Assignment 1

Assignment 1: SDLC Overview - Create a one-page infographic that outlines the SDLC phases (Requirements, Design, Implementation, Testing, Deployment), highlighting the importance of each phase and how they interconnect.

Waterfall Model

Advantages:

- Simple and easy to understand.
- Phases are clearly defined.
- Works well for smaller projects with well-defined requirements.

Disadvantages:

- Inflexible to changes after the project has started.
- Difficult to go back to any stage once it's completed.
- Not suitable for complex and evolving projects.

Applicability:

- Best suited for projects with clear, fixed requirements and low complexity, such as building construction or manufacturing processes.

Agile Model

Advantages:

- Highly flexible and adaptable to changes.
- Continuous customer feedback and involvement.
- Promotes iterative development and frequent releases.

Disadvantages:

- Requires strong collaboration and communication.
- Can lead to scope creep due to continuous changes.
- Less predictable in terms of cost and time.

Applicability:

- Ideal for projects with dynamic requirements and a need for frequent updates, such as software development and IT services.

Spiral Model

Advantages:

- Combines elements of both iterative and waterfall models.
- Focuses on risk analysis and mitigation.
- Suitable for large, complex projects.

Disadvantages:

- Can be expensive and time-consuming.
- Requires expertise in risk assessment.
- Complex to manage and implement.

Applicability:

- Suitable for high-risk projects with complex requirements, such as defense systems or large-scale software applications.

V-Model

Advantages:

- Emphasizes verification and validation.
- Each development phase has a corresponding testing phase.
- Clear and structured approach.

Disadvantages:

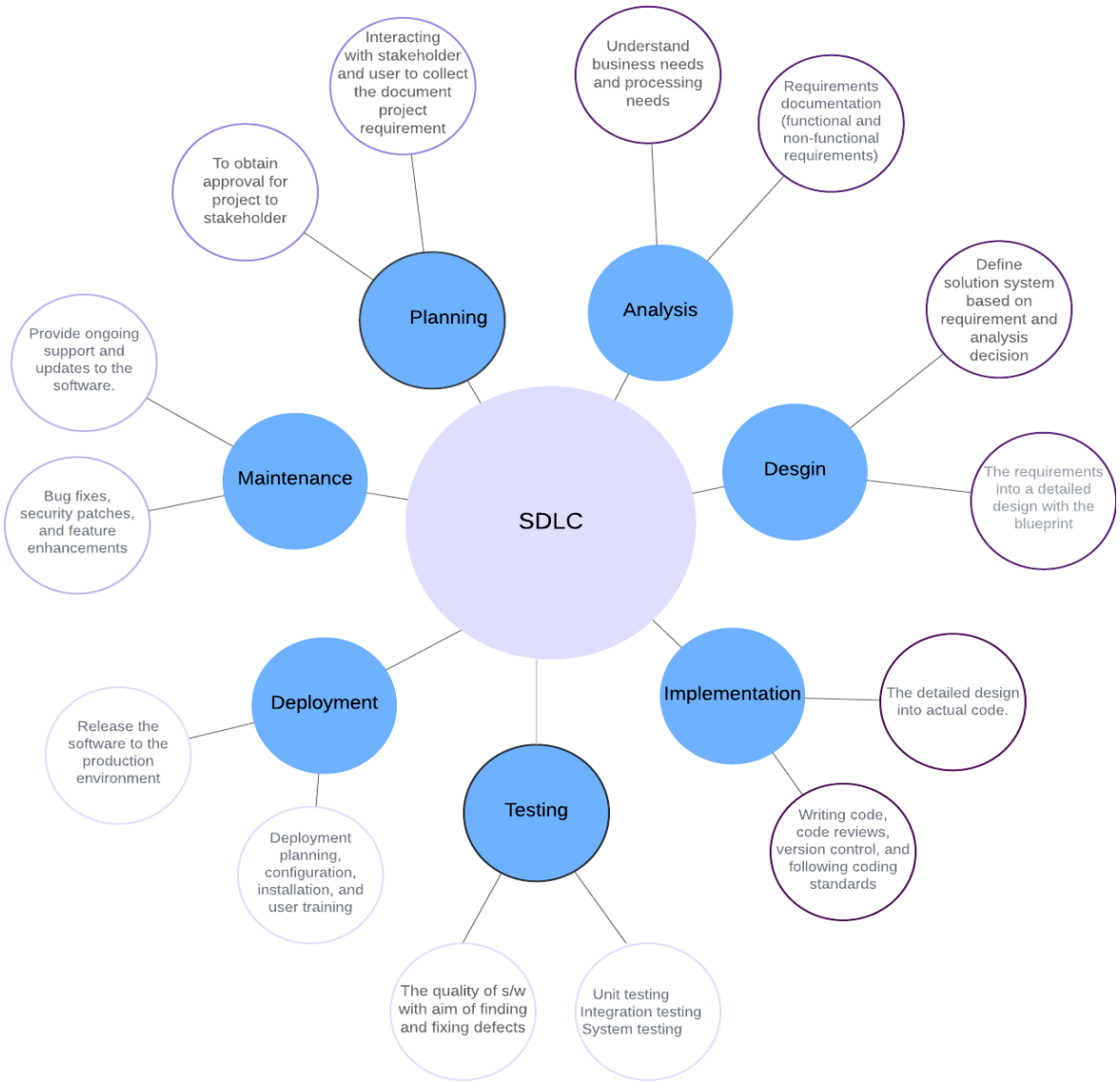
- Inflexible to changes during the development process.
- Can be costly and time-consuming.
- Not suitable for projects with unclear requirements.

