EXPERIMENT NO.:01

Date of Performance:

Date of Submission:

Aim: Application of at least two traditional process models.

Software Used: Ms Word

Theory: Software Processes is a coherent set of activities for specifying, designing, implementing and testing software systems. A software process model is an abstract representation of a process that presents a description of a process from some particular perspective. There are many different software processes but all involve:

- Specification – defining what the system should do;
- Design and implementation – defining the organization of the system and implementing the system;
- Validation – checking that it does what the customer wants;
- Evolution – changing the system in response to changing customer needs.

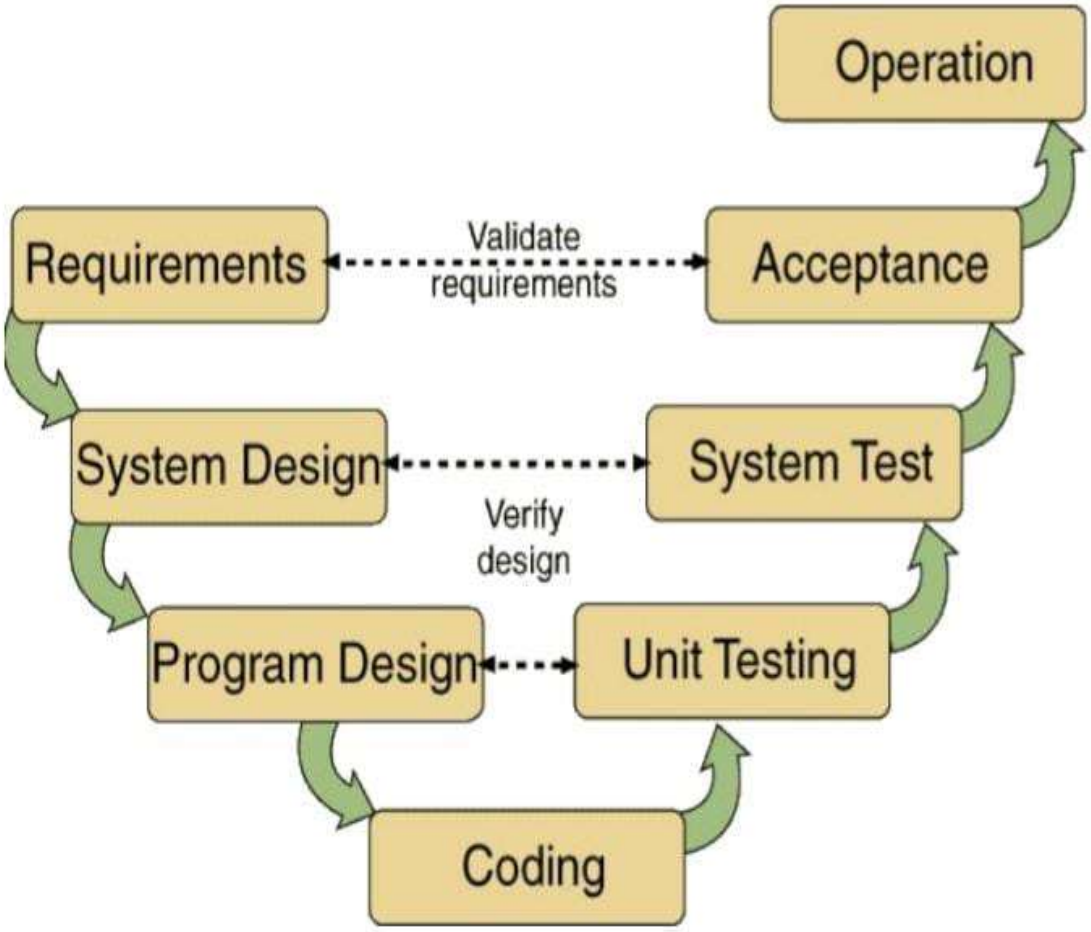
Types of Software Process Model

Software processes, methodologies and frameworks range from specific prescriptive steps that can be used directly by an organization in day-to-day work, to flexible frameworks that an organization uses to generate a custom set of steps tailored to the needs of a specific project or group. In some cases a "sponsor" or "maintenance" organization distributes an official set of documents that describe the process.

