**next →**

What is Software Engineering

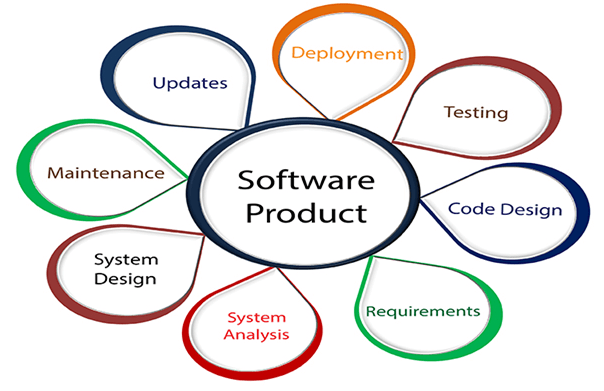
The term **software engineering** is the product of two words, **software**, and **engineering**.

The **software** is a collection of integrated programs. It consists of carefully-organized instructions written by developers with programming languages.

**Formally**, software is the implementation of algorithm by programming language with well defined data structure, documentation and long term support for user.

**Engineering** istheapplicationof**scientific**and**practical**knowledge **to invent, design, build, maintain,** and**improve frameworks, processes, etc.**

.



**Software Engineering** is an engineering branch related to the making and evolution of software product using well-defined scientific principles, techniques, and procedures. The result of software engineering is an effective and reliable software product.

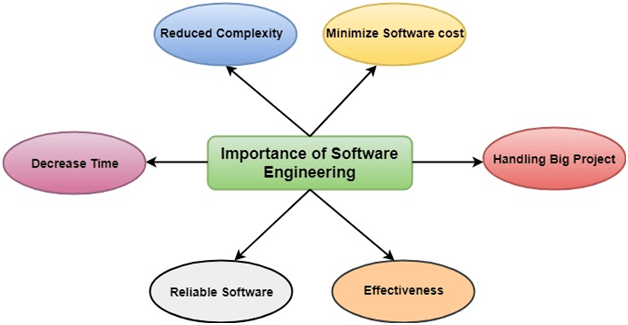
## Why is Software Engineering required?

Software engineering is necessary to become able to build software on time, on budget, within specification while minimizing defects, managing quality, cost, time, complexity, verifiability, maintainability, availability, reliability, dependability, performance, security, usability, and other attributes of software.

## Characteristics of a good software engineer

1. familiarity with software engineering principles.
2. Good technical knowledge of the project range (Domain knowledge).
3. Good programming abilities.
4. Good communication skills.
5. These skills comprise of oral, written, and interpersonal skills.
6. High motivation.
7. Sound knowledge of fundamentals of computer science.
8. Intelligence.
9. Ability to work in a team
10. Discipline,

## Importance of Software Engineering



**The importance of Software engineering is as follows:**

1. **Reduces complexity:** Big software is always complicated and challenging to progress. Software engineering has a great solution to reduce the complication of any project. Software engineering divides big problems into various small issues. And then start solving each small issue one by one. All these small problems are solved independently to each other.
2. **To minimize software cost:** Software needs a lot of hardwork and software engineers are highly paid experts. A lot of manpower is required to develop software with a large number of codes. But in software engineering, programmers project everything and decrease all those things that are not needed. In turn, the cost for software productions becomes less as compared to any software that does not use software engineering method.
3. **To decrease time:** Anything that is not made according to the project always wastes time. And if you are making great software, then you may need to run many codes to get the definitive running code. This is a very time-consuming procedure, and if it is not well handled, then this can take a lot of time. So if you are making your software according to the software engineering method, then it will decrease a lot of time.
4. **Handling big projects:** Big projects are not done in a couple of days, and they need lots of patience, planning, and management. And to invest six and seven months of any company, it requires heaps of planning, direction, testing, and maintenance. No one can say that he has given four months of a company to the task, and the project is still in its first stage. Because the company has provided many resources to the plan and it should be completed. So to handle a big project without any problem, the company has to go for a software engineering method.
5. **Reliable software:** Software should be secure, means if you have delivered the software, then it should work for at least its given time or subscription. And if any bugs come in the software, the company is responsible for solving all these bugs. Because in software engineering, testing and maintenance are given, so there is no worry of its reliability.
6. **Effectiveness:** Effectiveness comes if anything has made according to the standards. Software standards are the big target of companies to make it more effective. So Software becomes more effective in the act with the help of software engineering.

