Agile for Developers

Introduction to SDLC

The Software Development Life Cycle (SDLC) is the application of standard business practices to building software applications.

It's typically divided into six to eight steps: Planning, Requirements, Design, Build, Document, Test, Deploy, Maintain.

Some project managers will combine, split, or omit steps, depending on the project's scope.

SDLC is a way to measure and improve the development process

Various people in the organization can use the SDLC such as software engineers, developers, and cross-function teams. Developers and software engineers use it to create effective plans and designs. They also apply the various SDLC stages to develop innovative software products. Cross-function teams use the SDLC to collaborate across different software development stages.

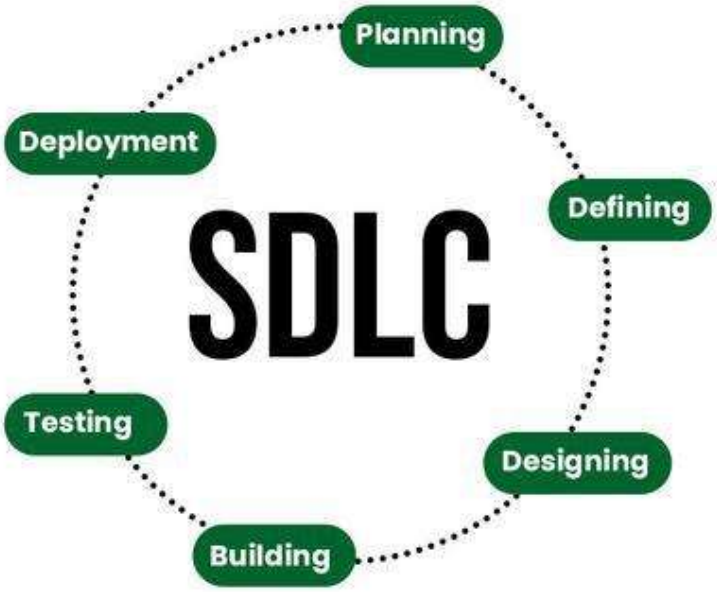
Importance of SDLC

The software development life cycle adds value to software development in the following ways:

- It provides an effective framework and method to develop software applications.
- It helps in effectively planning before starting the actual development. SDLC allows developers to analyze the requirements.
- It helps in reducing unnecessary costs during development. During the initial phases, developers can estimate the costs and predict costly mistakes.
- It enables developers to design and build high-quality software products. This is because they follow a systematic process that allows them to test the software before it is rolled out.
- It provides a basis when evaluating the effectiveness of the software. This further enhances the software product.

Stages of SDLC:

Agile for Developers



Planning

In the Planning phase, project leaders evaluate the terms of the project. This includes calculating labor and material costs, creating a timetable with target goals, and creating the project's teams and leadership structure.

Planning can also include feedback from stakeholders. Planning should clearly define the scope and purpose of the application. It also sets boundaries to help keep the project from expanding or shifting from its original purpose.

Define Requirements

Defining requirements is considered part of planning to determine what the application is supposed to do and its requirements.

Requirements also include defining the resources needed to build the project.

Design and Prototyping

The Design phase models the way a software application will work. Some aspects of the design include:

- Architecture – Specifies programming language, industry practices, overall design, and use of any templates or boilerplate
- User Interface – Defines the ways customers interact with the software, and how the software responds to input

- Platforms – Defines the platforms on which the software will run, such as Apple, Android, Windows version, Linux, or even gaming consoles
- Programming – Not just the programming language, but including methods of solving problems and performing tasks in the application
- Communications – Defines the methods that the application can communicate with other assets, such as a central server or other instances of the application
- Security – Defines the measures taken to secure the application, and may include SSL traffic encryption, password protection, and secure storage of user credentials

Prototyping can be a part of the Design phase. A prototype is like one of the early versions of software in the Iterative software development model. It demonstrates a basic idea of how the application looks and works.

Software Development

This is the actual writing of the program. A small project might be written by a single developer, while a large project might be broken up and worked on by several teams. Use an Access Control or Source Code Management application in this phase.

Testing

It's critical to test an application before making it available to users. Much of the testing can be automated, like security testing.

Testing should ensure that each function works correctly

Deployment

In the deployment phase, the application is made available to users. Many companies prefer to automate the deployment phase. This can be as simple as displaying a payment portal and download link on the company website. It could also be downloading an application on a smartphone.

Operations and Maintenance

At this point, the development cycle is almost finished. The application is done and being used in the field. The Operation and Maintenance phase is still important, though. In this phase, users discover bugs that weren't found during testing.

Agile

Agile for Developers

The Agile model was designed by developers to put customer needs first. This method focuses strongly on user experience and input. Plus, it makes the software highly responsive to customer feedback.

Advantages

- It provides a responsive approach to the development of software since it involves user and customer input.
- Agile is intended for everyone to focus on one task at a time.
- When compared to the waterfall model, the Agile model has a reduced scope which results in better time allocation and estimation.

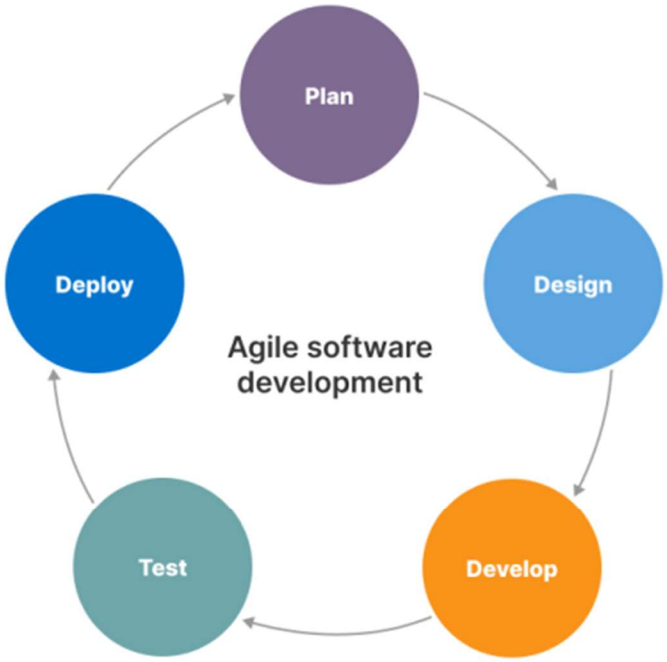
Disadvantages

- The final product depends on the customer. If the customer is not clear on what is needed, the development team may move in the wrong direction.
- If not implemented properly, it could lead to inadequate documentation which hinders technology transfer to new members.

Below is a short list of companies that implement the Agile development model:

- **Philips**
- **VistaPrint**

Agile for Developers



Story Pointing & Burndown Charts

Scrum Ceremonies