Software Process and Software Development Lifecycle Model

One of the basic notions of the software development process is SDLC models which stands for Software Development Life Cycle models. There are many development life cycle models that have been developed in order to achieve different required objectives. The models specify the various stages of the process and the order in which they are carried out. The most used, popular and important SDLC models are given below:

- Waterfall model
- V model
- Incremental model
- RAD model
- Agile model
- Iterative model
- Spiral model
- Prototype model



V-Model

The V-Model, or Verification and Validation Model, is a structured approach to software development that emphasizes parallel development and testing phases. The model is depicted as a "V," with the left side representing development stages and the right side representing corresponding testing phases.

Development Phases (Left Side of the V)

1. **Requirements Analysis:**
    - **Purpose:** Define and document what the system should achieve.
    - **Activities:** Collect and document both functional and non-functional requirements.
    - **Output:** Requirement Specification Document.
2. **System Design:**
    - **Purpose:** Develop the overall system architecture.
    - **Activities:** Create high-level design specifying major components and their interactions.
    - **Output:** System Design Document.
3. **Architecture Design:**
    - **Purpose:** Detail the design for each component.

    - **Activities:** Define detailed specifications for individual modules and their interactions.
    - **Output:** Detailed Design Document.
4. **Implementation:**
    - **Purpose:** Convert designs into code.
    - **Activities:** Write and test code modules individually (unit testing).
    - **Output:** Source code and unit test results.

Testing Phases (Right Side of the V)

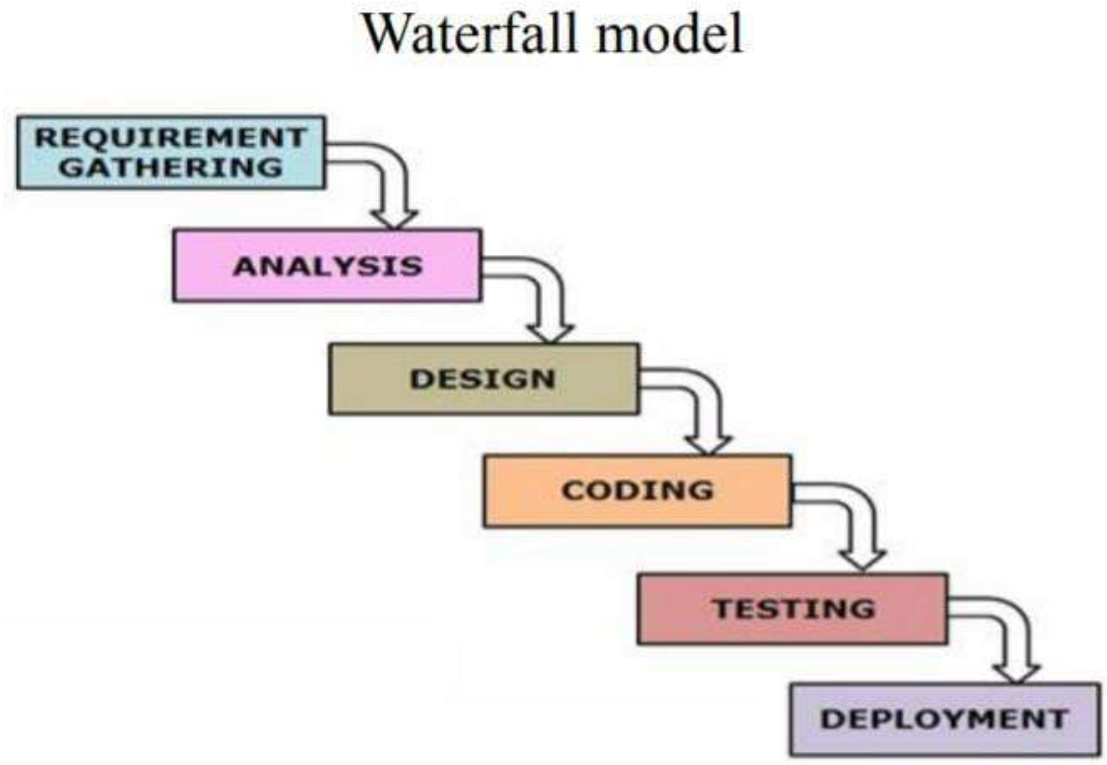
1. **Unit Testing:**
    - **Purpose:** Validate individual components.
    - **Activities:** Test code modules against design specifications.
    - **Output:** Unit Test Results.
2. **Integration Testing:**
    - **Purpose:** Ensure components work together.
    - **Activities:** Test interactions between integrated modules.
    - **Output:** Integration Test Results.
3. **System Testing:**
    - **Purpose:** Verify the complete system.
    - **Activities:** Perform end-to-end testing to ensure the system meets all requirements.
    - **Output:** System Test Results.
4. **Acceptance Testing:**
    - **Purpose:** Confirm the system meets user needs.
    - **Activities:** Conduct tests based on user scenarios.
    - **Output:** Acceptance Test Results.