**Software Applications**

* system software
* real-time software
* business software
* engineering/scientific software
* embedded software
* PC software
* WebApps (Web applications)
* AI software

**Hardware is wear out but software is does not wear nout**

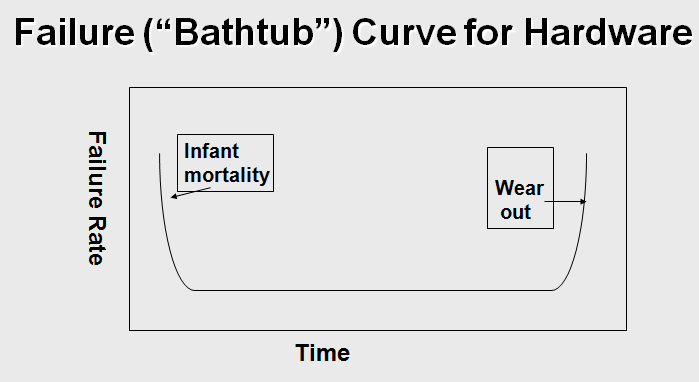
****

Figure 1 :Failure curve for hardware

failure rate as a function of time for hardware. The relationship, often called the "bathtub curve," indicates that hardware exhibits relatively high failure rates early in its life (these failures are often attributable to design or manufacturing defects); defects are corrected and the failure rate drops to a steady-state level (ideally, quite low) for some period of time. As time passes, however, the failure rate rises again as hardware components suffer from the cumulative affects of dust, vibration, abuse, temperature extremes, and many other environmental maladies. Stated simply, the hardware begins to wear out.

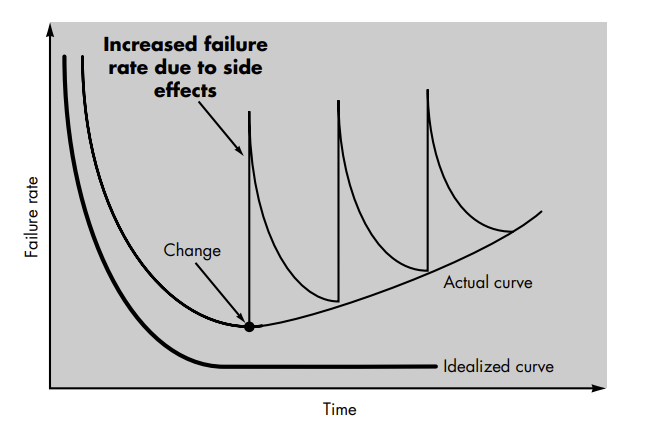
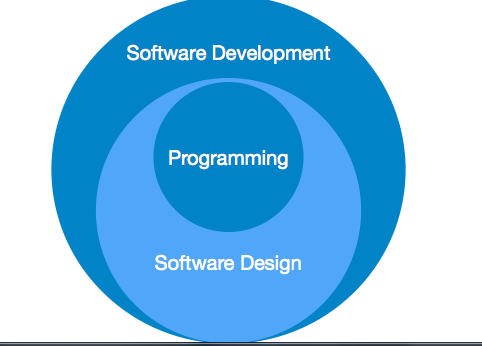


Fig : failure curves for software

Software is not susceptible to the environmental maladies that cause hardware to wear out. therefore, the failure rate curve for software should take the form of the “idealized curve”. Undiscovered defects will cause high failure rates early in the life of a program. However, these are corrected (ideally, without introducing other errors) and the curve flattens as shown.The idealized curve is a gross oversimplification of actual failure models for software. However, the implication is clear—software doesn't wear out.

**Software Paradigms**

Software paradigms refer to the methods and steps, which are taken while designing the software



* Requirement gathering
* Software design
* Programming

**Software Design Paradigm:** This paradigm is a part of Software Development and includes –

* Design
* Maintenance
* Programming

**Programming Paradigm:** This paradigm is related closely to programming aspect of software development. This includes –

* Coding
* Testing
* Integration