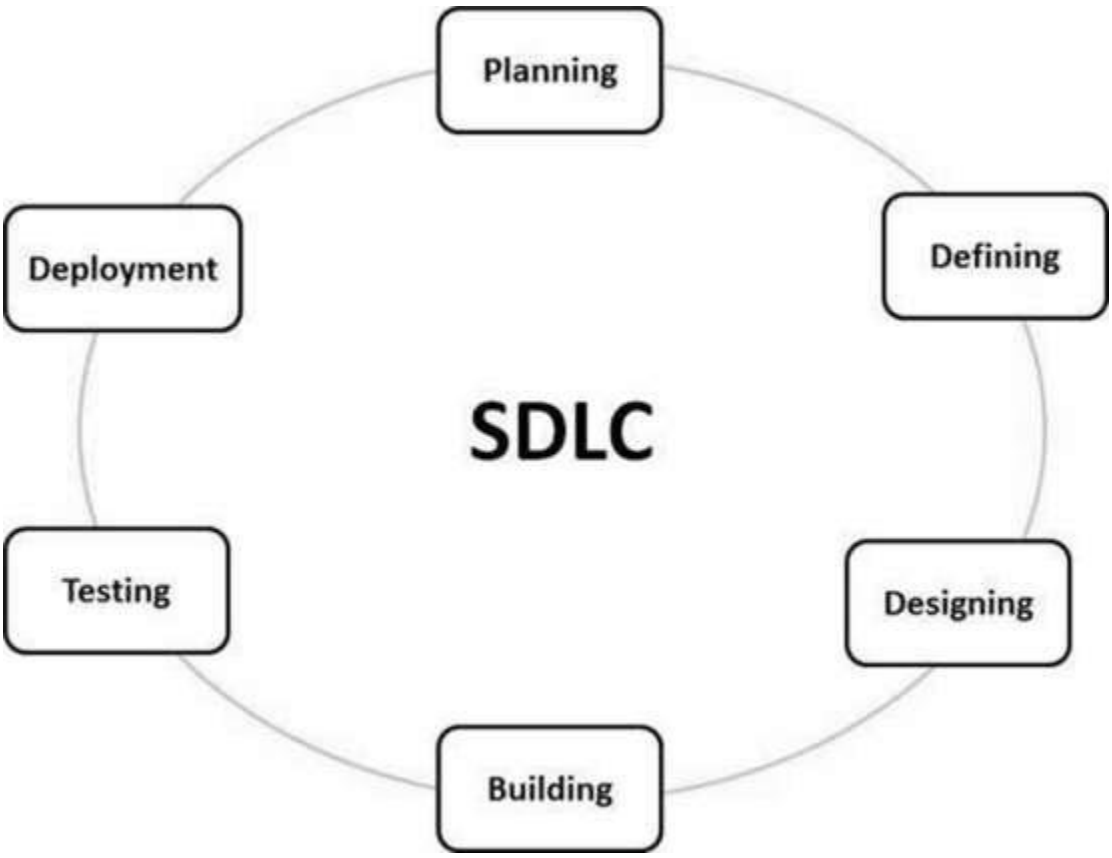
D. Waterfall Model

Long answer

1) What is SDLC? Explain with suitable block diagram.

SDLC is a process followed for a software project, within a software organization. It consists of a detailed plan describing how to develop, maintain, replace and alter or enhance specific software. The life cycle defines a methodology for improving the quality of software and the overall development process.

The following figure is a graphical representation of the various stages of a typical SDLC.



A typical Software Development Life Cycle consists of the following stages −

Stage 1: Planning and Requirement Analysis

Requirement analysis is the most important and fundamental stage in SDLC. It is performed by the senior members of the team with inputs from the customer, the sales department, market surveys and domain experts in the industry. This information is then used to plan the basic project approach and to conduct product feasibility study in the economical, operational and technical areas.

Planning for the quality assurance requirements and identification of the risks associated with the project is also done in the planning stage. The outcome of the technical feasibility study is to define the various technical approaches that can be followed to implement the project successfully with minimum risks.

