

**Software Testing Assignment**

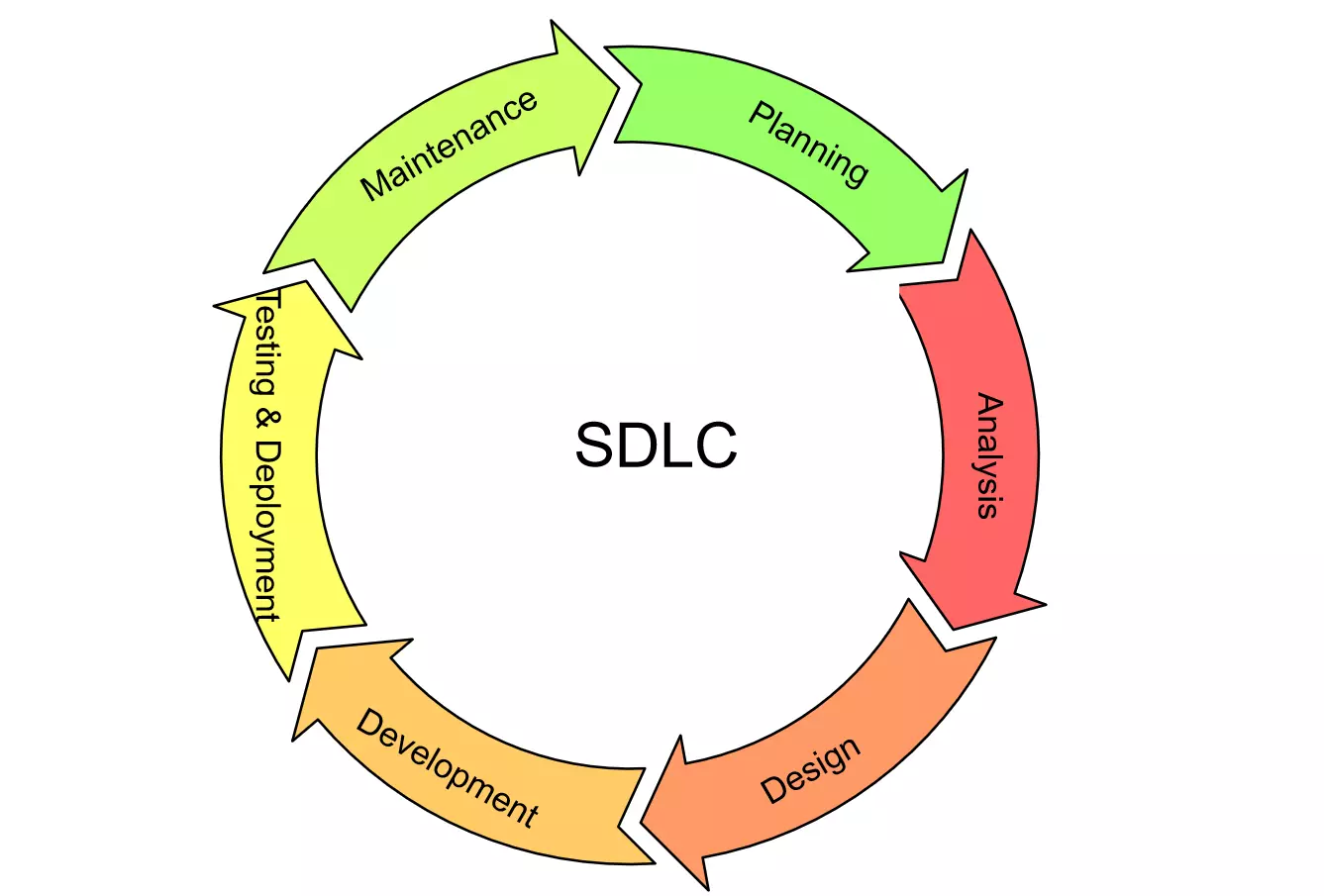
* **What is SDLC?**

**Software development life cycle (SDLC) is a structured process that is used to design, develop, and test good-quality software.**

SDLC, or software development life cycle, is a methodology that defines the entire procedure of software development step-by-step.

The goal of the SDLC life cycle model is to deliver high-quality, maintainable software that meets the user’s requirements.

SDLC in software engineering models outlines the plan for each stage so that each stage of the software development model can perform its task efficiently to deliver the software at a low cost within a given time frame that meets users’ requirements.



