SOFTWARE Development LIFE CYCLE

What is SDLC?

A software development lifecycle model describes the types of activity performed at each stage in a software development project, and how the activities relate to one another logically and chronologically. There are a number of different software development lifecycle models, each of which requires different approaches to testing.

The following figure is a graphical representation of the various stages of a typical SDLC.

**The Seven Phases of the SDLC**

1. **Planning and requirement analysis**

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.

Once the requirement is understood, the SRS (Software Requirement Specification) document is created. The developers should thoroughly follow this document and also should be reviewed by the customer for future reference.

**2. Define 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.

**3.** **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 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. 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.

**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.

**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.](https://phoenixnap.com/blog/devsecops-best-practices-automated-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.

**6.** **Deployment**

Once the software is certified, and no bugs or errors are stated, then it is deployed.

Based on the assessment, the software may be released as it is or with suggested enhancement in the object segment.

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.

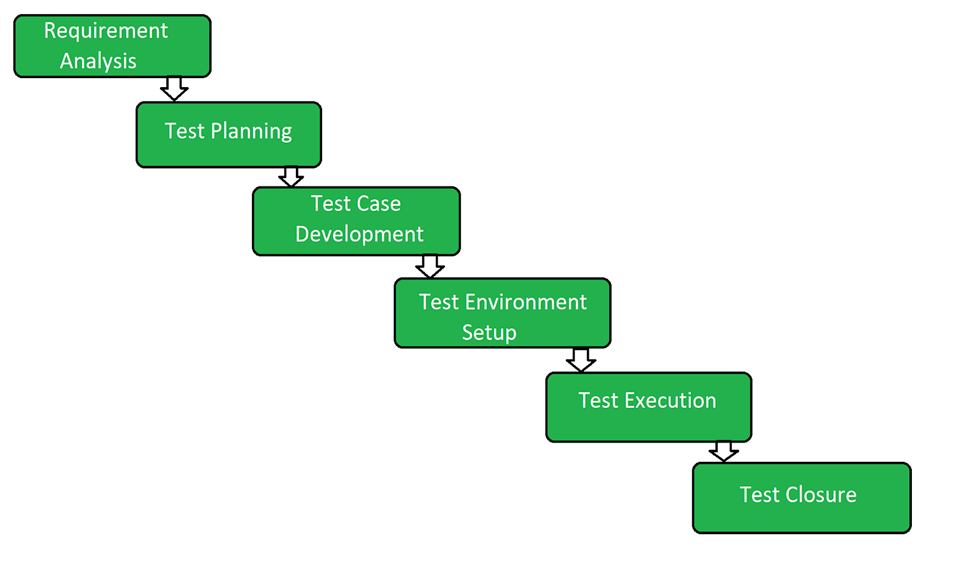
Deployment can also be complex. Upgrading a company-wide database to a newly-developed application is one example. Because there are several other systems used by the database, integrating the upgrade can take more time and effort.

**7.** **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.

**STLC (Software Testing Life Cycle) Phases, Entry, Exit Criteria**

****

**Software Testing Life Cycle (STLC)** is a sequence of specific activities conducted during the testing process to ensure software quality goals are met. STLC involves both verification and validation activities. Contrary to popular belief, Software Testing is not just a single/isolated activity, i.e. testing. It consists of a series of activities carried out methodologically to help certify your software product. STLC stands for Software Testing Life Cycle.

**Characteristics of STLC:**

* STLC is a fundamental part of [Software Development Life Cycle (SDLC)](https://practice.geeksforgeeks.org/problems/software-development-life-cycle) but STLC consists of only the testing phases.
* STLC starts as soon as requirements are defined or software requirement documents are shared by stakeholders.
* STLC yields a step-by-step process to ensure quality software.

**1. Requirement Analysis:**

Requirement Analysis is the first step of Software Testing Life Cycle (STLC). In this phase the QA team understands the requirements like what is to be tested. If anything is missing or not understandable then the QA team meets with the stakeholders to better understand the detailed knowledge of the requirements.

**Activities in Requirement Phase Testing**

* 1. Identify types of tests to be performed.
* 2. Gather details about testing priorities and focus.
* 3. Prepare [Requirement Traceability Matrix (RTM)](https://www.guru99.com/traceability-matrix.html).
* 4. Identify test environment details where testing is supposed to be carried out.
* 5. Automation feasibility analysis (if required).

**2. Test Planning:**

Test Planning is the most efficient phase of software testing life cycle where all testing plans are defined. In this phase the manager of the testing team calculates estimated effort and cost for the testing work. This phase gets started once the requirement gathering phase is completed.

**Test Planning Activities**

* Preparation of test plan/strategy document for various types of testing
* Test tool selection
* Test effort estimation
* Resource planning and determining roles and responsibilities.
* Training requirement

**3. Test Case Development:**

The test case development phase gets started once the test planning phase is completed. In this phase the testing team notes down the detailed test cases. Testing team also prepares the required test data for the testing. When the test cases are prepared then they are reviewed by the QA team.

**Test Case Development Activities**

* Create test cases, automation scripts (if applicable)
* Review and baseline test cases and scripts
* Create test data (If Test Environment is available)

**4. Test Environment Setup:**

Test environment setup is the vital part of the STLC. Basically the test environment decides the conditions on which software is tested. This is independent activity and can be started along with test case development. In this process the testing team is not involved. either the developer or the customer creates the testing environment.

**Test Environment Setup Activities**

* Understand the required architecture, environment set-up and prepare hardware and software requirement list for the Test Environment.
* Setup test Environment and test data
* Perform smoke test on the build

**5. Test Execution:**

After the test case development and test environment setup, the test execution phase gets started. In this phase the testing team starts executing test cases based on prepared test cases in the earlier step.

**Test Execution Activities**

* Execute tests as per plan
* Document test results, and log defects for failed cases
* Map defects to test cases in RTM
* Retest the [Defect](https://www.guru99.com/defect-management-process.html) fixes
* Track the defects to closure

**6. Test Closure:**

This is the last stage of STLC in which the process of testing is analyzed.

**Test Cycle Closure Activities**

* Evaluate cycle completion criteria based on Time, Test coverage, Cost,Software, Critical Business Objectives, Quality
* Prepare test metrics based on the above parameters.
* Document the learning out of the project
* Prepare Test closure report
* Qualitative and quantitative reporting of quality of the work product to the customer.
* Test result analysis to find out the defect distribution by type and severity.