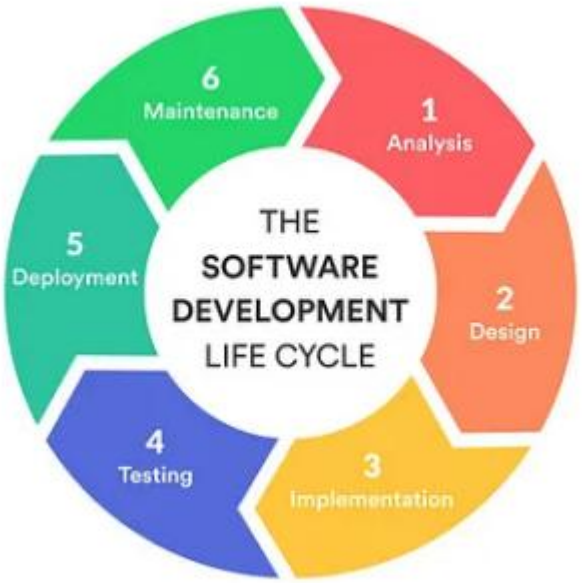
HARISH RONGALA
harishrongala123@gmail.com

Software Development Life Cycle and Agile Principles (part 1)

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



1. Requirements and Analysis:
Requirement Analysis is the most important and necessary stage in SDLC.
The senior members of the team perform it with inputs from all the stakeholders and domain experts or SMEs in the industry.

Planning for the quality assurance requirements and identifications of the risks associated with the projects is also done at this stage

2. Design:
The next phase is about to bring down all the knowledge of requirements, analysis, and design of the software project. This phase is the product of the last two, like inputs from the customer and requirement gathering.

3. Implementation:
In this phase of SDLC, the actual development begins, and the programming is built. The implementation of design begins concerning writing code. Developers have to follow the coding guidelines described by their management and programming tools like compilers, interpreters, debuggers, etc. are used to develop and implement the code.

4. Testing:
After the code is generated, it is tested against the requirements to make sure that the products are solving the needs addressed and gathered during the requirements stage.
During this stage, unit testing, integration testing, system testing, acceptance testing are done.

5. Deployment:
Once the software is certified, and no bugs or errors are stated, then it is deployed.
Then based on the assessment, the software may be released as it is or with suggested enhancement in the object segment.
After the software is deployed, then its maintenance begins.

6. Maintenance
Once when the client starts using the developed systems, then the real issues come up and requirements to be solved from time to time.
This procedure where the care is taken for the developed product is known as maintenance.

Assignment 2: Develop a case study analyzing the implementation of SDLC phases in a real-world engineering project. Evaluate how Requirement Gathering, Design, Implementation, Testing, Deployment, and Maintenance contribute to project outcomes.

Every development company will follow some standard models to implement a software. One of the popular model is called SDLC. It contains different phases. In each phase different people will be involved and perform different activities. Every phase will have some input and corresponding outputs.

The below are the phases in SDLC model.

Planning and Analysis
Design
Development
Testing
Deployment and Maintenance

Planning and Analysis: Planning and Analysis is the most important and fundamental stage in SDLC. In this stage business analyst and project manager set up a meeting with customer and gather information about the product. Once the requirement is clearly defined, a SRS (Software Requirement Specification) document will be created. This document should be thoroughly understood by the developers and also should be reviewed by the customer for future reference.

Design: Once planning and analyzing is done, the next step is, SRS document is used as an input and developer use it to design high level architecture.

Development: In this stage the actual development will be started. The software design is translated to code. Developers must follow some coding guide lines defined by the company.

Testing: Testing starts once the coding is complete and the modules are released for testing. In this phase, the developed software is tested thoroughly and any defects found are assigned to developers to get them fixed.

Deployment and Maintenance: After testing is done, code will be deployed to production environment and product will be available to end-user.

Real life example: **Develop a e-commerce platform**

Planning and Analysis: In this phase, business analyst and program manager will gather requirements from business stake holders about how e-commerce should work and what features it should contain. They will create detailed SRS (Software Requirement Specification) documents. Business analysts will get review SRS document with business stake holders and will get sign off on it.

Design: SRS documents will be sent to developers. Developers will go through the documents and will understand requirements. Designers will be design web pages. Developers will prepare high level system architecture.