* **There are 6 Phases of SDLC:**

**1. Requirements gathering and analysis:**

This phase involves gathering information about the software requirements from stakeholders, such as customers, end-users, and business analysts.

**2. Design:**

In this phase, the software design is created, which includes the overall architecture of the software, data structures, and interfaces. It has two steps:

* **High-level design (HLD):** It gives the architecture of software products.
* **Low-level design (LLD):** It describes how each and every feature in the product should work and every component.

**3. Implementation or coding:**

The design is then implemented in code, usually in several iterations, and this phase is also called Development.

things you need to know about this phase:

* This is the longest phase in SDLC model.
* This phase consists of Front end + Middleware + Back-end.
* **In front-end:** Development of coding is done even SEO settings are done.
* **In Middleware:** They connect both the front end and back end.
* **In the back-end:** A database is created.

**4. Testing:**

The software is thoroughly tested to ensure that it meets the requirements and works correctly.

**5. Deployment:**

After successful testing, the software is deployed to a production environment and made available to end-users.

**6. Maintenance:** This phase includes ongoing support, bug fixes, and updates to the software.

* **Explain Phases of the Waterfall Model**

The Waterfall Model is a classical software development methodology which is a linear and sequential approach to software development that consists of several phases that must be completed in a specific order.

The Waterfall Model has six phases:

**1.** [**Requirements Gathering**](https://www.geeksforgeeks.org/requirement-gathering-challenges-and-solution-in-software-development/amp/) **and Analysis:** The first phase involves gathering requirements from stakeholders and analyzing them to understand the scope and objectives of the project.

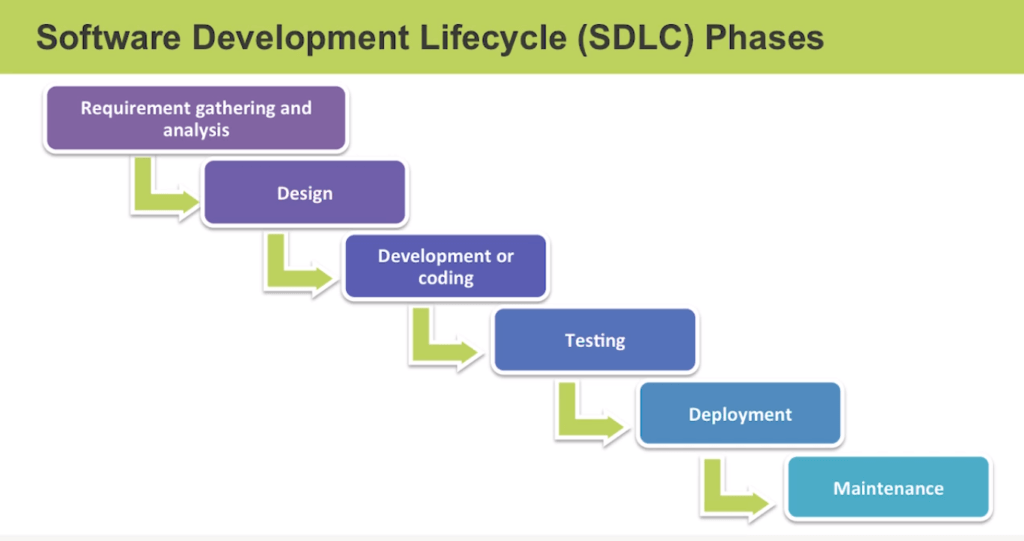
**2. Design Phase:** Once the requirements are understood, the design phase begins. This involves creating a detailed design document that outlines the software architecture, user interface, and system components.

**3. Implementation and Unit Testing:** The implementation phase involves coding the software based on the design specifications. This phase also includes unit testing to ensure that each component of the software is working as expected.

**4. Integration and System Testing:** In the testing phase, the software is tested as a whole to ensure that it meets the requirements and is free from defects.

