**Software Engineering**

Software is a program or set of programs containing instructions that provide desired functionality. And Engineering is the process of designing and building something that serves a particular purpose and finds a cost-effective solution to problems.

Software engineering is the process of designing, developing, testing, and maintaining software. It is a systematic and disciplined approach to software development that aims to create high-quality, reliable, and maintainable software. Software engineering includes a variety of techniques, tools, and methodologies, including requirements analysis, design, testing, and maintenance.

### Some key principles of software engineering include:

1. **Modularity**: Breaking the software into smaller, reusable components that can be developed and tested independently.
2. **Abstraction**: Hiding the implementation details of a component and exposing only the necessary functionality to other parts of the software.
3. **Encapsulation**: Wrapping up the data and functions of an object into a single unit, and protecting the internal state of an object from external modifications.
4. **Reusability**: Creating components that can be used in multiple projects, which can save time and resources.
5. **Maintenance**: Regularly updating and improving the software to fix bugs, add new features, and address security vulnerabilities.
6. **Testing**: Verifying that the software meets its requirements and is free of bugs.
7. **Design Patterns**: Solving recurring problems in software design by providing templates for solving them.
8. **Agile methodologies:**Using iterative and incremental development processes that focus on customer satisfaction, rapid delivery, and flexibility.
9. **Continuous Integration & Deployment:** Continuously integrating the code changes and deploying them into the production environment.

**Objectives of Software Engineering:**

1. **Maintainability –**   
   It should be feasible for the software to evolve to meet changing requirements.
2. **Efficiency –**  
   The software should not make wasteful use of computing devices such as memory, processor cycles, etc.
3. **Correctness –**   
   A software product is correct if the different requirements as specified in the SRS document have been correctly implemented.
4. **Reusability –**   
   A software product has good reusability if the different modules of the product can easily be reused to develop new products.
5. **Testability –**   
   Here software facilitates both the establishment of test criteria and the evaluation of the software with respect to those criteria.
6. **Reliability –**   
   It is an attribute of software quality. The extent to which a program can be expected to perform its desired function, over an arbitrary time period.
7. **Portability –**   
   In this case, the software can be transferred from one computer system or environment to another.
8. **Adaptability –**   
   In this case, the software allows differing system constraints and the user needs to be satisfied by making changes to the software.
9. **Interoperability**– Capability of 2 or more functional units to process data cooperatively.

# Software Development Life Cycle (SDLC)

A software life cycle model (also termed process model) is a pictorial and diagrammatic representation of the software life cycle. A life cycle model represents all the methods required to make a software product transit through its life cycle stages. It also captures the structure in which these methods are to be undertaken.

In other words, a life cycle model maps the various activities performed on a software product from its inception to retirement. Different life cycle models may plan the necessary development activities to phases in different ways. Thus, no element which life cycle model is followed, the essential activities are contained in all life cycle models though the action may be carried out in distinct orders in different life cycle models. During any life cycle stage, more than one activity may also be carried out.

## **Need of SDLC**

The development team must determine a suitable life cycle model for a particular plan and then observe to it.

Without using an exact life cycle model, the development of a software product would not be in a systematic and disciplined manner. When a team is developing a software product, there must be a clear understanding among team representative about when and what to do. Otherwise, it would point to chaos and project failure.

## **SDLC Cycle**

SDLC Cycle represents the process of developing software. SDLC framework includes the following steps:



## **The stages of SDLC are as follows:**

**Stage1: Planning and requirement 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.

**Stage2: Defining Requirements**

Once the requirement analysis is done, the next stage is to certainly represent and document the software requirements and get them accepted from the project stakeholders.

This is accomplished through "SRS"- Software Requirement Specification document which contains all the product requirements to be constructed and developed during the project life cycle.

**Stage3: Designing the Software**

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.

**Stage4: Developing the project**

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.

**Stage5: 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.

**Stage6: 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.

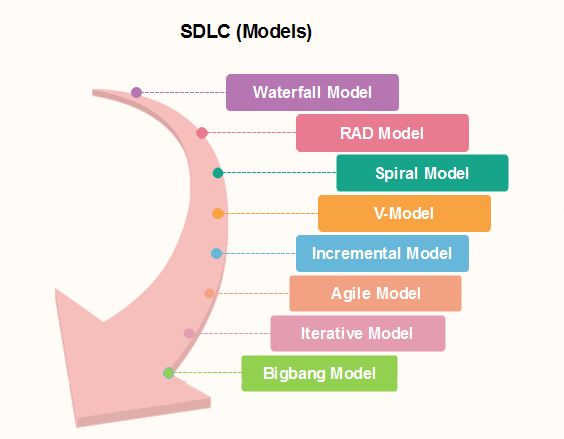
**Stage7: Maintenance**

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.

# SDLC Models

There are different software development life cycle models specify and design, which are followed during the software development phase. These models are also called "**Software Development Process Models**." Each process model follows a series of phase unique to its type to ensure success in the step of software development



# Agile Model

Agile Model is an incremental and iterative process of software development. It defines each iteration’s number, duration, and scope in advance. Every iteration is considered a short “frame” in the Agile process model, which mostly lasts from two to four weeks.

