What is Software Engineering?

- Engineering approach to develop software.
  - Building Construction Analogy.

- Systematic collection of past experience:
  - techniques,
  - methodologies,
  - guidelines.

IEEE Definition

- "Software engineering is 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 Crisis

- Software products:
  - fail to meet user requirements.
  - frequently crash.
  - expensive.
  - difficult to alter, debug, and enhance.
  - often delivered late.
  - use resources non-optimally.

Factors contributing to the software crisis

- Larger problems,
- Lack of adequate training in software engineering,
- Increasing skill shortage,
- Low productivity improvements.

Programs versus Software Products

• Usually small in size	• Large
• Author himself is sole user	• Large number of users
• Single developer	• Team of developers
• Lacks proper user interface	• Well-designed interface
• Lacks proper documentation	• Well documented & user-manual prepared
• Ad hoc development.	• Systematic development

Object-Oriented Design (80s)

- Object-oriented technique:
  - natural objects (such as employees, pay-roll-register, etc.) occurring in a problem are first identified.
- Relationships among objects:
  - such as composition, reference, and inheritance are determined.

Evolution of Other Software Engineering Techniques

- life cycle models,
- specification techniques,
- project management techniques,
- testing techniques,
- debugging techniques,
- quality assurance techniques,
- software measurement techniques,
- CASE tools, etc.

Differences between the exploratory style and modern software development practices

- Use of Life Cycle Models
- Software is developed through several well-defined stages:
  - requirements analysis and specification,
  - design,
  - coding,
  - testing, etc.

Differences between the exploratory style and modern software development practices

- Emphasis has shifted
  - from error correction to error prevention.
- Modern practices emphasize:
  - detection of errors as close to their point of introduction as possible.

Differences between the exploratory style and modern software development practices (CONT.)

- In exploratory style,
  - errors are detected only during testing,
- Now,
  - focus is on detecting as many errors as possible in each phase of development.

Differences between the exploratory style and modern software development practices (CONT.)

- During all stages of development process:
  - Periodic reviews are being carried out
- Software testing has become systematic:
  - standard testing techniques are available.

Differences between the exploratory style and modern software development practices (CONT.)

- Projects are being thoroughly planned:
  - estimation,
  - scheduling,
  - monitoring mechanisms.
- Use of CASE tools.

Life Cycle Model

- A software life cycle model (or process model):
  - a descriptive and diagrammatic model of software life cycle:
  - identifies all the activities required for product development,
  - establishes a precedence ordering among the different activities,
  - Divides life cycle into phases.

Why Model Life Cycle ?

- A written description:
  - forms a common understanding of activities among the software developers.
  - helps in identifying inconsistencies, redundancies in the development process.

Why Model Life Cycle ?

- Processes are tailored for special projects.
  - A documented process model
    - helps to identify where the tailoring is to occur.

Life Cycle Model (CONT.)

- The development team must identify a suitable life cycle model:
  - and then adhere to it.
  - Primary advantage of adhering to a life cycle model:
    - helps development of software in a systematic and disciplined manner.

Life Cycle Model (CONT.)

- When a software product is being developed by a team:
  - there must be a precise understanding among team members as to when to do what,
  - otherwise it would lead to and project failure.

Life Cycle Model (CONT.)

- A life cycle model:
  - defines entry and exit criteria for every phase.
  - A phase is considered to be complete:
    - only when all its exit criteria's are satisfied.

Life Cycle Model (CONT.)

- The phase exit criteria for the software requirements specification phase:
  - Software Requirements Specification (SRS) document is complete, reviewed, and approved by the customer.
- A phase can start:
  - only if its phase-entry criteria have been satisfied.

Life Cycle Model (CONT.)

- It becomes easier for software project managers:
  - to monitor the progress of the project.

Life Cycle Model (CONT.)

- Many life cycle models have been proposed.
- We will confine our attention to a few important and commonly used models.
  - classical waterfall model
  - iterative waterfall,
  - evolutionary,
  - prototyping, and
  - spiral model