Software Development Life Cycle Models

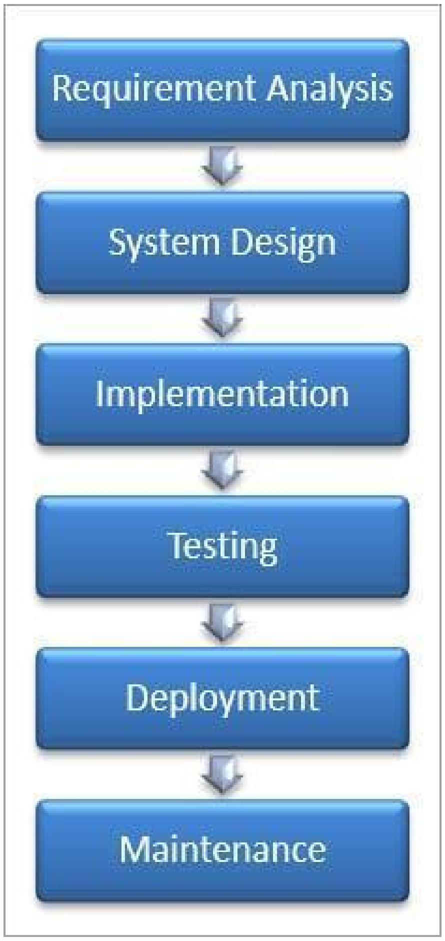
A software life cycle model is a diagram that depicts the software development process. Although each SDLC model takes a distinct approach, the core phases and activities are the same.

1. Waterfall Model

The waterfall model is the first model used in the SDLC. The linear sequential model is another name for it.

The output of one step is the input for the following phase in this paradigm. The next step can be developed only when the preceding phase has been completed.

- First, requirements are gathered and analyzed. Only after the condition has been frozen can the System Design begin. The SRS document results from the Requirement phase and serves as an input to the System Design phase.
- Documents that serve as input for the next phase, i.e., implementation and coding, are prepared in System Design Software Architecture and Design.
- Coding is completed in the Implementation step, and the software created serves as an input for the testing phase.
- The created code is rigorously checked during the testing process to uncover software faults. Defects are recorded in the defect tracking tool and retested once resolved. Bug logging, retesting, and regression testing continues until the software is ready to go live.
- The developed code is moved into production when the customer signs off in the Deployment phase.
- Any issues that arise in the production environment are dealt with through the developers in charge of maintenance.



- **The Waterfall Model has the following advantages:**

- The waterfall model is a concise model that is easy to understand and in which all phases are completed sequentially.
- Each phase's deliverables are carefully specified, resulting in no complexity and easy project management.

- **Disadvantages of the Waterfall model:**
    - The waterfall model is time-consuming and cannot be employed in short-term projects because a new phase cannot begin until the previous one is completed.
    - The waterfall model cannot be used for projects with vague requirements or where the conditions frequently change because it expects the requirements to be precise during the requirement gathering and analysis phase. Later changes would result in higher costs because they might require changes in all stages.