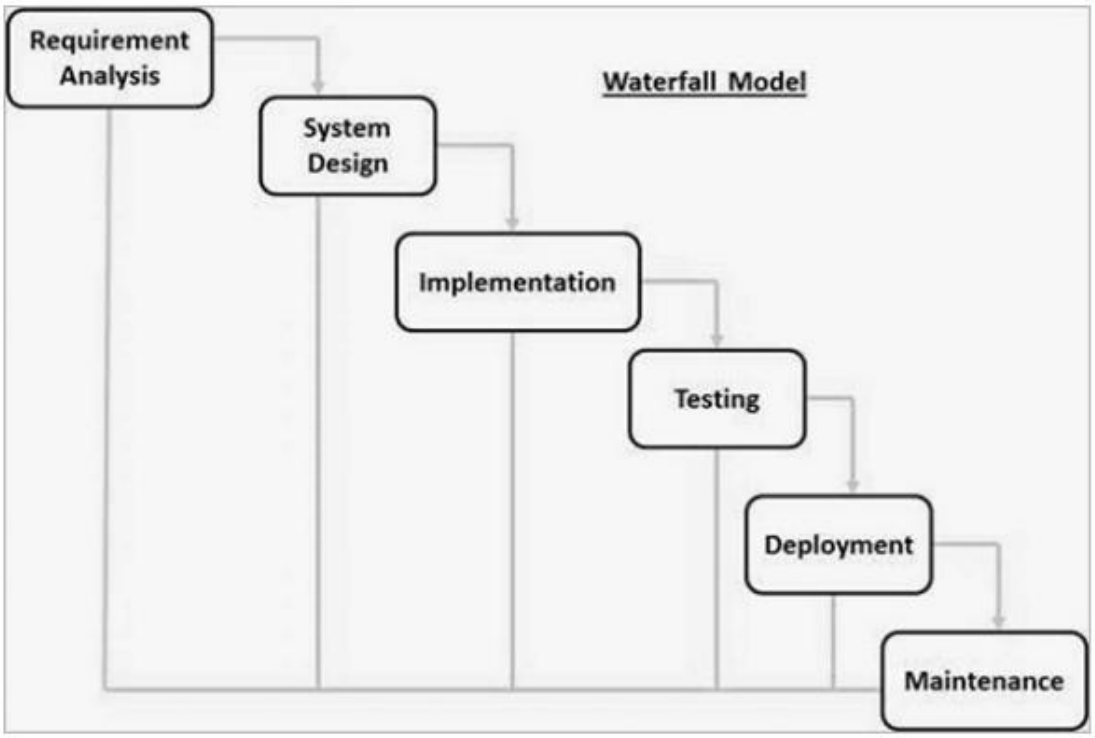
Development: Development will do coding in this phase. They will develop the web pages, API’s required to implement the functionality.

Testing: Testing will perform end to end functionality testing. They will make sure e-commerce platform working fine without any bugs.

Deployment and Maintenance: Code will be deployed and will be available to customer. Customer can use the e-commerce after this deployment. Same process will be applied for developing any new features.

Assignment 3: Research and compare SDLC models suitable for engineering projects. Present findings on Waterfall, Agile, Spiral, and V-Model approaches, emphasizing their advantages, disadvantages, and applicability in different engineering contexts.

Waterfall Model:
Waterfall approach was first SDLC Model to be used widely in Software Engineering to ensure success of the project. In "The Waterfall" approach, the whole process of software development is divided into separate phases. In this Waterfall model, typically, the outcome of one phase acts as the input for the next phase sequentially.



The sequential phases in Waterfall model are :-

Requirement Gathering and analysis : All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.

System Design : The requirement specifications from first phase are studied in this phase and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.

Implementation : With inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.

Integration and Testing : All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.

Deployment of system : Once the functional and non-functional testing is done; the product is deployed in the customer environment or released into the market.

Maintenance : There are some issues which come up in the client environment. To fix those issues, patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.

Waterfall Model - Application

Every software developed is different and requires a suitable SDLC approach to be followed based on the internal and external factors. Some situations where the use of Waterfall model is most appropriate are −

Requirements are very well documented, clear and fixed.

