❖ Prescriptive Process Models: The Waterfall Model

The Waterfall Model was the first Process Model to be introduced. It is also referred to as a linear-sequential life cycle model. It is very simple to understand and use. In a waterfall model, each phase must be completed before the next phase can begin and there is no overlapping in the phases.

The Waterfall model is the earliest SDLC approach that was used for software development.

The waterfall Model illustrates the software development process in a linear sequential flow. This means that any phase in the development process begins only if the previous phase is complete. In this waterfall model, the phases do not overlap.

The waterfall model is a software development model used in the context of large, complex projects, typically in the field of information technology. It is characterized by a structured, sequential approach to project management and software development.

The waterfall model is useful in situations where the project requirements are well-defined and the project goals are clear. It is often used for large-scale projects with long timelines, where there is little room for error and the project stakeholders need to have a high level of confidence in the outcome.

Features of Waterfall Model

- Sequential Approach: The waterfall model involves a sequential approach to software development, where each phase of the project is completed before moving on to the next one.
- Document-Driven: The waterfall model relies heavily on documentation to ensure that the project is well-defined and the project team is working towards a clear set of goals.
- Quality Control: The waterfall model places a high emphasis on quality control and testing at each phase of the project, to ensure that the final product meets the requirements and expectations of the stakeholders.
- Rigorous Planning: The waterfall model involves a rigorous planning process, where the project scope, timelines, and deliverables are carefully defined and monitored throughout the project lifecycle.

Prof. Sheetal Jadhav

Overall, the waterfall model is used in situations where there is a need for a highly structured and systematic approach to software development. It can be effective in ensuring that large, complex projects are completed on time and within budget, with a high level of quality and customer satisfaction.

Phases of Waterfall Model

Waterfall Model is a classical software development methodology that was first introduced by Winston W. Royce in 1970. It is a linear and sequential approach to software development that consists of several phases that must be completed in a specific order.

The Waterfall Model has six phases:

1. Requirements Gathering and Analysis: The first phase involves gathering requirements from stakeholders and analyzing them to understand the scope and objectives of the project.

2. Design Phase: Once the requirements are understood, the design phase begins. This involves creating a detailed design document that outlines the software architecture, user interface, and system components.

3. Implementation and Unit Testing: The implementation phase involves coding the software based on the design specifications. This phase also includes unit testing to ensure that each component of the software is working as expected.

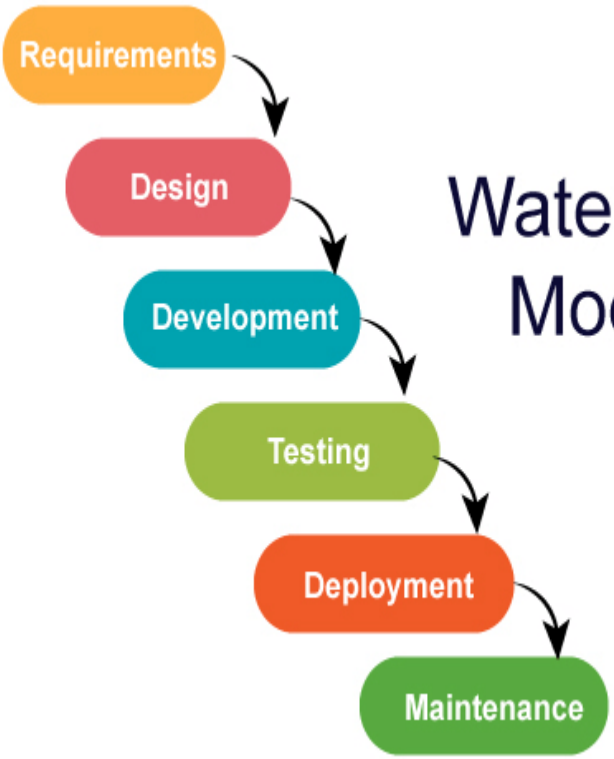
4. Integration and System Testing: In the testing phase, the software is tested as a whole to ensure that it meets the requirements and is free from defects.

5. Deployment: Once the software has been tested and approved, it is deployed to the production environment.

6. Maintenance: The final phase of the Waterfall Model is maintenance, which involves fixing any issues that arise after the software has been deployed and ensuring that it continues to meet the requirements over time.

The classical waterfall model divides the life cycle into a set of phases. This model considers that one phase can be started after the completion of the previous phase. That is the output of one phase will be the input to the next phase. Thus the development process can be considered as a sequential flow in the waterfall. Here the phases do not overlap with each other. The different sequential phases of the classical waterfall model are shown in the below figure.

Prof. Sheetal Jadhav



Let us now learn about each of these phases in detail.

1. Feasibility Study

The main goal of this phase is to determine whether it would be financially and technically feasible to develop the software.

The feasibility study involves understanding the problem and then determining the various possible strategies to solve the problem. These different identified solutions are analyzed based on their benefits and drawbacks, The best solution is chosen and all the other phases are carried out as per this solution strategy.

2. Requirements Analysis and Specification

The aim of the requirement analysis and specification phase is to understand the exact requirements of the customer and document them properly. This phase consists of two different activities.

Requirement gathering and analysis: Firstly all the requirements regarding the software are gathered from the customer and then the gathered requirements are analyzed. The goal of the analysis part is to remove incompleteness (an incomplete requirement is one in which some parts of the actual requirements have been omitted) and inconsistencies (an inconsistent requirement is one in which some part of the requirement contradicts some other part).

Requirement specification: These analyzed requirements are documented in a software requirement specification (SRS) document. SRS document serves as a contract between the development team and customers. Any future dispute between the customers and the developers can be settled by examining the SRS document.

