**UNIT 1**

# Introduction

**What is software?**

Software=Program + Documentation + Operating Procedures

**Importance of Software:**

**Software** plays an **important** role in managing business operations as per needs. With the help of **software**, we can manage and maintain business easily and eliminate human errors. Also, It is helpful in increasing productivity, efficiency and effectiveness of the organization activities.

**Software Crisis**

**It is often a case that software products:**

* Projects running over-budget.
* Projects running over-time.
* Software was very inefficient.
* Software was of low quality.
* Software often did not meet requirements.
* Projects were unmanageable and code difficult to maintain.
* Software was never delivered.

**What is Software Engineering?**

The term software engineering is composed of two words, software and engineering.

**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.**

**Engineering** on the other hand, is all about developing products, using well-defined, scientific principles and methods.

So, we can define **software engineering** as 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.*

We can alternatively view it as a systematic collection of past experience. The experience is arranged in the form of **methodologies and guidelines**. A small program can be written without using software engineering principles. But if one wants to develop a large software product, then software engineering principles are absolutely necessary to achieve a good quality software cost effectively.

Without using software engineering principles it would be difficult to develop large programs. In industry it is usually needed to develop large programs to accommodate multiple functions. A problem with developing such large commercial programs is that the complexity and difficulty levels of the programs increase exponentially with their sizes. Software engineering helps to reduce this programming complexity. Software engineering principles use two important techniques to reduce problem complexity: **abstraction and decomposition.** The principle of abstraction implies that a problem can be simplified by omitting irrelevant details. In other words, the main purpose of abstraction is to 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.

Abstraction is a powerful way of reducing the complexity of the problem. The other approach to tackle problem complexity is decomposition. In this technique, a complex problem is divided into several smaller problems and then the smaller problems are solved one by one. However, in this technique 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.

**Program vs. Software, and Software Engineering**

A **program** is an algorithm expressed using a precise notation which can be executed by a computer. In other words, a program is a sequence of instructions that tell the computer what to do. Software is not merely a collection of computer programs.

**Software** is a logical rather than a physical system element. Therefore, software has characteristics that are considerably different than those of hardware.

Software is: 

* Instructions (computer programs)—that when executed provide desired function and performance
* Data Structures—that enable programs to be adequately manipulate information, and 
* Documents—that describe the operation and use of the programs. IEEE defines software as the collection of computer programs, procedures, rules, and associated documentation and data.

Engineering is the analysis, design, construction, verification, and management of technical (or social) entities. [By a single entity we mean computer software]

IEEE defines Software Engineering as the systematic approach or a discipline to the development, operation, maintenance, and retirement of software. The work associated with software engineering can be categorized into three generic phases (i.e., definition, development, and support), regardless of application are, project size, or complexity each phase.

**NEED OF SOFTWARE ENGINEERING**

The need of software engineering arises because of higher rate of change in user requirements and environment on which the software is working.

**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

Well-engineered and crafted software is expected to have the following characteristics:

**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

In short, Software engineering is a branch of computer science, which uses well-defined engineering concepts required to produce efficient, durable, scalable, in-budget and on-time software products.