**Software Engineering And testing**:

**Module1**

**Software is**:

(1) Instructions (computer programs) that when executed provide desired features, function, and performance;

(2) Data structures that enable the programs to adequately manipulate information, and

(3) Descriptive information in both hard copy and virtual forms that describes the operation and use of the programs

**Definitions**

IEEE defines software engineering as:

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

*Fritz Bauer defined as:*

Software engineering is the establishment and use of sound engineering principles in order to obtain economically software that is reliable and works efficiently on real machines

**Characteristics**:

* Software is developed or engineered; it is not manufactured in the classical sense
* Software doesn’t “wear out.”
* Although the industry is moving toward component-based construction, most software continues to be custom built

**Domains which challenge the software engineers**:

* System software
* Application software
* Product-line software
* Web-apps
* Embedded software
* Engineering/scientific software
* Artificial intelligence software

**Legacy Software**

• Legacy software is older programs that are developed decades ago.

• The quality of legacy software is poor because it has inextensible design, convoluted code, poor and non-existent documentation, test cases and results that are not achieved.

**As time passes legacy systems evolve due to following reasons**:

• The software must be adapted to meet the needs of new computing environment or technology.

• The software must be enhanced to implement new business requirements.

• The software must be extended to make it interoperable with more modern systems or database

• The software must be re-architected to make it viable within a network environment

*“Software engineering a layered technology”*

1. The bedrock that supports software engineering is a **quality focus**.
2. The software engineering **process** is the glue that holds the technology layers together and enables rational and timely development of computer software.

Process defines a framework that must be established for effective delivery of software engineering technology

1. software methods provide the technical how-to’s for software development.
2. Software engineering tools provide automated or semi-automated support for the process and the methods

**Unique nature of web-App**:

* Network intensiveness.
* Concurrency
* Unpredictive loads
* Performance
* Availability
* Data driven
* Content sensitive
* Continuous evolution
* Immediacy
* Security
* Aesthetics

**Software Engineering challenges**:

* concerted effort should be made to understand the problem before a software solution is developed
* software should exhibit high quality
* software should be maintainable.

**Software process**:

* communication
* planning
* modelling
* construction
* deployment

**Umbrella activities**:

* software project tracking and control
* risk management
* software quality assurance
* technical reviews
* measurement
* software configuration and management
* reusability management
* work product preparation and production

**problem solving:**

1. understanding the problem
2. planning the solution
3. carry out the plan
4. examine the result

General principles:

* the reason it all exist
* kiss (keep it simple stupid)
* maintain the vision
* what you produce others will consume
* be open for the future
* plan ahead for reuse
* think

**Process Model**

**Process flow:**

* A linear process flow executes each of the five framework activities in sequence, beginning with communication and culminating with deployment
* An iterative process flow repeats one or more of the activities before proceeding to the next
* An evolutionary process flow executes the activities in a “circular” manner. Each circuit through the five activities leads to a more complete version of the software
* A parallel process flow executes one or more activities in parallel with other activities (e.g., modelling for one aspect of the software might be executed in parallel with construction of another aspect of the software).

**Process pattern**:

A process pattern describes a process-related problem that is encountered during software engineering work, identifies the environment in which the problem has been encountered, and suggests one or more proven solutions to the problem.

* Patterns can be defined at any level of abstraction.
* In some cases, a pattern might be used to describe a problem (and solution) associated with a complete process model (e.g., prototyping).
* In other situations, patterns can be used to describe a problem (and solution) associated with a framework activity (e.g., planning) or an action within a framework activity (e.g., project estimating)

Types of pattern:

1. Stage pattern
2. task pattern
3. phase pattern

**Process assessment and improvement**:

A number of different approaches to software process assessment and improvement have been proposed over the past few decades:

* Standard CMMI Assessment Method for Process Improvement (SCAMPI)—provides a five-step process assessment model that incorporates five phases: initiating, diagnosing, establishing, acting, and learning
* CMM-Based Appraisal for Internal Process Improvement (CBA IPI)— provides a diagnostic technique for assessing the relative maturity of a software organization; uses the SEI CMM as the basis for the assessment
* SPICE (ISO/IEC15504)—a standard that defines a set of requirements for software process assessment. The intent of the standard is to assist organizations in developing an objective evaluation of the efficacy of any defined software process [ISO08].
* ISO 9001:2000 for Software—a generic standard that applies to any organization that wants to improve the overall quality of the products, systems, or services that it provides.

**Perspective Process model:**

* The Waterfall Model or V model
* Incremental process model
* Evolutionary process model
* Spiral model
* Concurrent model
* Evolutionary process model

Specialised process model:

Specialized process models take on many of the characteristics of one or more of the traditional models presented in the preceding sections.

1. Component based model

* It incorporates many of the characteristics of the spiral model.
* It is evolutionary in nature [Nie92], demanding an iterative approach to the creation of software
* Modelling and construction activities begin with the identification of candidate components.
* These components can be designed as either conventional software modules or object-oriented classes or packages16 of classes.

Steps involved:

1. Available component-based products are researched and evaluated for the application domain in question.

2. Component integration issues are considered.

3. A software architecture is designed to accommodate the components.

4. Components are integrated into the architecture.

5. Comprehensive testing is conducted to ensure proper functionality

1. The Formal Methods Model:

* The formal methods model encompasses a set of activities that leads to formal mathematical specification of computer software.
* Formal methods enable you to specify, develop, and verify a computer-based system by applying a rigorous, mathematical notation
* Ambiguity, incompleteness, and inconsistency can be discovered and corrected more easily

its applicability in a business environment has been voiced:

* The development of formal models is currently quite time consuming and expensive.
* Because few software developers have the necessary background to apply formal methods, extensive training is required.
* It is difficult to use the models as communication mechanism for technically unsophisticated customers.

1. Unified process

* The inception phase.
* The elaboration phase
* The construction phase
* The transition phase.
* The production phase