UNIT - 1

Software Engineering Introduction

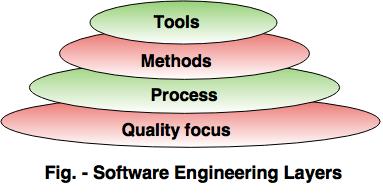
* Computer software is a product or program code developed by software engineers.
* The applications of computer software are: Telecommunication, military, medical sciences, online shopping, office products, IT industry etc.
* A Software consists of data and the related documents.
* The software is the key element in all computer based systems and products.
* The main purpose behind software engineering is to give a framework for building a software with best quality.

Software engineering definitions

* The establishment and use of sound engineering principles in order to obtain economical software that is reliable and works efficiently on real machines.
* Software engineering is a systematic and disciplined approach towards the development of the software operation and maintenance.
* Software engineering is an engineering branch associated with the development of software product using well-defined scientific principles, methods and procedures.
* Characteristics of a software
* Software should achieve a good quality in design and meet all the specifications of the customer.
* Software does not wear out i.e. it does not lose the material.
* Software should be inherently complex.
* Software must be efficient i.e. the ability of the software to use system resources in an effective and efficient manner.
* Software must be integral i.e. it must prevent from unauthorized access to the software or data.

Software Engineering - Layered Technology

* Software engineering is a fully layered technology.
* To develop a software, we need to go from one layer to another.
* All these layers are related to each other and each layer demands the fulfilment of the previous layer.

  
  
**The layered technology consists of:**  
  
**1. Quality focus**

**The characteristics of good quality software are:**

* Correctness of the functions required to be performed by the software.
* Maintainability of the software
* Integrity i.e. providing security so that the unauthorized user cannot access information or data.
* Usability i.e. the efforts required to use or operate the software.

**2. Process**

* It is the base layer or foundation layer for the software engineering.
* The software process is the key to keep all levels together.
* It defines a framework that includes different activities and tasks.
* In short, it covers all  activities, actions and tasks required to be carried out for software development.

**3. Methods**

* The method provides the answers of all 'how-to' that are asked during the process.
* It provides the technical way to implement the software.
* It includes collection of tasks starting from communication, requirement analysis, analysis and design modelling, program construction, testing and support.

**4. Tools**

* The software engineering tool is an automated support for the software development.
* The tools are integrated i.e the information created by one tool can be used by the other tool.
* For example: The Microsoft publisher can be used as a web designing tool.

Software Development Life Cycle (SDLC)

A software life cycle model (also termed process model) is a pictorial and diagrammatic representation of the software life cycle.

A life cycle model represents all the methods required to make a software product transit through its life cycle stages.

Need of SDLC

The development team must determine a suitable life cycle model for a particular plan and then observe to it.

Without using an exact life cycle model, the development of a software product would not be in a systematic and disciplined manner.

When a team is developing a software product, there must be a clear understanding among team representative about when and what to do.

A software life cycle model describes entry and exit criteria for each phase.

A phase can begin only if its stage-entry criteria have been fulfilled.

So without a software life cycle model, the entry and exit criteria for a stage cannot be recognized.

Without software life cycle models, it becomes tough for software project managers to monitor the progress of the project.



**Stage1: Planning and requirement analysis**

Requirement Analysis is the most important and necessary stage in SDLC.

The senior members of the team perform it with inputs from all the stakeholders and domain experts in the industry.

Once the required function is done, an analysis is complete with auditing the feasibility of the growth of a product. In case of any ambiguity, a signal is set up for further discussion.

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.

**Stage2: Defining 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.

**Stage3: Designing the Software**

The next phase is about to bring down all the knowledge of requirements, analysis, and design of the software project.

This phase is the product of the last two, like inputs from the customer and requirement gathering.

**Stage4: Developing the project**

in this phase of SDLC, the actual development begins, and the programming is built. The implementation of design begins concerning writing code.

Developers have to follow the coding guidelines described by their management and programming tools like compilers, interpreters, debuggers, etc. are used to develop and implement the code.

**Stage5: Testing**

After the code is generated, it is tested against the requirements to make sure that the products are solving the needs addressed and gathered during the requirements stage.

During this stage, unit testing, integration testing, system testing, acceptance testing are done.

**Stage6: Deployment**

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

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

After the software is deployed, then its maintenance begins.

