MONISHA S

SDLC – SOFTWARE DEVELOPMENT LIFE CYCLE

The Software Development Life Cycle (SDLC) refers to a methodology with clearly defined processes for creating high-quality software.

Phases of software development:

- Requirement analysis
- Design
- Coding
- Testing
- Deployment
- Maintenance

In Requirement Analysis phase,

1. Functional Requirements
2. Non-Functional Requirements

Functional Requirements:

These are the requirements that the end user specifically demands as basic facilities that the system should offer. A functional requirement defines what a system must or must not, do.

Non-Functional Requirements:

These are basically the quality constraints that the system must satisfy according to the project contract.

They basically deal with issues like:

- Portability
- Security
- Maintainability
- Reliability
- Scalability
- Performance
- Reusability
- Flexibility

In Design phase,

High-Level Design (HLD) and Low-Level Design (LLD):

- High-Level Design:

  HLD is the general system design means it refers to the overall system design. The HLD involves system architecture, database design, a brief description of systems, services, platforms, and relationships among modules. The HLD is also known as macro-level or system design. It changes the business or client requirement into a High-Level Solution.
  Example: Our college website.

- Low-Level Design:

  The LLD stands for Low-Level Design, in which the designer will focus on the components like a User interface (UI). The Low-level design is created by the developer manager and designers. It is also known as micro-level or detailed design. The LLD can change the High-Level Solution into a detailed solution.
  Example: Each module of our college website.

SDLC Methodologies:
There are many types of SDLC models of which some includes,

- Waterfall Model
- RAD Model
- Spiral Model
- V- Model
- Incremental Model
- Agile Model
- Iterative Model
- Bigbang Model

Advantages and Disadvantages of Models:

Waterfall Model:
Waterfall model was used to develop enterprise applications like Customer Relationship Management (CRM) systems, Human Resource Management Systems (HRMS), Supply Chain Management Systems, Inventory Management Systems, Point of Sales (POS) systems for Retail chains etc.

Advantages:

- Before the next phase of development, each phase must be completed.
- Suited for smaller projects where requirements are well defined.
- Elaborate documentation is done at every phase of the software's development cycle.

Disadvantages:

- Error can be fixed only during the phase.
- It is not desirable for complex project where requirement changes frequently.
- Testing period comes quite late in the developmental process.
- Clients valuable feedback cannot be included with ongoing development phase.

RAD Model:

Rapid Application Development (RAD) is a development model that prioritizes rapid prototyping and quick feedback over long drawn-out development and testing cycles. With rapid application development, developers can make multiple iterations and updates to a software quickly without starting from scratch each time.

Advantages:

- It is useful when you have to reduce the overall project risk.
- It is adaptable and flexible to changes.
- It is easier to transfer deliverables as scripts, high-level abstractions and intermediate codes are used.
- Due to code generators and code reuse, there is a reduction of manual coding.

Disadvantages:

- Not all application is compatible with RAD.
- It can't be used for smaller projects.
- When technical risk is high, it is not suitable.
- If developers are not committed to delivering software on time, RAD projects can fail.

V-Model:
It can be used for software projects of any size, whether in commerce, the military or the public sector. The model serves as a tool to facilitate the organization and execution of development, maintenance, and advancement of diverse IT systems.

Advantages:
- Higher success chances because development of test plans early on during the life cycle.
- Verification and validation of the product in the early stages of product development.
- An apt fit for small projects with easily understandable requirements and easy to use.

Disadvantages:

- Costly and required more time, in addition to a detailed plan
- Less flexible like waterfall model.
- Requirement and test documents need to be updated if any changes had to be made amid the software development.

Spiral Model:
The best-suited example for the spiral model is MS-Excel, because MS-Excel sheet having several cells, which are the components of an excel sheet.

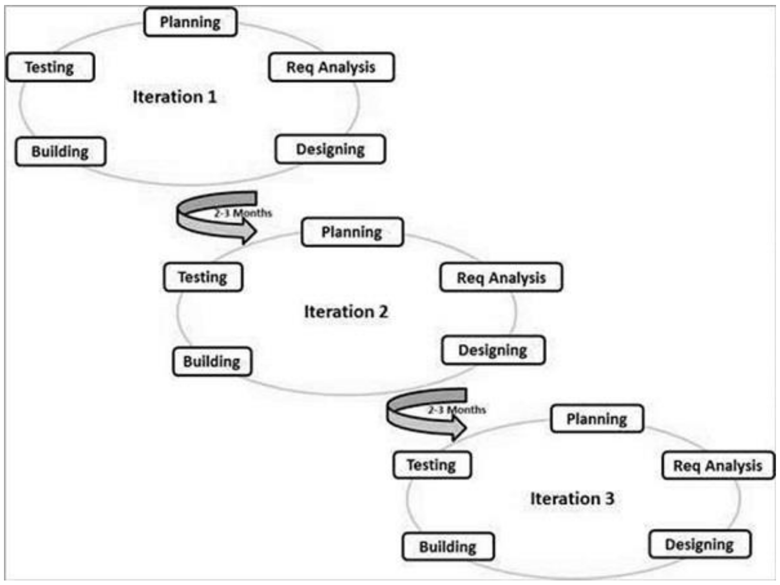
Advantages:
- Early involvement of developers.
- Manages risks and develops the system into phases.
- As the prototype build is done in small increments, cost estimation is easy.

Disadvantages:
- Much more documentation due to intermediate phases
- High cost and time to reach the final product.
- Needs special skills to evaluate the risks and assumptions.
- Highly customized limiting re-usability.

Agile Model:

Agile SDLC model is a combination of iterative and incremental process models with focus on process adaptability and customer satisfaction by rapid delivery of working software product. Agile Methods break the product into small incremental builds. These builds are provided in iterations. Each iteration typically lasts from about one to three weeks.

Graphical illustration of the Agile Model:



Types of Agile Model:

There are 5 main Agile methodologies:

- Scrum
- Kanban
- Extreme Programming (XP)
- Lean Development
- Crystal

Advantages:

- Customer satisfaction by rapid, continuous delivery of useful software.
- People and interactions are emphasized rather than process and tools. Customers, developers and testers constantly interact with each other.
- Working software is delivered frequently (weeks rather than months).
- Face-to-face conversation is the best form of communication.
- Close, daily cooperation between business people and developers.
- Continuous attention to technical excellence and good design.
- Regular adaptation to changing circumstances.

- Even late changes in requirements are welcomed.

Disadvantages:

- In case of some software deliverables, especially the large ones, it is difficult to assess the effort required at the beginning of the software development life cycle.
- There is lack of emphasis on necessary designing and documentation.
- The project can easily get taken off track if the customer representative is not clear what final outcome that they want.
- Only senior programmers are capable of taking the kind of decisions required during the development process. Hence it has no place for newbie programmers, unless combined with experienced resources.

When to use Agile model:

- When new changes are needed to be implemented. New changes can be implemented at very little cost because of the frequency of new increments that are produced.
- To implement a new feature the developers need to lose only the work of a few days, or even only hours, to roll back and implement it.
- Unlike the waterfall model in agile model very limited planning is required to get started with the project. Agile assumes that the end users needs are ever changing in a dynamic business and IT world. Changes can be discussed and features can be newly effected or removed based on feedback. This effectively gives the customer the finished system they want or need.