**5. Deployment:** Once the software has been tested and approved, it is deployed to the production environment.

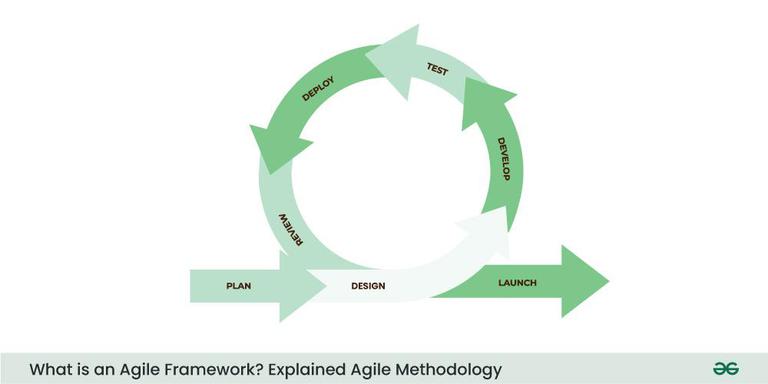
**6. Maintenance:** The final phase of the Waterfall Model is maintenance, which involves fixing any issues that arise after the software has been deployed and ensuring that it continues to meet the requirements over time.



The classical waterfall model divides the life cycle into a set of phases. This model considers that one phase can be started after the completion of the previous phase. That is the output of one phase will be the input to the next phase. Thus, the development process can be considered as a sequential flow in the waterfall. Here the phases do not overlap with each other. The different sequential phases of the classical waterfall model are shown in the figure.

## **What is Agile Methodology?**

Agile Software Development Methodology is a project management approach that allows successful and efficient execution of the project while emphasizing the improvement of a project and team collaboration.

The approach is applicable in software development for flexibility, customer satisfaction, and collaboration. It refers to the application of a set of principles that functions through an interactive and incremental approach.

The Agile Software Development Methodology emphasizes the importance of team collaboration and delivering a working product quickly to meet customer needs and expectations.

* **Agile manifesto principles**

1. Individuals and Interaction
2. Development of working software takes precedence over detailed documentation and paperwork.
3. Customer Coloration
4. Responding to change.

* **What is Software Testing?**

**Software testing** involves executing a program to identify any **error or bug** in the software product’s code.

This process considers all aspects of the software, including its reliability, scalability, portability, reusability, and usability.

The main goal of software testing is to ensure that the system and its components meet the specified requirements and work accurately in every case.

* **What is SRS?**

**Software Requirement Specification (SRS) Format** as the name suggests, is a complete specification and description of requirements of the software that need to be fulfilled for the successful development of the software system.

These requirements can be functional as well as non-functional depending upon the type of requirement.

The interaction between different customers and contractors is done because it is necessary to fully understand the needs of customers.

* **What is OOPS?**

**Object-oriented programming (OOP)** is nothing but that which allows the writing of programs with the help of certain classes and real-time objects.

We can say that this approach is very close to the real-world and its applications because the state and behavior of these classes and objects are almost the same as real-world objects.

The Pillers of Opps are: -

1) Object

2) Class

3) Encapsulation

4) Inheritance

5) Polymorphism

6) Abstraction

* **What is Class?**

A class is a user-defined data type.

It consists of data members and member functions, which can be accessed and used by creating an instance of that class.

It represents the set of properties or methods that are common to all objects of one type. A class is like a blueprint for an object.

***For Example:*** Consider the Class of Cars. There may be many cars with different names and brands but all of them will share some common properties like all of them will have 4 wheels, Speed Limit, Mileage range, etc. So here, Car is the class, and wheels, speed limits, mileage are their properties.

* **What is Object?**

It is a basic unit of Object-Oriented Programming and represents the real-life entities. An Object is an instance of a Class.

Objects can interact without having to know details of each other’s data or code, it is sufficient to know the type of message accepted and type of response returned by the objects.

Object is made of:

1) Data (Information) and

2) Function (Behavior)

***For Example:*** “Dog” is a real-life Object, which has some characteristics like color, Breed, Bark, Sleep, and Eats.

* **What is Encapsulation?**

Encapsulation is one of the fundamental concepts in object-oriented programming (OOP). It describes the idea of wrapping data and the methods that work on data within one unit.

e.g., a class in Java. This concept is often used to hide the internal state representation of an object from the outside.

* **What is** **Inheritance?**

Inheritance is the ability of one class to inherit capabilities or properties of another class, called the parent class. When we write a class, we inherit properties from other classes. So, when we create a class, we do not need to write all the properties and functions again and again, as these can be inherited from another class that possesses it. Inheritance allows the user to reuse the code whenever possible and reduce its redundancy.

* **What is Polymorphism?**

Polymorphism is the ability of data to be processed in more than one form. It allows the performance of the same task in various ways. It consists of method overloading and method overriding, i.e., writing the method once and performing a number of tasks using the same method name.

