Software Development Life Cycle (SDLC)

The Software Development Life Cycle simply outlines each task required to put together a software application. SDLC helps to reduce waste and increase the efficiency of the development process.

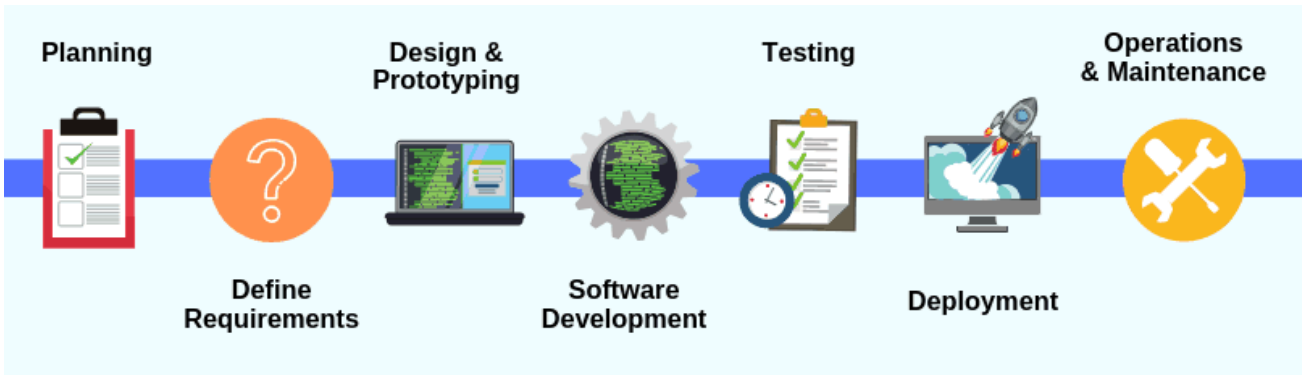
Many companies will subdivide these steps into smaller units. Planning might be broken into technology research, marketing research, and a cost-benefit analysis. Other steps can merge with each other. The Testing phase can run concurrently with the Development phase, since developers need to fix errors that occur during testing.

What is SDLC?

SDLC is a process followed for a software project, within a software organisation. It consists of a detailed plan describing how to develop, maintain, replace and alter or enhance specific software. The life cycle defines a methodology for improving the quality of software and the overall development process.

The Seven Phases of the SDLC

1. Planning
2. Define Requirements
3. Design and Prototyping
4. Software Development
5. Testing
6. Deployment
7. Operations and Maintenance



Stage 1: 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. Stakeholders are anyone who stands to benefit from the application. Try to get feedback from potential customers, developers, subject matter experts, and sales reps.Planning should clearly define the scope and purpose of the application. It plots the course and provisions the team to effectively create the software. It also sets boundaries to help keep the project from expanding or shifting from its original purpose

Feasibility study

A feasibility study is done to measure how practical the development of a software system will be to an organisation. This analysis recurs throughout the life cycle. There we consider five aspects.

1. Technical

In technical feasibility, we check whether the IT solution practical or not, whether the company has the necessary technology to implement the project and whether the company has the necessary personnel with the required capabilities.

2. Operational

In operational feasibility, we analyse whether the project is worth implementing, whether it solves the addressed issue and the feelings of the end-user about the proposed solution or the problem.

3. Schedule

In schedule feasibility, we estimate the time which will take to develop the proposed system and check whether it can be completed within the given timeline.

4. Legal

In legal feasibility, we analyse whether the proposed system violate any kinds of laws. Though legal issues may not arise while the development of the proposed system, problems may arise once it is launched.

5. Economic

Economic analysis is done to analyse the cost and the benefits of the proposed system and to determine whether the initial cost for the proposed system is bearable to the company.

Stage 2 : Define Requirements

Defining requirements is considered part of planning to determine what the application is supposed to do and its requirements. Product design is started with a clear definition of requirements. Software Requirement Specification (SRS) document which consists of all the details of the product requirements should be approved by the clients or the customers before product design begins.

With SRS in hands, more than one design of the product architecture will be proposed based on the requirements in SRS. They will be documented in a Design Document Specification (DDS) by the junior members of the team and passed to the senior members, project stakeholders for review. DDS will be evaluated based on various criteria but not limited to budget, time, user-friendliness, risk, integration, etc.

Stage 3 : Design and Prototyping

The Design phase models the way a software will work. Below are some aspects of the design phase.

1. Architecture - Specifies programming language, industry practices, overall design, and use of any templates or boilerplate

2. User Interface - Defines the ways customers interact with the software, and how the software responds to input

3. Platforms - Defines the platforms on which the software will run, such as Apple, Android, Windows version, Linux, or even gaming consoles

4. Programming - Not just the programming language, but including methods of solving problems and performing tasks in the application

5. Communications - Defines the methods that the application can communicate with other assets, such as a central server or other instances of the application

6. 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. This “hands-on” design can be shown to stakeholders. Use feedback to improve the application. It’s less expensive to change the Prototype phase than to rewrite code to make a change in the Development phase.

Stage 4 : 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 by several teams. Use an Access Control or Source Code Management application in this phase. These systems help developers track changes to the code. They also help ensure compatibility between different team projects and to make sure target goals are being met.

The coding process includes many other tasks. Many developers need to brush up on skills or work as a team. Finding and fixing errors and glitches is critical. Tasks often hold up the development process, such as waiting for test results or compiling code so an application can run. SDLC can anticipate these delays so that developers can be tasked with other duties.

Software developers appreciate instructions and explanations. Documentation can be a formal process, including wiring a user guide for the application. It can also be informal, like comments in the source code that explain why a developer used a certain procedure. Even companies that strive to create software that's easy and intuitive benefit from the documentation.

Documentation can be a quick guided tour of the application's basic features that display on the first launch. It can be video tutorials for complex tasks. Written documentation like user guides, troubleshooting guides, and FAQ's help users solve problems or technical questions.

Stage 5 : Testing

It's critical to test an application before making it available to users. Much of the testing can be automated, like security testing. Other testing can only be done in a specific environment – consider creating a simulated production environment for complex deployments. Testing should ensure that each function works correctly. Different parts of the application should also be tested to work seamlessly together—performance test, to reduce any hangs or lags in processing. The testing phase helps reduce the number of bugs and glitches that users encounter. This leads to a higher user satisfaction and a better usage rate

Stage 6 : 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 a payment portal and download link on the company website. It could also be downloading an application on a smartphone.

Deployment can also be complex. Upgrading a company-wide database to a newly-developed application is one example. Because there are several other systems used by the database, integrating the upgrade can take more time and effort.

Stage 7 : Operational 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. These errors need to be resolved, which can spawn new development cycles.

In addition to bug fixes, models like Iterative development plan additional features in future releases. For each new release, a new Development Cycle can be launched.

Why do we need SDLC?

There are certain reasons to explain why we need Software Development Life Cycle.

1. We can assure the quality of the product and control the quality of the product ensuring the quality of the finished product eliminating defects as much as possible.
2. Within the seven (7) core stages in SDLC, multiple documentations should be prepared to give guidelines and instructions for the programmers and testers to follow and for the managements and approvers to be acknowledged and approve on the activities and action taken. Business Case, Software Requirement Specification (SRS), Design Document Specification (DDS), Functional Specification, Test Plan, Deployment Plan, etc. are all well-defined at each stage. With all the documentations, not much surprises or free-style works can be found in the unexpected areas which implies requirements can be fulfilled and project schedule can be met as planned.
3. Successfully fulfilling user requirements or even exceeding the requirements of the user assuring, they get the best user experience possible.