### The waterfall model

The waterfall model was the first software process model to be introduced. It is also referred to as a linear-sequential life cycle model. The model was originally proposed by Royce. The waterfall model was popular in 1970s. it is simple but idealistic. The classical waterfall model divides the life cycle into a set of phases. The principal stages of the model represent the fundamental development activities:


The different phases are :-

Feasibility Study.

Requirement Analysis and Specification.

Design

Coding and Unit Testing

Integration and System Testing

Maintenance.

The phases starting from the feasibility study to the integration and system testing phases are known as the development phases. A software is developed during the development phases, and at the completion of development phases, the software is delivered to the customer. Therefore, the last phase is also known as the maintenance phase of the life cycle.

In the waterfall model different life cycle phases typically require relatively different amounts of efforts to be put in by the development team. The maintenance phase normally requires the maximum effort. On the average about 60% of the total effort put in by the development team in the entire life cycle is spent on the maintenance activity alone.

This can be shown by graph according phase and effort.

**Feasibility Study:-** The main focus of the feasibility study stage is to determine whether it would be financially and technically feasible to develop the software or not.

**Requirement analysis and specification :-** The aim of the requirement analysis and specification phase is to understand the exact requirement of the customer and to document them properly. This phase consists of two distinct activities namely requirements gathering and analysis ,and requirement specification.

 **Requirement gathering and analysis:-** The goal of the requirement gathering activity is to collect all relevant information regarding the software to be developed from the customer with a view to clearly understand the requirements. For this first requirements are gathered from the customer and then the gathered requirements are analysed.

**Requirement specification :-** After the requirement gathering and analysis activities are complete, the identified requirements are documented. This is called a software requirements specification(SRS) document. The SRS document is written using end-user terminology. This makes the SRS document understandable to the customer. Therefore understandability of the SRS document is an important issue. The SRS document normally serves as a contract between the customer and development team . Any future dispute between the customer and the developers team can be settled by examining the SRS document. The SRS document is therefore an important which must be thoroughly understood by the development team and reviewed jointly with the customer. The SRS document not only forms the basis for carrying out all the development activities but several documents such as user’s manuals, system test plan etc. are prepared directly based on it.

**Design:-** The goal of the design phase is to transform the requirements specified in the SRS document into a structure that is suitable for implementation in some programming language. In technical terms during the design phase the software architecture is derived from the SRS document. Two distinctly different design approaches are popularly being used at present:- the procedural and object oriented design approaches.

**Coding and unit testing:-** The purpose of the coding and unit testing is to translate a software design into source code and to ensure that individually each function is working correctly. The coding phase is also sometime called the implementation phase, since the design is implemented into a workable solution in this phase. Each component of the design is implemented as a program module. The end product of this phase is a set of program module that have been individually unit tested. The main objective of unit testing is to determine the correct working of the individual modules.

**Integration and system testing:-** Integration of different modules is undertaken soon after they have been coded and unit tested. During the integration and system testing phase the different modules are integrated in a planned manner. Various modules making up a software are almost never integrated in a one shot. Integration of various modules are normally 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.

 System testing usually consist of three different kinds of testing activities:-

**Alfa testing:-** A testing is the system testing performed by the development team.

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

**Acceptance testing:-** After the software has been delivered the customer performs system testing to determine whether to accept the delivered software or to reject it.

Maintenance:- The total effort spent on maintenance of a typical software during its operation phase is much more than that required for developing the software itself. The relative effort ratio of developing a typical software product and the total effort spent on its maintenance is roughly 40:60.

Maintenance is required in three situations:-

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 improve the performance of the system or enhance the functionality of the system based on customer’ requests.

Adaptive Maintenance:- Adaptive Maintenance is usually required for porting the software to work on a new computer platform or with a new operating system.

Shortcomings of the waterfall model

This is very simple model however it suffers from several shortcomings.

No Feedback Path

Difficult to accommodate change requests

Inefficient error correction

No overlapping of phases