Agile Model divides tasks into time boxes to provide specific functionality for the release. Each build is incremental in terms of functionality, with the final build containing all the attributes. The division of the entire project into small parts helps minimize the project risk and the overall project delivery time. Each iteration involves a team working through a full software development life cycle including planning, requirements analysis, design, coding, and testing before a working product is demonstrated to the client.



## **Phases of Agile Model:**

Following are the phases in the Agile model are as follows:

1. Requirements gathering
2. Design the requirements
3. Construction/ iteration
4. Testing/ Quality assurance
5. Deployment
6. Feedback

**AGILE FRAMEWORKS**

An Agile framework is a specific approach to planning, managing, and executing work. Agile frameworks typically fall into two categories: Frameworks designed for teams, and frameworks designed to help organizations practice Agile at scale, across many teams.

* Agile is a **flexible** way to manage projects.
* It is a set of **values and principles** designed to handle projects with changing requirements and **deliver value** both early and frequently in a project's timeline.
* There are several different frameworks used to **apply agile principles**.
* A framework can be defined as an **approach** or set of **techniques and guidelines** used to implement agile and uphold its values.
* Think of agile as the overall philosophy, and frameworks as **tools** you use to carry out that philosophy.

## **Agile methodologies and frameworks**

There are several Agile methodologies and frameworks, each with its own pros and cons. Some are hybrids of multiple methodologies. Scrum is the most commonly used Agile methodology. Digital.ai found that 66 percent of Agile adopters used Scrum, with the next most-used methodology being ScrumBan, at 9 percent.

Popular Agile methodologies include:

* Scrum
* Kanban
* Lean
* Crystal
* Extreme Programming (XP)
* Feature-Driven Development (FDD)
* Domain-Driven Design (DDD)
* Dynamic Systems Development Method (DSDM)
* ScrumBan
* Agile-Waterfall/Hybrid Agile
* Scrum XP Hybrid

**APM Framework**

The Agile project management framework (APM) provides effective project management throughout the lifecycle of the project in an iterative and cohesive manner. Within the project scope, incorporates both traditional and modern aspects of project management.

It defines the APM framework as a series of steps that takes a project from an initial vision of the product to the final delivery of product. There are five different phases of Agile Project Management framework that occurs within a project development.

# Agile Project Management APM Framework

Agile Project Management Framework is a modern agile framework that covers the entire lifecycle of a project.  APM framework was introduced by Mr. [Jim Highsmith in the book,](http://jimhighsmith.com/books/)

**APM Framework**

The Agile project management framework (APM) provides effective project management throughout the lifecycle of the project in an iterative and cohesive manner. Within the project scope, incorporates both traditional and modern aspects of project management.

It defines the APM framework as a series of steps that takes a project from an initial vision of the product to the final delivery of product. There are five different phases of Agile Project Management framework that occurs within a project development.

## **1. Envision Phase:**

The envision phase is the initial phase of project management within an APM framework. In general, after approval of a business case, the agile key members are involved in the envision phase where they collaborate to create the compelling vision for a project. Envision phase identifies customer’s vision of the project, decides the key capabilities required in the project, set the business objectives of the project, identifies the quality objectives of the project, and identifies the right participants and stakeholders of the project and plans how the team will deliver the project.

**2. Speculate Phase:**

In the Speculate phase, the product vision into a backlog of requirements is translated, the overall approach to meet the requirements is realized and a high level release plan for the product is presented. There are two key activities in the Speculate phase:  
  
1. The team must come up with at least an initial understanding of the requirements for the project. Each feature will be further broken down into one or more “user stories” for the team to discuss and estimate. The requirements also have to be prioritized so that the team knows in what order to start working on them.

2. The second aspect is to determine a high level milestone based plan that speculates how long it would take to create those features. This planning happens at multiple levels such as release level, wave level and iteration level.

## **3. Explore Phase:**

As the name suggests, in this phase agile team members explores various alternatives to implement and fulfill the requirements of a project. In this phase, work deliveries and testing takes place. Here, the product vision needs to be transformed to a release plan and then to the respective iteration plan. The team works in an iterative manner in the explore phase that means, they take a sub-set of the product’s features or stories and accept it into a plan for an iteration. Then it will proceed to work on the development for the stories. It goes hand-in-hand with the adapt phase, wherein the team learns from the experiences of development and the feedback from the customer.

## **4. Adapt Phase:**

In the adapt phase, the agile team reviews the results of execution, the current situation, performance of the team against the plan and adapt as per the requirements. Adaptation can be changing the approach to project, changing the process, changing the environment, changing the project’s objectives and so on as per the requirements of the customers. Taking feedback, acknowledging it and adapting to the situation based on the feedback is the major work of this phase.

## **5. Close Phase:**

This is the last phase within agile project management framework. It concludes the project in an ordered manner capturing the project’s key lessons. Know more about agile processes, [agile methodologies](https://www.simplilearn.com/tutorials/agile-scrum-tutorial/what-is-agile), agile tool and techniques by registering for Simplilearn’s **PMI-ACP certification online training**..