Introduction to Software Engineering

In software engineering the **systematic** and **organized approach** is adopted. Based on the nature of the problem and development constraints various **tools and techniques** are applied in order to develop **quality software**.

The definition of software engineering is based on two terms :

- **Discipline :** For finding the solution to the problem an Engineer applies appropriate theories, methods and tools. While finding the solutions, Engineers must think ofthe organizational and financial constraints. Within these constraints only, he/she has to find the solution.
- **Product :** The software product gets developed after following systematic theories, methods and tools along with the appropriate management activities.

Defining Software

Software is nothing but a collection of computer programs and related documents that are intended to provide desired features, functionalities and better performance.

Software products may be :

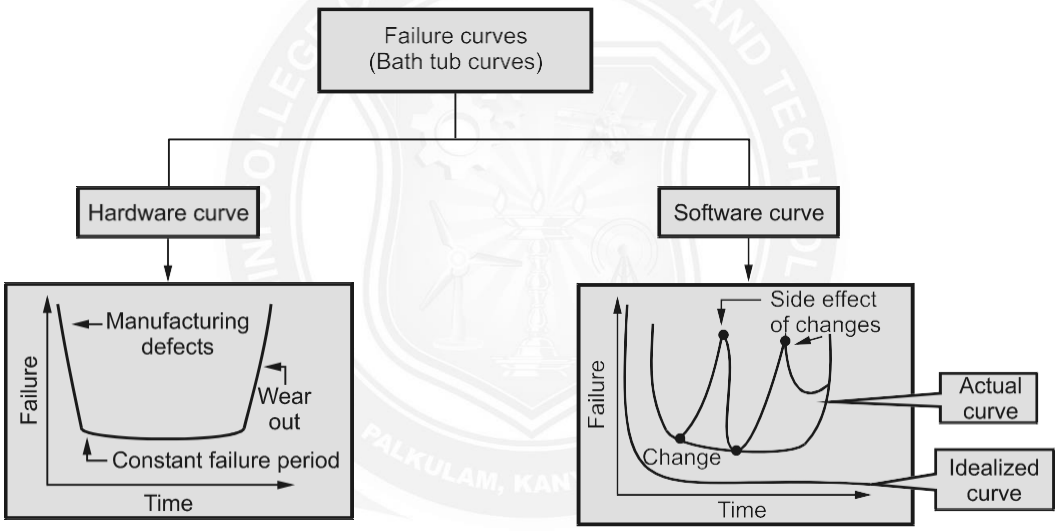
1. **Generic** - That means developed to be sold to a range of different customers.
2. **Custom** - That means developed for a single customer according to their specification.

Software Characteristics

Software development is a logical activity and therefore it is important to understandbasic characteristics of software. Some important characteristics of software are :

- **Software is engineered, not manufactured**
  - Software development and hardware development are two different activities.
  - A good design is a backbone for both the activities.
  - Quality problems that occur in hardware manufacturing phase can not be removed easily. On the other hand, during software development process such problems can be rectified.
  - In both the activities, developers are responsible for producing qualitative product.

- **Software does not ware out**
  - In early stage of hardware development process the failure rate is very high because of manufacturing defects. But after correcting such defects the failure rate gets reduced.
  - The failure rate remains constant for some period of time and again it starts increasing because of environmental maladies (extreme temperature, dusts, and vibrations).
  - On the other hand software does not get affected from such environmental maladies. Hence ideally it should have an *"idealized curve"*. But due to some **undiscovered errors** the failure rate is high and drops down as soon as the errors get corrected.
  - Hence in failure rating of software the *"actual curve"* is as shown below :



Failure curves for hardware and software

  - During the life of software if any change is made, some **defects** may get **introduced**.
  - This causes failure rate to be high. Before the curve can return to original steady state another change is requested and again the failure rate becomes high.
  - Thus the failure curve looks like a spike. Thus frequent changes in software cause it to deteriorate.
  - Another issue with software is that there are ***no spare parts for software***.
  - If hardware component wears out it can be replaced by another component but it is not possible in case of software.
  - Therefore **software maintenance** is **more difficult** than the hardware maintenance.

Most software is custom built rather than being assembled from components

- While developing any hardware product firstly the circuit design with desired functioning properties is created.
- Then required hardware components such as ICs, capacitors and registers are assembled according to the design, but this is not done while developing software product.
- Most of the software is custom built.
- However, now the software development approach is getting changed and we look for reusability of software components.
- It is practiced to reuse algorithms and data structures.
- Today software industry is trying to make library of reusable components.
- **For example :** In today's software, GUI is built using the reusable components such as message windows, pull down menus and many more such components. The approach is getting developed to use in-built components in the software. This stream of software is popularly known as *component engineering.*

Categories of Software

Software can be applied in a situation for which a predefined set of procedural steps (algorithm) exists. Based on a complex growth of software it can be classified into following categories.

- **System software** - It is collection of programs written to service other programs. Typical programs in this category are compiler, editors, and assemblers. The purpose of the system software is to establish a communication with the hardware.
- **Application software** - It consists of standalone programs that are developed for specific business need. This software may be supported by database systems.
- **Engineering/Scientific software** - This software category has a wide range of programs from astronomy to volcanology, from automative stress analysis to space shuttle orbital dynamics and from molecular biology to automated manufacturing. This software is based on complex numeric computations.
- **Embedded software** - This category consists of program that can reside within a product or system. Such software can be used to implement and control features and functions for the end-user and for the system itself.
- **Web applications** - Web application software consists of various web pages that can be retrieved by a browser. The web pages can be developed using programming languages like JAVA, PERL, CGI, HTML, DHTML.

- **Artificial Intelligence software** - This kind of software is based on knowledge based expert systems. Typically, this software is useful in robotics, expert systems, image and voice recognition, artificial neural networks, theorem proving and game playing.

Goals and Objectives of Software

While developing software following are common objectives.

1. **Satisfy users requirements** - Many programmers simply don't do what the end user wants because they do not understand user requirements. Hence it becomes necessary to understand the demand of end user and accordingly software should be developed.
2. **High reliability** - Mistakes or bugs in a program can be expensive in terms of human lives, money, and customer relation. For instance, Microsoft has faced many problems because earlier release of windows has many problems. Thus software should be delivered only if high reliability is achieved.
3. **Low maintenance costs** - Maintenance of software is an activity that can be done only after delivering the software to the customer. Any small change in software should not cause restructuring of whole software. This indicates that the design of software has poor quality.
4. **Delivery on time** - It is very difficult to predict the exact time on which the softwarecan be completed. But a systematic development of software can lead to meet the given deadline.
5. **Low production costs** - The software product should be cost effective.
6. **High performance** - The high performance software are expected to achieve optimization in speed and memory usage.
7. **Ease of reuse** - Use same software in different systems and software.

Environments reduce development costs and also improve the reliability. Hence reusability of developed software is an important property.

Difference between Software Product and Program

Sr. No.	Program	Software product
1.	Programs are developed by individual userand it is used for personal use.	Software product is developed by multiple users and it is used by large number of people or customers.
2.	Programs are small in size and possessinglimited functionality.	Software product consists of multiple program codes, related documents such as SRS, design documents user manuals, testcases and so on.
3.	Generally only one person uses the program, hence there is a lack of user interface.	Good graphical user interface is mos required by any software t product.
4.	Program is generally developed by **programme r**.	Software product is developed by **software engineers** who are large in number and work in a team. Therefore systematic approach of developing software product must be applied.
5.	**For example :** Program of sorting n elements.	**For example :** A word processing software.