Stage 2: Defining Requirements

Once the requirement analysis is done the next step is to clearly define and document the product requirements and get them approved from the customer or the market analysts. This is done through an **SRS (Software Requirement Specification)** document which consists of all the product requirements to be designed and developed during the project life cycle.

Stage 3: Designing the Product Architecture

SRS is the reference for product architects to come out with the best architecture for the product to be developed. Based on the requirements specified in SRS, usually more than one design approach for the product architecture is proposed and documented in a DDS - Design Document Specification.

This DDS is reviewed by all the important stakeholders and based on various parameters as risk assessment, product robustness, design

modularity, budget and time constraints, the best design approach is selected for the product.

A design approach clearly defines all the architectural modules of the product along with its communication and data flow representation with the external and third party modules (if any). The internal design of all the modules of the proposed architecture should be clearly defined with the minutest of the details in DDS.

Stage 4: Building or Developing the Product

In this stage of SDLC the actual development starts and the product is built. The programming code is generated as per DDS during this stage. If the design is performed in a detailed and organized manner, code generation can be accomplished without much hassle.

Developers must follow the coding guidelines defined by their organization and programming tools like compilers, interpreters, debuggers, etc. are used to generate the code. Different high level programming languages such as C, C++, Pascal, Java and PHP are used for coding. The programming language is chosen with respect to the type of software being developed.

Stage 5: Testing the Product

This stage is usually a subset of all the stages as in the modern SDLC models, the testing activities are mostly involved in all the stages of SDLC. However, this stage refers to the testing only stage of the product where product defects are reported, tracked, fixed and retested, until the product reaches the quality standards defined in the SRS.

Stage 6: Deployment in the Market and Maintenance

Once the product is tested and ready to be deployed it is released formally in the appropriate market. Sometimes product deployment happens in stages as per the business strategy of that organization. The product may first be released in a limited segment and tested in the real business environment (UAT- User acceptance testing).

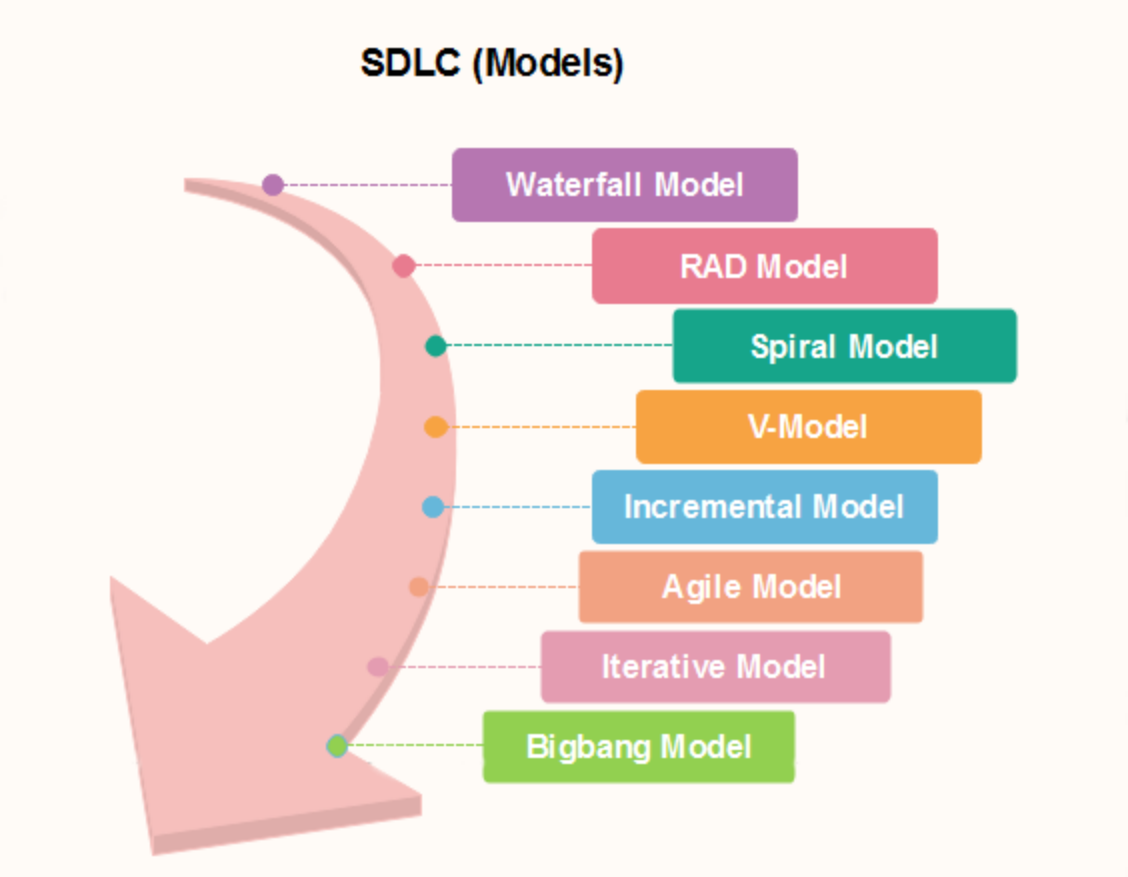
Then based on the feedback, the product may be released as it is or with suggested enhancements in the targeting market segment. After the product is released in the market, its maintenance is done for the existing customer base.