Product definition is stable.

Technology is understood and is not dynamic.

There are no ambiguous requirements.

Ample resources with required expertise are available to support the product.

The project is short.

Advantages of Waterfall Model:

The advantages of waterfall development are that it allows for departmentalization and control. A schedule can be set with deadlines for each

stage of development and a product can proceed through the development process model phases one by one.

Development moves from concept, through design, implementation, testing, installation, troubleshooting, and ends up at operation and maintenance. Each phase of development proceeds in strict order.

Some of the major advantages of the Waterfall Model are as follows −

1. Simple and easy to understand and use

2. Easy to manage due to the rigidity of the model. Each phase has specific deliverable and a review process.

3. Phases are processed and completed one at a time.

4. Works well for smaller projects where requirements are very well understood.

5. Clearly defined stages.

6. Well understood milestones.

7. Easy to arrange tasks.

8. Process and results are well documented.

Disadvantages of Waterfall Model

The disadvantage of waterfall development is that it does not allow much reflection or revision. Once an application is in the testing stage, it is very difficult to go back and change something that was not well-documented or thought upon in the concept stage.

The major disadvantages of the Waterfall Model are as follows −

1. No working software is produced until late during the life cycle.

2. High amounts of risk and uncertainty.

3. Not a good model for complex and object-oriented projects.

4. Poor model for long and ongoing projects.

5. Not suitable for the projects where requirements are at a moderate to high risk of changing. So, risk and uncertainty is high with this process model.

6. It is difficult to measure progress within stages.

7. Cannot accommodate changing requirements.

Agile Model:

Agile is an iterative and flexible approach that prioritizes adaptive responses to changing requirements and customer feedback. It emphasizes collaboration, self-organization, and incremental development, with shorter development cycles known as sprints. Agile promotes delivering working software in short iterations and welcomes changing requirements even late in the development process.

Advantages:

*Corrections of functional requirements are implemented into the development process to provide the competitiveness.

*Project is divided by short and transparent iterations.

*Risks are minimized thanks to the flexible change process.

*Fast release of the first product version

Disadvantages:

*Difficulties with measuring the final cost because of permanent changes

*The team should be highly professional and client-oriented

*New requirements may conflict with the existing architecture

*With all the corrections and changes there is possibility that the project will exceed expected time.

Applicability:
Well-suited for projects with evolving requirements, high uncertainty, or rapidly changing market conditions. Widely used in software development but can be applied to various engineering contexts such as product development or research projects.

Spiral Model:
The Spiral model is a risk-driven approach that combines iterative development with risk management activities. It involves repeated cycles of prototyping, risk analysis, and refinement, allowing for progressive elaboration of requirements and design. Each cycle begins with risk assessment and ends with a prototype or incremental release, enabling early validation and risk mitigation.

Advantages:

* Life-cycle is divided into small parts, and if the risk concentration is higher,the phase can be finished earlier to address the treats.
* The development process is precisely documented yet scalable to the changes
* The scalability allows to make changes and add new functionality even at the relatively late stages

Disadvantages:
* Can be quite expensive
* The risk control demands involvement of the highly-skilled professionals
* Can be ineffective for the small projects
* Big number of the intermediate stages requires excessive documentation

Applicability:

Suitable for large-scale projects with high complexity and uncertainty, where risk management is critical. Commonly used in engineering contexts such as aerospace, defense, and critical infrastructure projects where iterative refinement and risk mitigation are necessary.

V-Model:

The V-Model is a structured approach that emphasizes the correlation between development and testing activities. It follows a phased approach where each development phase is paired with a corresponding testing phase. Requirements and design phases are followed by verification activities, while implementation and testing phases are followed by validation activities. This model ensures comprehensive test coverage and trace ability throughout the development life-cycle.

Advantages:

* Every stage of V-shaped model has strict results so it’s easy to control
* Testing and verification take place in the early stages
* Good for the small projects, where requirements are static and clear

Disadvantages:

* Lack of the flexibility
* Bad choice for the small projects
* Relatively big risks

Applicability:

Suitable for projects with clear and stable requirements, where comprehensive testing and quality assurance are paramount. Commonly used in engineering contexts such as automotive, medical devices, and regulated industries where compliance and trace ability are essential