**Stage7: Maintenance**

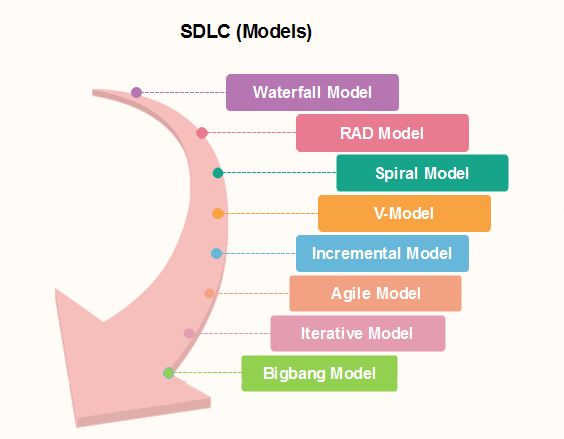
when the client starts using the developed systems, then the real issues come up and requirements to be solved from time to time.

This procedure where the care is taken for the developed product is known as maintenance.

SDLC Models

Software Development life cycle (SDLC) is a spiritual model used in project management that defines the stages include in an information system development project, from an initial feasibility study to the maintenance of the completed application.

There are different software development life cycle models specify and design, which are followed during the software development phase. These models are also called "**Software Development Process Models**." Each process model follows a series of phase unique to its type to ensure success in the step of software development.

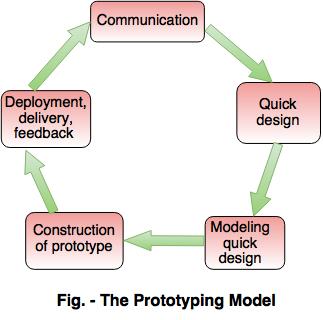


Evolutionary Process Models

**Following are the evolutionary process models.**  
  
1. The prototyping model  
2. The spiral model  
3. Concurrent development model

1. The Prototyping model

* Prototype is defined as first or preliminary form using which other forms are copied or derived.
* Prototype model is a set of general objectives for software.
* It does not identify the requirements like detailed input, output.
* It is software working model of limited functionality.
* In this model, working programs are quickly produced.

  
  
**The different phases of Prototyping model are:**  
  
**1. Communication**

In this phase, developer and customer meet and discuss the overall objectives of the software.  
  
**2. Quick design**

* Quick design is implemented when requirements are known.
* It includes only the important aspects like input and output format of the software.
* It focuses on those aspects which are visible to the user rather than the detailed plan.
* It helps to construct a prototype.

**3. Modeling quick design**

* This phase gives the clear idea about the development of software because the software is now built.
* It allows the developer to better understand the exact requirements.

**4. Construction of prototype**

The prototype is evaluated by the customer itself.  
  
**5. Deployment, delivery, feedback**

* If the user is not satisfied with current prototype then it refines according to the requirements of the user.
* The process of refining the prototype is repeated until all the requirements of users are met.
* When the users are satisfied with the developed prototype then the system is developed on the basis of final prototype.

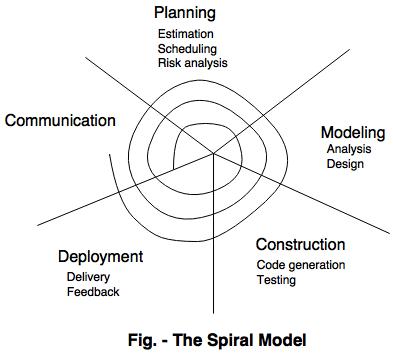
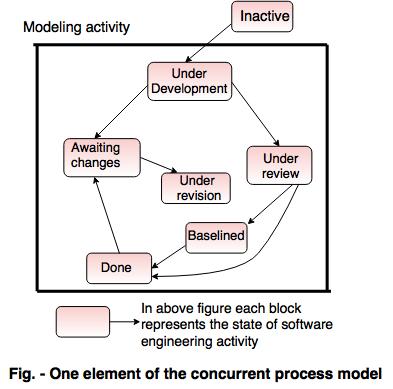
**Advantages of Prototyping Model**

* Prototype model need not know the detailed input, output, processes, adaptability of operating system and full machine interaction.
* In the development process of this model users are actively involved.
* The development process is the best platform to understand the system by the user.
* Errors are detected much earlier.
* Gives quick user feedback for better solutions.
* It identifies the missing functionality easily. It also identifies the confusing or difficult functions.

Disadvantages of Prototyping Model:

* The client involvement is more and it is not always considered by the developer.
* It is a slow process because it takes more time for development.
* Many changes can disturb the rhythm of the development team.
* It is a thrown away prototype when the users are confused with it.