2)What are SDLC models? Describe briefly.

software Development life cycle (SDLC) is a spiritual model used in project management that defines the stages include in an information system development project, from an initial feasibility study to the maintenance of the completed application.

There are different software development life cycle models specify and design, which are followed during the software development phase. These models are also called "**Software Development Process Models**." Each process model follows a series of phase unique to its type to ensure success in the step of software development.

Here, are some important phases of SDLC life cycle:



Waterfall Model

The waterfall is a universally accepted SDLC model. In this method, the whole process of software development is divided into various phases.

The waterfall model is a continuous software development model in which development is seen as flowing steadily downwards (like a waterfall) through the steps of requirements analysis, design, implementation, testing (validation), integration, and maintenance.

Linear ordering of activities has some significant consequences. First, to identify the end of a phase and the beginning of the next, some certification techniques have to be employed at the end of each step. Some verification and validation usually do this mean that will ensure that the output of the stage is consistent with its input (which is the output of the

previous step), and that the output of the stage is consistent with the overall requirements of the system.

RAD Model

RAD or Rapid Application Development process is an adoption of the waterfall model; it targets developing software in a short period. The RAD model is based on the concept that a better system can be developed in lesser time by using focus groups to gather system requirements.

- Business Modeling
- Data Modeling
- Process Modeling
- Application Generation
- Testing and Turnover

Spiral Model

The spiral model is a **risk-driven process model**. This SDLC model helps the group to adopt elements of one or more process models like a waterfall, incremental, waterfall, etc. The spiral technique is a combination of rapid prototyping and concurrency in design and development activities.

Each cycle in the spiral begins with the identification of objectives for that cycle, the different alternatives that are possible for achieving the goals, and the constraints that exist. This is the first quadrant of the cycle (upper-left quadrant).

The next step in the cycle is to evaluate these different alternatives based on the objectives and constraints. The focus of evaluation in this step is based on the risk perception for the project.

The next step is to develop strategies that solve uncertainties and risks. This step may involve activities such as benchmarking, simulation, and prototyping.

V-Model

In this type of SDLC model testing and the development, the step is planned in parallel. So, there are verification phases on the side and the validation phase on the other side. V-Model joins by Coding phase.

Incremental Model

The incremental model is not a separate model. It is necessarily a series of waterfall cycles. The requirements are divided into groups at the start of the project. For each group, the SDLC model is followed to develop software. The SDLC process is repeated, with each release adding more functionality until all requirements are met. In this method, each cycle act as the maintenance phase for the previous software release. Modification to the incremental model allows development cycles to overlap. After that subsequent cycle may begin before the previous cycle is complete.

Agile Model

Agile methodology is a practice which promotes continues interaction of development and testing during the SDLC process of any project. In the Agile method, the entire project is divided into small incremental builds. All of these builds are provided in iterations, and each iteration lasts from one to three weeks.

Any agile software phase is characterized in a manner that addresses several key assumptions about the bulk of software projects:

1. It is difficult to think in advance which software requirements will persist and which will change. It is equally difficult to predict how user priorities will change as the project proceeds.
2. For many types of software, design and development are interleaved. That is, both activities should be performed in tandem so that design models are proven as they are created. It is difficult to think about how much design is necessary before construction is used to test the configuration.
3. Analysis, design, development, and testing are not as predictable (from a planning point of view) as we might like.

