Software Engineering & Information System Design

Course Code: CSE-3319

Lecture 1

What is Software Engineering

- Software
  - programs that provide function & performance
  - data structures for information manipulation
  - documents that describe the operations and use of the programs
- Engineering
  - A discipline that applies scientific and technical methods in the design and production of a product

Definition of Software Engineering

IEEE Definition:

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

Another Definition of Software Engineering

The practical application of scientific knowledge in the design and construction of computer programs and the associated documentation required to develop, operate, and maintain them. (Boehm)

Objectives of Software Engineering

- To improve quality of software products
- To increase customer satisfaction
- To increase productivity
- To increase job satisfaction

Software engineering is not programming. Programming is an important part of software engineering.

"This is not a programming course"

Software Characteristics

- Software is developed or engineered, not manufactured in the classical sense
- Software doesn't "wear out"
- Most software is custom-built, rather than being assembled from existing components

What Is A Good Software?

- Software is intangible
- Good software is subjective
- Some qualities that are used to assess software:
  - Correctness: a program satisfies its specifications.
  - Reliability: a program satisfies its intended functions.
  - Usability: the effort required to learn, operate, prepare input, and interpret the output.
  - Integrity: Control of access to unauthorized persons.

What Is A Good Software continue?

- Efficiency: amount of computing resources required.
- Maintainability: effort required to locate and fix errors in an operational programs.
- Portability: required effort to transfer a program from one hardware/software environment to another.
- Testability: required effort to test a program to ensure its performing its intended functions.
- Interoperability: effort required to couple programs.
- Reusability: reuse of programs in other applications.

Software Applications

- System Software
- Real-time Software
- Business Software
- Engineering & Scientific Software
- Embedded Software
- Personal Computer Software
- Artificial Intelligence Software

Challenges

- Why does it take so long to get software finished?
- Why are the development costs so high?
- Why can't we find all errors?
- Why do we spend so much time and effort maintaining existing programs?
- Why is it difficult to measure progress?

Topics

- Software Life Cycle Models
- Requirements Engineering
- Software Specification
- Software Design and Architecture
- Software Metrics
- Software Quality Assurance
- Agile
- Software Maintenance and Evolution
- Software Project Management