Applicability:

- Best for projects where requirements are well-defined and validation is critical, such as medical device development and automotive software.

Conclusion

Each SDLC model has its own strengths and weaknesses, making them suitable for different types of engineering projects. The Waterfall model is ideal for simple, well-defined projects, while the Agile model excels in dynamic and evolving environments. The Spiral model is best for large, high-risk projects, and the V-Model is suited for projects requiring rigorous validation and verification. Selecting the appropriate model depends on the project's specific requirements, complexity, and risk factors.


SDLC
Planning
Interacting with stakeholder and user to collect the document project requirement
To obtain approval for project to stakeholder
Analysis
Understand business needs and processing needs
Requirements documentation (functional and non-functional requirements)
Desgin
Define solution system based on requirement and analysis decision
The requirements into a detailed design with the blueprint
Implementation
The detailed design into actual code.
Writing code, code reviews, version control, and following coding standards
Testing
The quality of s/w with aim of finding and fixing defects
Unit testing
Integration testing
System testing
Deployment
Release the software to the production environment
Deployment planning, configuration, installation, and user training
Maintenance
Provide ongoing support and updates to the software.
Bug fixes, security patches, and feature enhancements

Assignment 2: Case Study on the Implementation of SDLC Phases in a Real-World Engineering Project

Case Study: Implementation of a Smart Home Automation System

Project Overview: The project involves developing a Smart Home Automation System that integrates various home appliances and devices into a single network, enabling remote control and automation. The project follows the Software Development Life Cycle (SDLC) phases: Requirement Gathering, Design, Implementation, Testing, Deployment, and Maintenance.

Requirement Gathering:

- **Activities:**
  - Conducted meetings with stakeholders to understand their needs.
  - Collected data through surveys and questionnaires.
  - Defined functional and non-functional requirements.
- **Outcome:**
  - A detailed requirement specification document.
  - Identification of key features such as remote control, energy monitoring, and security alerts.

Design:

- **Activities:**
  - Created architectural designs and system models.
  - Developed wireframes and UI/UX designs.
  - Defined hardware and software interfaces.
- **Outcome:**
  - Design specifications document.
  - Prototypes of user interfaces and system architecture.

Implementation:

- **Activities:**
  - Coding and integration of software modules.
  - Hardware setup and integration.
  - Development of mobile and web applications.
- **Outcome:**
  - A working prototype of the Smart Home Automation System.
  - Code repositories and documentation.

Testing:

- **Activities:**
  - Conducted unit, integration, and system testing.
  - Performed user acceptance testing (UAT) with selected stakeholders.

    - Identified and fixed bugs and issues.
- **Outcome:**
    - Test reports and bug logs.
    - A stable and reliable system ready for deployment.

Deployment:

- **Activities:**
    - Deployed the system in real-world home environments.
    - Provided training to users and stakeholders.
    - Set up continuous monitoring and support.
- **Outcome:**
    - Successful installation and configuration of the system.
    - User manuals and training materials.

Maintenance:

- **Activities:**
    - Provided ongoing technical support and updates.
    - Collected user feedback for continuous improvement.
    - Managed system upgrades and scalability.
- **Outcome:**
    - Regular maintenance schedules.
    - Improved system performance and user satisfaction.

Conclusion: The implementation of SDLC phases in this project ensured a systematic and structured approach, leading to a successful deployment of the Smart Home Automation System. Each phase contributed significantly to the project's outcomes, from capturing accurate requirements to maintaining the system post-deployment.