Iterative Model

It is a particular implementation of a software development life cycle that focuses on an initial, simplified implementation, which then progressively gains more complexity and a broader feature set until the final system is complete. In short, iterative development is a way of breaking down the software development of a large application into smaller pieces.

3) What is information system and its type? Write down the components of

information system.

An Information System (IS) is indeed a crucial framework designed to collect, process, store, and distribute information within an organization. Here's a breakdown of its components based on your description:

4. **Computer Hardware**: This includes physical devices such as computers, input/output devices (like keyboards and monitors), storage devices (hard drives), and networking components (routers, switches). Hardware is essential for processing data and executing software instructions.
5. **Computer Software**: Software refers to the programs and applications that control and coordinate the hardware components. It can be categorized into:
    - **System Software**: Manages the hardware and provides a platform for running applications. Examples include operating systems (Windows, macOS) and utilities.
    - **Application Software**: Specific programs designed to perform tasks for users, like word processors, databases, and web browsers.
    - **Procedures**: These are the policies and protocols that govern the use of hardware and software systems within an organization.
6. **Databases**: Databases are organized collections of data, typically managed by Database Management Systems (DBMS). They store, retrieve, and update data, providing a structured way to handle large amounts of information efficiently.
7. **Networks**: Networks consist of interconnected devices and communication mediums that allow information to be shared and accessed. This includes local area networks (LANs), wide area networks (WANs), and the internet. Network components include hardware (routers, switches, cables) and software (protocols, servers).
8. **Human Resources**: People are crucial to the operation and management of information systems. They include end-users who utilize the information generated by the system for their tasks, as well as professionals who develop, maintain, and support the IS

infrastructure. Roles can range from system analysts and programmers to network administrators and end-users across various departments.

In summary, an Information System is a comprehensive structure that integrates hardware, software, data, networks, and people to support organizational processes by facilitating the collection, processing, and dissemination of information. Its primary goal is to provide timely and accurate information to support decision-making and operational activities within an organization.

4) Prototyping in the Software Development Life Cycle (SDLC) is an iterative process where a preliminary version of a system or its part is built to test various aspects of its design, functionality, and usability before the final system is developed. Here's a detailed explanation of prototyping in the SDLC process:

Purpose of Prototyping:

1. **Requirements Clarification**: Prototyping helps in eliciting and clarifying user requirements by providing a tangible representation of the proposed system early in the development process. Users can interact with the prototype to better understand their needs and provide feedback.

2. **Risk Reduction**: It mitigates risks by allowing stakeholders to identify potential issues and challenges early. This early feedback helps in refining the requirements and design, reducing the likelihood of costly changes during later stages of development.

3. **Improving Design**: Prototyping enables designers and developers to experiment with different design alternatives and solutions. It facilitates exploration of various functionalities and user interfaces to find the optimal design that meets user expectations.

4. **User Involvement**: Users can actively participate in the development process by evaluating and testing the prototype. Their input is valuable for ensuring that the final system meets their needs and preferences.

Steps Involved in Prototyping:

1. **Identify Requirements**: Gather initial requirements and identify the key functionalities that the prototype should demonstrate.

2. **Develop Initial Prototype**: Build a basic version of the system or a part of it that showcases essential features and functionalities.

3. **Review and Refine**: Review the prototype with stakeholders, including users, developers, and designers. Collect feedback on usability, functionality, and design.

4. **Revise and Enhance**: Based on feedback, revise the prototype to address issues and incorporate suggested changes. This may involve adding new features, modifying existing ones, or refining the user interface.

5. **Repeat**: Iterate through the prototyping process as necessary to refine the system until stakeholders are satisfied with the prototype.

Types of Prototypes:

1. **Throwaway or Rapid Prototyping**: Quick prototypes built with the intention of discarding them after gathering feedback. They focus on demonstrating specific features or solving particular design challenges.

2. **Evolutionary Prototyping**: Incrementally refined prototypes that evolve into the final system. Each iteration incorporates feedback and adds new features until the complete system is developed.