2. The Spiral model

* Spiral model is a risk driven process model.
* It is used for generating the software projects.
* In spiral model, an alternate solution is provided if the risk is found in the risk analysis, then alternate solutions are suggested and implemented.
* It is a combination of prototype and sequential model or waterfall model.
* In one iteration all activities are done, for large project's the output is small.
* The framework activities of the spiral model are as shown in the following figure.  
    
    
    
  NOTE: The description of the phases of the spiral model is same as that of the process model.  
    
  Advantages of Spiral Model
* It reduces high amount of risk.
* It is good for large and critical projects.
* It gives strong approval and documentation control.
* In spiral model, the software is produced early in the life cycle process.
* Disadvantages of Spiral Model
* It can be costly to develop a software model.
* It is not used for small projects.
* 3. The concurrent development model
* The concurrent development model is called as concurrent model.
* The communication activity has completed in the first iteration and exits in the awaiting changes state.
* The modeling activity completed its initial communication and then go to the underdevelopment state.
* If the customer specifies the change in the requirement, then the modeling activity moves from the under development state into the awaiting change state.
* The concurrent process model activities moving from one state to another state.
*   
    
  **Advantages of the concurrent development model**
* This model is applicable to all types of software development processes.
* It is easy for understanding and use.
* It gives immediate feedback from testing.
* It provides an accurate picture of the current state of a project.

**Disadvantages of the concurrent development model**

* It needs better communication between the team members. This may not be achieved all the time.
* It requires to remember the status of the different activities.

# **Requirement Engineering**

* **Requirements engineering (RE)** refers to the process of defining, documenting, and maintaining requirements in the engineering design process. Requirement engineering provides the appropriate mechanism to understand what the customer desires, analyzing the need, and assessing feasibility, negotiating a reasonable solution, specifying the solution clearly, validating the specifications and managing the requirements as they are transformed into a working system. Thus, requirement engineering is the disciplined application of proven principles, methods, tools, and notation to describe a proposed system's intended behavior and its associated constraints.

Requirement Engineering Process

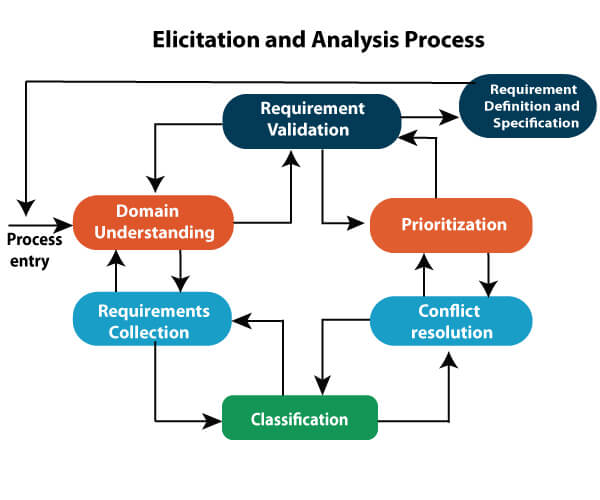
It is a five-step process, which includes –

1. Feasibility Study
2. Requirement Elicitation and Analysis
3. Software Requirement Specification
4. Software Requirement Validation
5. Software Requirement Management
6. **1.** Feasibility Study:
7. The objective behind the feasibility study is to create the reasons for developing the software that is acceptable to users, flexible to change and conformable to established standards.
8. **Technical Feasibility** - Technical feasibility evaluates the current technologies, which are needed to accomplish customer requirements within the time and budget.
9. **Operational Feasibility** - Operational feasibility assesses the range in which the required software performs a series of levels to solve business problems and customer requirements.
10. **Economic Feasibility** - Economic feasibility decides whether the necessary software can generate financial profits for an organization.

2. Requirement Elicitation and Analysis:

This is also known as the **gathering of requirements**. Here, requirements are identified with the help of customers and existing systems processes, if available.

1. Analysis of requirements starts with requirement elicitation. The requirements are analyzed to identify inconsistencies, defects, omission, etc. We describe requirements in terms of relationships and also resolve conflicts if any.

–––

### **Software Requirement Specification:**

1. Software requirement specification is a kind of document which is created by a software analyst after the requirements collected from the various sources - the requirement received by the customer written in ordinary language. It is the job of the analyst to write the requirement in technical language so that they can be understood and beneficial by the development team.
2. The models used at this stage include ER diagrams, data flow diagrams (DFDs), function decomposition diagrams (FDDs), data dictionaries, etc.