3. Design

The goal of this phase is to convert the requirements acquired in the SRS into a format that can be coded in a programming language. It includes high-level and detailed design as well as the overall software architecture. A Software Design Document is used to document all of this effort (SDD).

4. Coding and Unit Testing

In the coding phase software design is translated into source code using any suitable programming language. Thus each designed module is coded. The aim of the unit testing phase is to check whether each module is working properly or not.

5. Integration and System testing

Integration of different modules is undertaken soon after they have been coded and unit tested. Integration of various modules is carried out incrementally over a number of steps. During each integration step, previously planned modules are added to the partially integrated system and the resultant system is tested. Finally, after all the modules have been successfully integrated and tested, the full working system is obtained and system testing is carried out on this.

System testing consists of three different kinds of testing activities as described below.

Alpha testing: Alpha testing is the system testing performed by the development team.

Beta testing: Beta testing is the system testing performed by a friendly set of customers.

Acceptance testing: After the software has been delivered, the customer performed acceptance testing to determine whether to accept the delivered software or reject it.

Prof. Sheetal Jadhav

6. Maintenance

Maintenance is the most important phase of a software life cycle. The effort spent on maintenance is 60% of the total effort spent to develop a full software. There are basically three types of maintenance.

Corrective Maintenance: This type of maintenance is carried out to correct errors that were not discovered during the product development phase.

Perfective Maintenance: This type of maintenance is carried out to enhance the functionalities of the system based on the customer's request.

Adaptive Maintenance: Adaptive maintenance is usually required for porting the software to work in a new environment such as working on a new computer platform or with a new operating system.

Advantages of Classical Waterfall Model

The classical waterfall model is an idealistic model for software development. It is very simple, so it can be considered the basis for other software development life cycle models. Below are some of the major advantages of this SDLC model.

Easy to Understand: Classical Waterfall Model is very simple and easy to understand.

Individual Processing: Phases in the Classical Waterfall model are processed one at a time.

Properly Defined: In the classical waterfall model, each stage in the model is clearly defined.

Clear Milestones: Classical Waterfall model has very clear and well-understood milestones.

Properly Documented: Processes, actions, and results are very well documented.

Reinforces Good Habits: Classical Waterfall Model reinforces good habits like define-before-design and design-before-code.

Working: Classical Waterfall Model works well for smaller projects and projects where requirements are well understood.

Prof Sheetal Jadhav

Disadvantages of Classical Waterfall Model

The Classical Waterfall Model suffers from various shortcomings, basically, we can't use it in real projects, but we use other software development lifecycle models which are based on the classical waterfall model. Below are some major drawbacks of this model.

No Feedback Path: In the classical waterfall model evolution of software from one phase to another phase is like a waterfall. It assumes that no error is ever committed by developers during any phase. Therefore, it does not incorporate any mechanism for error correction.

Difficult to accommodate Change Requests: This model assumes that all the customer requirements can be completely and correctly defined at the beginning of the project, but actually customer's requirements keep on changing with time. It is difficult to accommodate any change requests after the requirements specification phase is complete.

No Overlapping of Phases: This model recommends that a new phase can start only after the completion of the previous phase. But in real projects, this can't be maintained. To increase efficiency and reduce cost, phases may overlap.

Limited Flexibility: The Waterfall Model is a rigid and linear approach to software development, which means that it is not well-suited for projects with changing or uncertain requirements. Once a phase has been completed, it is difficult to make changes or go back to a previous phase.

Limited Stakeholder Involvement: The Waterfall Model is a structured and sequential approach, which means that stakeholders are typically involved in the early phases of the project (requirements gathering and analysis) but may not be involved in the later phases (implementation, testing, and deployment).

Late Defect Detection: In the Waterfall Model, testing is typically done toward the end of the development process. This means that defects may not be discovered until late in the development process, which can be expensive and time-consuming to fix.

Lengthy Development Cycle: The Waterfall Model can result in a lengthy development cycle, as each phase must be completed before moving on to the next. This can result in delays and increased costs if requirements change or new issues arise.

Not Suitable for Complex Projects: The Waterfall Model is not well-suited for complex projects, as the linear and sequential nature of the model can make it difficult to manage multiple dependencies and interrelated components.

Prof.Sheetal Jadhav

When to Use a Waterfall Model?

Here are some cases where use of Waterfall Model is best suited:

Only well-defined, unambiguous, and fixed requirements are employed with this paradigm.

The definition of a product is constant.

People understand technology.

There are no unclear prerequisites.

There are many resources with the necessary knowledge readily available.

When it's a brief project.

The Waterfall approach involves little client engagement in the product development process. The product can only be shown to end consumers when it is ready.

Applications of Classical Waterfall Model

Large-scale Software Development Projects: The Waterfall Model is often used for large-scale software development projects, where a structured and sequential approach is necessary to ensure that the project is completed on time and within budget.

Safety-Critical Systems: The Waterfall Model is often used in the development of safety-critical systems, such as aerospace or medical systems, where the consequences of errors or defects can be severe.

Government and Defense Projects: The Waterfall Model is also commonly used in government and defense projects, where a rigorous and structured approach is necessary to ensure that the project meets all requirements and is delivered on time.

Projects with well-defined Requirements: The Waterfall Model is best suited for projects with well-defined requirements, as the sequential nature of the model requires a clear understanding of the project objectives and scope.

Projects with Stable Requirements: The Waterfall Model is also well-suited for projects with stable requirements, as the linear nature of the model does not allow for changes to be made once a phase has been completed.

Prof. Sheetal Jadhav