**Software Engineering**  
**Lecture # 2**

**Course Objective**

* To familiarize students to the fundamental concepts, techniques, processes, methods and tools of Software Engineering,
* To help students to develop basic skills that will enable them to construct software of high quality – software that is reliable, and that is reasonably easy to understand, modify and maintain
* To foster an understanding of why these skills are important

***Some Definitions***

***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 collection of executable programming code, associated libraries and documentations.

***Software Product:***

* Software, when made for a specific requirement is called **software product.**

***Software Engineering:***

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

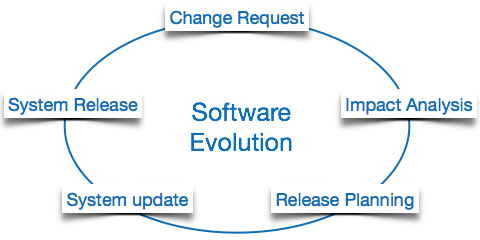
***IEEE* defines software engineering as:**

* The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software.

***Software Evolution:***

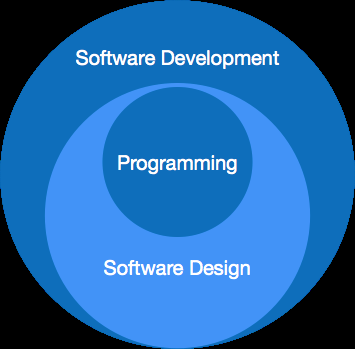
* The process of developing a software product using software engineering principles and methods is referred to as **Software Evolution.**
* This includes the initial development of software and its maintenance and updates, till desired software product is developed, which satisfies the expected requirements.

***Software Evolution (Cont...):***



***Software Paradigms (Cont...):***

* Software paradigms refer to the methods and steps, which are taken while designing the software.
* There are many methods proposed and are implemented.



***It consists of:***

* Requirement gathering
* Software design
* Programming

***Software Development Paradigm:***

* This paradigm is known as software engineering paradigms; where all the engineering concepts pertaining to the development of software are applied.
* It includes various researches and requirement gathering which helps the software product to build.

***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***

***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.

***Following are some of the needs stated:***

**Large software:**

It is easier to build a wall than a house or building, likewise, as the size of the software becomes 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, cost of the software remains high if proper process is not adapted.

***Dynamic Nature:***

Always growing and adapting nature of the 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 the software engineering plays a good role.

***Quality Management:***

Better process of software development provides better and quality software product.

***Characteristics 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:***

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

***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 the software has the capabilities to maintain itself in the ever-changing environment:

* Modularity
* Maintainability
* Flexibility
* Scalability

***What is CASE?***

* The acronym **CASE** stands for ***Computer-Aided Software Engineering***.
* It covers a wide range of different types of programs that are used to support software process activities such as requirements analysis, system modelling, debugging, verification, validation and testing.
* All methods now come with associated ***CASE*** technology such as editors for the notations used in the method, analysis modules which check the system model according to the method rules and report generators to help create system documentation.
* The ***CASE*** tools may also include a code generator that automatically generates source code from the system model and some process guidance for software engineers.