* **Data Flow Diagrams:** Data Flow Diagrams (DFDs) are used widely for modeling the requirements. DFD shows the flow of data through a system. The system may be a company, an organization, a set of procedures, a computer hardware system, a software system, or any combination of the preceding. The DFD is also known as a data flow graph or bubble chart.
* **Data Dictionaries:** Data Dictionaries are simply repositories to store information about all data items defined in DFDs. At the requirements stage, the data dictionary should at least define customer data items, to ensure that the customer and developers use the same definition and terminologies.
* **Entity-Relationship Diagrams:** Another tool for requirement specification is the entity-relationship diagram, often called an "*E-R diagram*." It is a detailed logical representation of the data for the organization and uses three main constructs i.e. data entities, relationships, and their associated attributes.

### **4.** Software Requirement Validation:

After requirement specifications developed, the requirements discussed in this document are validated. The user might demand illegal, impossible solution or experts may misinterpret the needs. Requirements can be the check against the following conditions -

* If they can practically implement
* If they are correct and as per the functionality and specially of software
* If there are any ambiguities
* If they are full
* If they can describe

**Requirements Validation Techniques**

* **Requirements reviews/inspections:** systematic manual analysis of the requirements.
* **Prototyping:** Using an executable model of the system to check requirements.
* **Test-case generation:** Developing tests for requirements to check testability.
* **Automated consistency analysis:** checking for the consistency of structured requirements descriptions.

### Software Requirement Management:

Requirement management is the process of managing changing requirements during the requirements engineering process and system development.

New requirements emerge during the process as business needs a change, and a better understanding of the system is developed.

The priority of requirements from different viewpoints changes during development process.

The business and technical environment of the system changes during the development.

## Prerequisite of Software requirements

Collection of software requirements is the basis of the entire software development project. Hence they should be clear, correct, and well-defined.

Software Requirements: Largely software requirements must be categorized into two categories:

1. Functional Requirements: Functional requirements define a function that a system or system element must be qualified to perform and must be documented in different forms. The functional requirements are describing the behavior of the system as it correlates to the system's functionality.
2. **Non-functional Requirements:** This can be the necessities that specify the criteria that can be used to decide the operation instead of specific behaviors of the system.

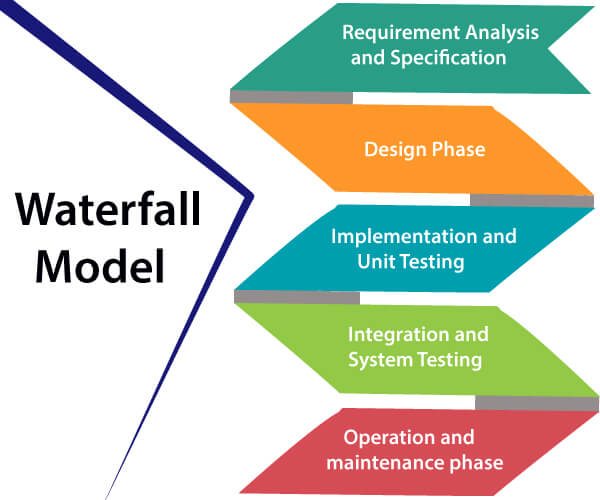
Non-functional requirements are divided into two main categories:

* + **Execution qualities** like security and usability, which are observable at run time.
  + **Evolution qualities** like testability, maintainability, extensibility, and scalability that embodied in the static structure of the software system.

# **Waterfall model**

Winston Royce introduced the Waterfall Model in 1970.This model has five phases: Requirements analysis and specification, design, implementation, and unit testing, integration and system testing, and operation and maintenance. The steps always follow in this order and do not overlap. The developer must complete every phase before the next phase begins. This model is named "**Waterfall Model**", because its diagrammatic representation resembles a cascade of waterfalls.

**1. Requirements analysis and specification phase:** The aim of this phase is to understand the exact requirements of the customer and to document them properly. Both the customer and the software developer work together . In this phase, a large document called **Software Requirement Specification (SRS)** document is created which contained a detailed description of what the system will do in the common language.



**2. Design Phase:** This phase aims to transform the requirements gathered in the SRS into a suitable form which permits further coding in a programming language. It defines the overall software architecture together with high level and detailed design. All this work is documented as a Software Design Document (SDD).

**3. Implementation and unit testing:** During this phase, design is implemented. If the SDD is complete, the implementation or coding phase proceeds smoothly, because all the information needed by software developers is contained in the SDD.