Key Concepts

- **Verification:** Ensures the system is built correctly according to specifications (development).
- **Validation:** Ensures the system meets user needs (testing).

The V-Model is ideal for projects with stable requirements, providing a clear, structured approach to development and testing. However, it can be inflexible to changes and may delay defect detection until later phases.

The V-model is useful in a product price comparison project by providing a structured approach to development and validation. It ensures that each stage of the project, from requirements gathering to final testing, is meticulously planned and verified. By emphasizing validation and verification at each phase, the V-model helps maintain accuracy in price comparisons and ensures that the final product meets user expectations and requirements. This structured approach minimizes errors and improves overall project reliability.



The Waterfall Model is a traditional software development methodology characterized by a linear and sequential approach. Each phase of the project must be completed before the next phase begins, creating a "waterfall" effect. Here's a concise overview of how it works:

1. Phases of the Waterfall Model

a. Requirements Gathering and Analysis:

- **Purpose:** Understand and document what the software needs to do.
- **Activities:** Collect and analyze requirements from stakeholders. Document functional and non-functional requirements.
- **Output:** Requirement Specification Document.

b. System Design:

- **Purpose:** Define the system architecture and design.
- **Activities:** Create high-level and detailed design documents, outlining system architecture, data structures, and interfaces.
- **Output:** System Design Document.

c. Implementation:

- **Purpose:** Develop the actual software.

- **Activities:** Write code according to the design specifications. This phase involves coding and unit testing of individual components.
- **Output:** Source code and unit test results.

d. Integration and Testing:

- **Purpose:** Ensure that all components work together and meet requirements.
- **Activities:** Integrate various components and perform system testing, including functional, performance, and security tests.
- **Output:** Test reports and defect logs.

e. Deployment:

- **Purpose:** Release the software to users.
- **Activities:** Deploy the software in a production environment. Conduct user training and prepare for initial support.
- **Output:** Deployed software and user documentation.

f. Maintenance:

- **Purpose:** Address any issues or updates needed post-deployment.
- **Activities:** Fix defects, apply updates, and make enhancements based on user feedback.
- **Output:** Updated software and maintenance reports.

Key Concepts

- **Sequential Process:** Each phase must be completed before the next begins, with limited overlap.
- **Documentation:** Extensive documentation is produced at each phase, providing a clear project blueprint.

Advantages and Disadvantages

Advantages:

- **Clear Structure:** Well-defined stages and milestones.
- **Documentation:** Detailed documentation helps in managing and tracking progress.

Disadvantages:

- **Inflexibility:** Difficult to accommodate changes once a phase is completed.
- **Late Testing:** Issues are identified only after the implementation phase, which can be costly to fix.

The Waterfall Model is best suited for projects with well-defined requirements and little expected change.

Conclusion: Applying traditional process models like Waterfall and V-Model demonstrates their effectiveness in providing structured development and thorough validation, each suited to different project needs**.**

Sign and Remark:

R1	R2	R3	Total Marks	Signature
(5)	(5)	(5)	(15)	

EXPERIMENT NO.:02