SOFTWARE ENGINEERING

What is Software?

Software is more than just a program code. A program is an executable code, which serves some computational purpose. Software is considered to be a collection of executable programming code, associated libraries and documentations. Software, when made for a specific requirement is called **software product.**

What is Engineering?

Engineering is all about developing products, using well-defined, scientific principles and methods.

Software Engineering?

Software engineering is an engineering branch associated with the development of software product using well-defined scientific principles, methods and procedures. The outcome of software engineering is an efficient and reliable software product.

IEEE defines software engineering as:

The application of a systematic, disciplined, quantifiable approach to the development, operation and maintenance of software.

Why Software Engineering?

- Without using software engineering principles it would be difficult to develop large programs.
- Complexity and difficulty levels of the programs increase exponentially with their sizes.
- Software engineering principles use two important techniques to reduce problem complexity:
  - *Abstraction*
  - *Decomposition*

Abstraction

- A problem can be simplified by omitting irrelevant details.
- Consider only those aspects of the problem that are relevant for certain purpose and suppress other aspects that are not relevant for the given purpose.
- Once the simpler problem is solved, then the omitted details can be taken into consideration to solve the next lower level abstraction, and so on.
- A powerful way of reducing the complexity of the problem.

Decomposition

- a complex problem is divided into several smaller problems and then the smaller problems are solved one by one.
- any random decomposition of a problem into smaller parts will not help.
- The problem has to be decomposed such that each component of the decomposed problem can be solved independently and then the solution of the different components can be combined to get the full solution.
- A good decomposition of a problem should minimize interactions among various components.
- If the different subcomponents are interrelated, then the different components cannot be solved separately and the desired reduction in complexity will not be realized.

Need of Software Engineering

- **Large software** - It is easier to build a wall than to a house or building, likewise, as the size of software become large engineering has to step to give it a scientific process.
- **Scalability**- If the software process were not based on scientific and engineering concepts, it would be easier to re-create new software than to scale an existing one.
- **Cost**- As hardware industry has shown its skills and huge manufacturing has lower down the price of computer and electronic hardware. But the cost of software remains high if proper process is not adapted.
- **Dynamic Nature**- The always growing and adapting nature of software hugely depends upon the environment in which the user works. If the nature of software is always changing, new enhancements need to be done in the existing one. This is where software engineering plays a good role.
- **Quality Management**- Better process of software development provides better and quality software product.

CHARACTERESTICS OF GOOD SOFTWARE

A software product can be judged by what it offers and how well it can be used.

This software must satisfy on the following grounds:

- **Operational**
- **Transitional**
- **Maintenance**

Operational

This tells us how well software works in operations. It can be measured on:

- **Budget**
- **Usability**
- **Efficiency**
- **Correctness**
- **Functionality**
- **Dependability**
- **Security**
- **Safety**

Transitional

This aspect is important when the software is moved from one platform to another:

- Portability
- Interoperability
- Reusability
- Adaptability

Maintenance

This aspect briefs about how well a software has the capabilities to maintain itself in the everchanging environment:

- **Modularity**
- **Maintainability**
- **Flexibility**
- **Scalability**