Software Development life cycle (SDLC)

PHASES OF SDLC

STEP 01

Requirements

- Gathering business and system requirements from stakeholders.
- Ensures that the final product meets the needs and expectations of users.

STEP 02

Design

- Creating architecture and design specifications for the system.
- Establishes the blueprint for the development process.
- Translates requirements into a structured plan for implementation.

STEP 03

Implementation

- Actual coding and building of the system according to design specifications.
- Turns designs and plans into a functioning software product.
- Relies on the design specifications and sets the stage for testing.

STEP 04

Testing

- Verifying that the system works as intended and is free of defects
- Ensures the quality and reliability of the software.
- Feedback loop to implementation if issues are found; necessary before deployment.

STEP 05

Deployment

- Releasing the final product to users and making it operational.
- Makes the software available for use and ensures it works in the real environment.
- Concludes the development process and begins the maintenance phase.

STEP 06

Maintenance

Monitored the project performance, applied updates, and addressed user feedback.

Assignment-2:

Case Study: Using SDLC on an e-commerce website

Introduction:
This case study explores how SDLC can be used to create an e-commerce website for small businesses and offer customers a unique online shopping experience.

Phases of SDLC:

1. Requirements:
Action: Contact business owners to write requirements for features such as product listings, shopping carts, and payment methods.
Required: Ensure the website meets business and customer needs.
Result: Feedback file created.

2. Design:
Action: Create wireframes and mockups that describe the website layout and user interface.
Required: Provide clear instructions for developers.
Result: Create a document design process and user interface model.

3. Implementation:
Action: Developers create websites using a content management system (CMS), including any necessary plugins and rules.
Required: Designs converted to working websites.
Result: Submit work to e-commerce site.

4. Testing:
Action: Perform functional testing, usability testing, and security testing.
Required: Ensure the website is user-friendly, secure and error-free.
Result: Issues identified and resolved, improving site reliability and security.

5. Deployment:
Action: Phased launch site that begins with discounted sales for a select group of users.
Required: Make changes to the address with minimal disruption.
Result: Public website completed.

6. Maintenance:
Action: Monitor site performance, request updates, and respond to user feedback.
Required: Keep your website functional, up-to-date and secure.
Result: Increase user satisfaction and site performance.

SDLC ensures ecommerce websites meet business goals, deliver a great user experience, and maintain consistent performance and security. Each stage is important for the success and operation of the project.

Assignment 3: Comparison of SDLC Models for Engineering Projects

This assignment compares four SDLC models: Waterfall, Agile, Spiral, and V-Model. Each model's advantages, disadvantages, and applicability in various engineering contexts are highlighted.

---

1. Waterfall Model

Overview:

The Waterfall model is a linear and sequential approach where each phase must be completed before moving to the next. The phases include Requirements, Design, Implementation, Testing, Deployment, and Maintenance.

Advantages:

- **Simplicity**: Easy to understand and use.
- **Structured**: Clear milestones and deadlines.
- **Documentation**: Comprehensive documentation at each phase.

Disadvantages:

- **Inflexibility**: Difficult to accommodate changes once a phase is completed.
- **Late Testing**: Bugs and issues are only discovered during the testing phase, potentially leading to costly fixes.
- **Poor Adaptability**: Not suitable for projects with uncertain or evolving requirements.

Suitability:

- **Best for**: Projects with well-defined requirements, such as civil engineering projects or regulatory-driven software.
- **Not suitable for**: Projects requiring iterative development or frequent changes.

---

2. Agile Model

Overview:

Agile is an iterative and incremental approach that emphasizes flexibility, customer feedback, and rapid delivery. Development is divided into small, manageable iterations or sprints.

Advantages:

- **Flexibility**: Can accommodate changes at any stage of the development process.
- **Customer Involvement**: Regular feedback from stakeholders ensures the product meets their needs.
- **Early Detection of Issues**: Continuous testing and integration help identify and fix problems early.

Disadvantages:

- **Documentation**: Less emphasis on documentation can lead to issues in complex projects.
- **Scope Creep**: Frequent changes can lead to project scope creep.
- **Team Dependency**: Success heavily depends on the team's skill and collaboration.

Suitability:

- **Best for**: Projects with evolving requirements, such as software development and IT projects.
- **Not suitable for**: Projects requiring extensive documentation and well-defined requirements from the outset.

---

3. Spiral Model

Overview:

The Spiral model combines iterative development with systematic aspects of the Waterfall model. It focuses on risk assessment and reduction at each iteration or spiral.

Advantages:

- **Risk Management**: Emphasizes identifying and mitigating risks early in the project.
- **Flexibility**: Allows for iterative development and refinement.
- **Customer Feedback**: Regular customer feedback ensures alignment with user needs.

Disadvantages:

- **Complexity**: Managing spirals can be complex and requires expertise.
- **Cost**: Can be more expensive due to extensive risk analysis and management.
- **Time-Consuming**: Iterative approach can lead to longer development times.

Suitability:

- **Best for**: Large, complex projects with high-risk factors, such as aerospace and defense engineering.
- **Not suitable for**: Small projects with limited budgets and straightforward requirements.

---

4. V-Model

Overview:

The V-Model, also known as the Verification and Validation model, is an extension of the Waterfall model. Each development phase is paired with a corresponding testing phase, forming a V shape.

Advantages:

- **Quality Assurance**: Emphasizes verification and validation, leading to high-quality outputs.
- **Structured**: Clear stages and milestones with corresponding testing phases.
- **Early Detection of Defects**: Defects are identified and fixed early in the development process.

Disadvantages:

- **Inflexibility**: Similar to the Waterfall model, it is difficult to accommodate changes once a phase is completed.
- **Costly**: High cost of repeated testing and validation activities.
- **Documentation-Heavy**: Requires extensive documentation, which can be time-consuming.

Suitability:

- **Best for**: Projects with well-defined requirements that prioritize quality, such as medical device software and safety-critical systems.
- **Not suitable for**: Projects with rapidly changing requirements or where iterative development is needed.

Conclusion

Each SDLC model has its strengths and weaknesses, making them suitable for different types of engineering projects. The Waterfall model is ideal for projects with clear, unchanging requirements, while Agile is best for projects that benefit from flexibility and ongoing customer feedback. The Spiral model is advantageous for complex projects with significant risk factors, and the V-Model is suitable for projects where quality assurance and validation are paramount.

Choosing the right SDLC model depends on the specific needs and context of the engineering project, including factors like project size, complexity, budget, timeline, and risk.