* **What Are the Phases of Spiral Model?**

The Spiral Model is a risk-driven model, meaning that the focus is on managing risk through multiple iterations of the software development process. It consists of the following phases:

### **1. Planning**

The first phase of the Spiral Model is the planning phase, where the scope of the project is determined and a plan is created for the next iteration of the spiral.

### **2. Risk Analysis**

In the risk analysis phase, the risks associated with the project are identified and evaluated.

### **3. Engineering**

In the engineering phase, the software is developed based on the requirements gathered in the previous iteration.

### **4. Evaluation**

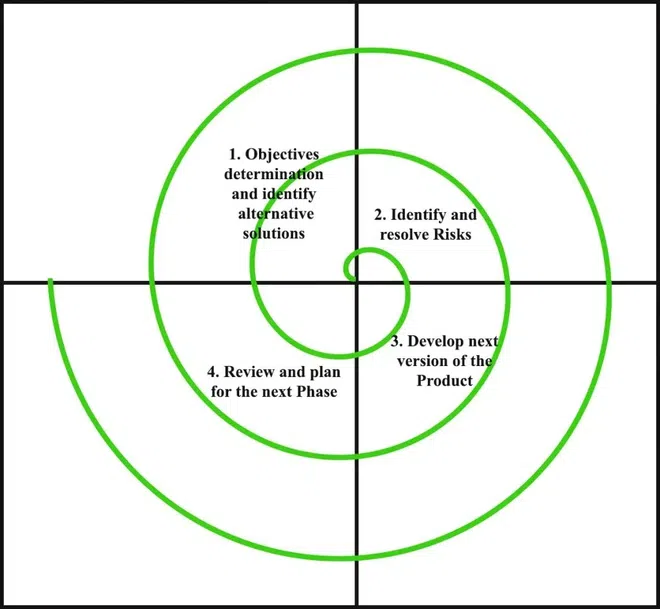
In the evaluation phase, the software is evaluated to determine if it meets the customer’s requirements and if it is of high quality.

### **5. Planning**

The next iteration of the spiral begins with a new planning phase, based on the results of the evaluation.

The Spiral Model is often used for complex and large software development projects, as it allows for a more flexible and adaptable approach to [software development](https://www.geeksforgeeks.org/software-development/?ref=lbp). It is also well-suited to projects with significant uncertainty or high levels of risk.

The Radius of the spiral at any point represents the expenses(cost) of the project so far, and the angular dimension represents the progress made so far in the current phase.



Each phase of the Spiral Model is divided into four quadrants as shown in the above figure. The functions of these four quadrants are discussed below:

1. **Objectives determination and identify alternative solutions:** Requirements are gathered from the customers and the objectives are identified, elaborated, and analyzed at the start of every phase. Then alternative solutions possible for the phase are proposed in this quadrant.
2. **Identify and resolve Risks:** During the second quadrant, all the possible solutions are evaluated to select the best possible solution. Then the risks associated with that solution are identified and the risks are resolved using the best possible strategy. At the end of this quadrant, the Prototype is built for the best possible solution.
3. **Develop the next version of the Product:** During the third quadrant, the identified features are developed and verified through testing. At the end of the third quadrant, the next version of the software is available.
4. **Review and plan for the next Phase:** In the fourth quadrant, the Customers evaluate the so-far developed version of the software. In the end, planning for the next phase is started.

## **Advantages of the Spiral Model**

Below are some advantages of the Spiral Model.

1. **Risk Handling:** The projects with many unknown risks that occur as the development proceeds, in that case, Spiral Model is the best development model to follow due to the risk analysis and risk handling at every phase.
2. **Good for large projects:** It is recommended to use the Spiral Model in large and complex projects.
3. **Flexibility in Requirements:** Change requests in the Requirements at a later phase can be incorporated accurately by using this model.
4. **Customer Satisfaction:** Customers can see the development of the product at the early phase of software development and thus they get used to the system by using it before completion of the total product.
5. **Iterative and Incremental Approach:** The Spiral Model provides an iterative and incremental approach to software development, allowing for flexibility and adaptability in response to changing requirements or unexpected events.
6. **Emphasis on Risk Management:** The Spiral Model places a strong emphasis on risk management, which helps to minimize the impact of uncertainty and risk on the software development process.
7. **Improved Communication:** The Spiral Model provides for regular evaluations and reviews, which can improve communication between the customer and the development team.
8. **Improved Quality:** The Spiral Model allows for multiple iterations of the software development process, which can result in improved software quality and reliability.