Advantages of Prototyping:

- **Early Detection of Issues**: Identifying and addressing issues early reduces rework and improves overall system quality.
- **User Involvement**: Encourages active participation of users, leading to a system that better meets their expectations and needs.
- **Clarity in Requirements**: Helps in clarifying and refining requirements through tangible demonstrations rather than abstract discussions.
- **Enhanced Communication**: Improves communication among stakeholders by providing a visual representation of the system.

Limitations of Prototyping:

- **Time and Cost**: Depending on the complexity, prototyping can be time-consuming and may require significant resources.
- **Scope Creep**: There's a risk of expanding the scope of the prototype beyond the original objectives, leading to delays and increased costs.
- **Documentation**: Prototypes may not always result in detailed documentation, which can be challenging for future maintenance and scalability.

In conclusion, prototyping in the SDLC process is a valuable technique for gathering requirements, reducing risks, and refining designs through iterative development cycles. It enhances user involvement and communication, ultimately leading to the development of more effective and user-friendly systems.

Certainly! Here are the answers to each question:

6. **Explain different environments related to development when following SDLC:**

In the Software Development Life Cycle (SDLC), different environments are used to manage the progression of software from initial development to deployment and maintenance. These environments typically include:

- **Development Environment**: Where developers write and test code. It's isolated and may have tools for debugging and version control.

- **Testing Environment**: Where testers validate the software against requirements and use cases. It mimics the production environment but is usually not accessed by end-users.

- **Staging Environment**: A replica of the production environment where final testing occurs before deployment. It ensures the software works correctly in a production-like setup.

- **Production Environment**: The live environment where the software runs and is accessed by end-users. It's optimized for performance, security, and scalability.

7. **Define SRS (Software Requirements Specification):**

SRS is a document that clearly defines the functional and non-functional requirements of a software system. It serves as a blueprint for developers, guiding the design, implementation, and testing phases of the SDLC. An SRS typically includes a description of the system's purpose, functional requirements (features and capabilities), non-functional requirements (performance, security, usability), and constraints.

8. **What do you know about a feasibility study?**

A feasibility study assesses the practicality and viability of a proposed project or system. It evaluates various aspects such as technical, economic, operational, and scheduling feasibility to determine whether the project should proceed. Key factors considered include cost, resource availability, technological capabilities, legal and regulatory requirements, and potential risks. The outcome of a feasibility study

helps stakeholders make informed decisions about whether to invest in the project.

9. **Differentiate between JAD (Joint Application Development) and RAD (Rapid Application Development) model:**

- **JAD (Joint Application Development)**:

JAD is a structured workshop-based technique involving stakeholders, users, and developers to gather requirements and create a system design. It emphasizes collaborative decision-making and accelerates the requirements gathering process.

- **RAD (Rapid Application Development)**:

RAD is an iterative development approach that prioritizes rapid prototyping and quick feedback over extensive planning. It focuses on delivering software quickly by using reusable components and involves close interaction between developers and users throughout the development cycle.

10. **What is the use of JAD session?**

JAD sessions are used to:

- **Gather Requirements**: Bring together stakeholders and users to discuss and define system requirements in detail.

- **Facilitate Communication**: Encourage open communication and collaboration between stakeholders, users, and developers.

- **Reach Consensus**: Resolve conflicts, clarify ambiguities, and reach consensus on project requirements and goals.

- **Accelerate Decision-Making**: Make decisions quickly regarding system functionalities, design elements, and priorities.

11. **How is the design phase important?**

The design phase in SDLC is crucial because it:

- **Translates Requirements**: Converts system requirements into a detailed blueprint or technical design that developers can use to build the software.

- **Guides Development**: Provides a structured approach to software development by defining system architecture, data structures, interfaces, and algorithms.

- **Reduces Risks**: Identifies potential issues early in the process, allowing for adjustments before significant resources are committed.

- **Ensures Quality**: Focuses on creating a robust, scalable, and maintainable system architecture that meets both functional and non-functional requirements.

- **Facilitates Communication**: Helps stakeholders, developers, and testers understand the system's structure and functionality, ensuring everyone is aligned on the project goals.