THE SOFTWARE DEVELOPMENT LIFECYCLE (SDLC): AN INTRODUCTION

Software Development Lifecycle (SDLC) refers to the systematic development process of software. The lifecycle spans several stages, which we detail below, that ensure high quality software is built and released to end-users at a fast pace and optimized cost. Software quality might be determined by the robustness of its functionality, its performance, its security and, ultimately, the end-user experience.

In this article, we will discuss the SDLC process and the individual stages that comprise it.

Who uses the SDLC?

Not so long ago, Watt S. Humphrey, known as the father of quality in software, quoted, “Every business is a software business”. More recently, Microsoft CEO Satya Nadella repeated the quote: “Every company is a software company”. Business organizations driven by software technologies must go beyond digital transformation by adapting off-the-shelf technology solutions. These companies must develop capabilities internally to develop software solutions that optimize business operations. Additionally, conventional organizations are innovating their business models to serve end-users through digital channels or leverage technology platforms for improved delivery of physical goods and services.

As such, the SDLC approach is not limited only to developers or engineers. Cross-function teams adopt the SDLC mechanism in order to collaborate across various stages of the SDLC and work

collectively using various SDLC frameworks such as Agile and DevOps. By following modern SDLC practices and frameworks (see below), the software development process can improve significantly.

Stages of the SDLC

The SDLC follows a series of phases involved in software development. Depending on the SDLC framework, these phases may be adopted sequentially or in parallel. The SDLC workflows may involve repeated transitions or iterations across the phases before reaching the final phase.

Phase 1: Requirement Analysis

The initial stage of the SDLC involves stakeholders from tech, business, and leadership segments of the organization. Activities include:

- Analyze and translate business questions into engineering problems by considering a variety of factors: cost, performance, functionality, and risk.
- Evaluate he broad scope of the project and then identify available resources.
- Consider project opportunities and risks across the technical and business aspect for every decision choice in each SDLC phase.

This stage may continue for a prolonged period and includes provision for strategic changes as the SDLC evolves.

Phase 2: Feasibility Study

During this stage, evaluate the requirements for feasibility. The goal is to quantify the opportunities and risk of addressing the agreed requirements with the variety of resources and strategies available to the organization. The feasibility study evaluates the following key aspects, among others:

- **Economic:** Is it financially viable to invest in the project based on the available resources?
- **Legal:** What is the scope of regulations and the organization's capacity to guarantee compliance?
- **Operational:** Can we satisfy the requirements within scope definition according to the proposed operational framework and workflows?
- **Technical:** What is the availability of technology and HR resources to support the SLDC process?
- **Schedule:** Can we finish the project in time?

Executive decision makers should answer and document these questions and study them

carefully—before proceeding with the software design and implementation process.

Phase 3: Architectural Design

At this stage, the design specifications are documented and reviewed by appropriate technical and business stakeholders. Evaluate design choices against the risk, opportunities, practical modalities, and constraints.

Technical documentation specifies systems architecture, configurations, data structure, resource procurement model. Desired output can include prototypes, pseudocode. and architecture reports and diagrams that include the necessary technology details. The high-level design details include the desired functionality of software and system modules. The low-level design details can include the functional logic, interface details, dependency issues, and errors.

Phase 4: Software Development

Implementation follows the design phase. Several independent teams and individuals collaborate on feature development and coding activities. Individual developers may build their own codebase within the development environment before merging it with the collaborating teams in a common build environment.

While the requirements analysis and design choices are already defined, feedback from the development teams is reviewed for potential change in direction of the design strategies. This is the longest process in the SDLC pipeline and it assists subsequent phases of software testing and deployment.

Phase 5: Testing

Activities in this phase are focused on investigating the performance of the software and discovering potential issues. Testing teams develop a test plan based on the predefined software requirements. The plan identifies the resources available for testing, instructions and assignments for testers, selects types of tests to be conducted and reports to technical executives and decision makers. Testers often work collectively with development teams and rework the codebase to improve test results.

At SDLC phases 4 and 4, software builds may be improved several times before the final product is sent to the deployment and production environment.

Phase 6: Deployment

In the final phase of the SDLC pipeline, the finished product has passed the necessary tests. Now, make it available for release to end users in the real environment. Several procedures and preparation activities are involved before a software product can be shipped, including:

- Documentation
- Transferring ownership and licensing,
- Deploying and installing the product on customer systems

Traditional vs modern SDLC methodologies

With conventional SDLC methodologies, such as Waterfall, these phases are performed independently in series by disparate teams. Under the Agile methodology, these phases are performed in short, iterative, incremental sprints. An SDLC pipeline and framework can be as varied as the number of organizations adopting them -- virtually every company identifies specific components of every SDLC phase and tries to adopt a strategy that works best for their organization.

In today's era of software development, however, these stages are not always followed sequentially. Modern SDLC frameworks such as DevOps and Agile encourage cross-functional organizations to share responsibilities across these phases conducted in parallel.

For instance, the DevOps SDLC framework encourages Devs, Ops, and QA personnel to work together for continuous development, testing and deployment activities. Additionally, the testing procedure is shifted left and early in the SDLC pipeline such that software defects are identified before it's too late to fix them.

Additional resources

For more information on software development and the SDLC, check out these BMC Blogs:

- Differences Between Continuous Integration (CI), Delivery (CD), and Deployment
- What is DevOps? A Basic Introduction
- Orchestration in SDLC for DevOps
- Agile Roles and Responsibilities
- Intro to Agile with Scrum: 4 Tips for Getting Started