Lecture I:

Introduction

General Views/expectations/opinions

We shall require a substantially new manner of thinking if mankind is to survive.

Albert Einstein

Software engineering: Basic definitions

- What is software?
- What is software engineering?
- Why software engineering?
- What is the difference between software engineering and system engineering?
- What is a software process?
- What is a process software model?
- What are the costs of software engineering?

Software engineering: Basic definitions cont’d

- What approach was used before and what is being used now?
- What are software engineering methods?
- What is CASE (Computer-Aided Software Engineering)?
- What are the attributes of good software?
- What are the key challenges facing software engineering?

What is software?

- Computer programs and associated documentation such as requirements, design models and user manuals.

- Software products may be developed for a particular customer or may be developed for a general market.

- Software products may be:
  - Generic - developed to be sold to a range of different customers e.g. PC software such as Excel or Word.
  - Bespoke (custom) - developed for a single customer according to their specification.
- New software can be created by developing new programs, configuring generic software systems or reusing existing software.

What is software? Cont’d

- Software could also refer to the instructions that tell the computer what to do.
- Software is also a general term for the various kinds of programs used to operate computers and related devices. It can be thought of as the variable part of a computer.
- Software is often divided into;
  - Application software (programs that do the work that users are directly interested in - ***for example??***).
  - System software (programs that help the integration of all the constituent components of the computer into a system – ***for example??***).
  - Utilities (programs that are used to perform routine operations in a computer - ***for example??***).

What is software engineering?

- The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software.
- The art and science of creating huge, complex software systems by a team of programmers (more than one), over a relatively long period of time (sometimes years), within given constraints (time, with emphasis on particular features as per user needs) and within a limited budget.
- The use of techniques, methods, and methodologies to develop high quality software that has a set of desirable attributes.

What is software engineering? Cont'd

- Software engineering is an engineering discipline that is concerned with all aspects of software production.

- Software engineers should adopt a systematic and organised approach to their work and use appropriate tools and techniques depending on the problem to be solved, the development constraints and the resources available.

Why software engineering?

- Software engineering in its pure and formal approach is important in that;
  - It makes it easier to manage software development (in many ways cost management, team management, quality assurance management, schedule management for timely release).
  - It enables the reusability and duplication of experience, effort and even some deliverables in the project.
  - It enables the creation of software standards for development efforts across many platforms.
  - Raises software quality.

What is the difference between software engineering and system engineering?

- System engineering is concerned with all aspects of computer-based systems development including hardware, software and process engineering.

- Software engineering is part of this process concerned with developing the software infrastructure, control, applications and databases in the system.

- System engineers are involved in system specification, architectural design, integration and deployment.

What is a software process?

- A set of activities whose goal is the development or evolution of software.

- Generic phases in all software processes are:
  - Requirements phase
  - Specification phase
  - Planning phase
  - Design phase
  - Implementation phase
  - Integration phase
  - Maintenance phase

What is a software process model?

- A simplified representation of a software process, presented from a specific perspective.

- Examples of process perspectives are
  - Workflow perspective - sequence of activities;
  - Data-flow perspective - information flow;
  - Role/action perspective - who does what.

- Generic process models
  - Waterfall;
  - Iterative development;
  - Component-based software engineering.

What are the costs of software engineering?

- Costs vary depending on the type of system being developed and the requirements of system attributes such as performance and system reliability.

- Distribution of costs depends on the development model that is used.

What approaches were used before and what is used to date?

- The traditional approach
  - The development efforts were not structured - sometimes it could be done by an individual single-handedly, and at other times by a group of programmers.
  - There were no metrics whosoever of the software - what it cost, how much effort was needed, approx. time it would take to develop, its quality and how it was being addressed, how well it worked for users etc.
  - Deployment was freestyle.

What approaches were used before and what is used to date? Cont'd

- Bugs took a lot of time to be identified and fixed
- Software was released/deployed without any timelines and with no organizational impact assessment.
- Documentation was poor, if any existed.
- Maintenance was poor, and the software had a poor lifecycle if any.
- Due to the free-style approach, it was very difficult to reuse code.

What approaches were used before and what is used to date? Cont'd

- The software engineering approach
  - The software engineering approach exhibits the following attributes;
    - The software development efforts are spearheaded by **TEAMS** (composed of programmers, Systems analysts, Project manager, Database Designers etc).
    - Users are consulted time and again regarding features and functionalities they desire in the new software.
    - The development efforts are very structured - each team player has a very distinct and unique role to play in the whole exercise.
    - There are a number of metrics, and there are personnel in the teams to look out for such - what it will cost and whether the project is within budget, how much effort will be needed (i.e. the number of personnel), approx. time it will take to develop, product quality and how it was being addressed, how well it worked for users etc

What approaches were used before and what is used to date? Cont'd

- Deployment is very much structured - every implementation aspect is scheduled; from training user needs, installation of databases, test runs etc.
- Bugs can be quickly identified and fixed since the system's quality and performance is measured against already defined user needs.
- Software products are released within predetermined timelines and with lots of organizational impact assessment.
- Documentation is a must; it is a continuous process, which gives a snapshot of the whole development of the software.
- Maintenance is very easy and systematic.
- Due to the formal approach, it is very easy to reuse code.

What are software engineering methods?

- Structured approaches to software development which include system models, notations, rules, design advice and process guidance.

- Model descriptions
  - Descriptions of graphical models which should be produced;
- Rules
  - Constraints applied to system models;
- Recommendations
  - Advice on good design practice;
- Process guidance
  - What activities to follow.

What are CASE tools? (Computer-Aided Software Engineering tools)?

- Software systems that are intended to provide automated support for software process activities.
- CASE tools are often used for method support.
- Upper-CASE tools
  - Support the early process activities of requirements and design.
- Lower-CASE tools
  - Support later activities such as programming, debugging and testing.

What are the attributes of good software?

- The software should deliver the required functionality and performance to the user and should be maintainable, dependable and acceptable.

- Maintainability
  - Software must evolve to meet changing needs;
- Dependability
  - Software must be trustworthy;
- Efficiency
  - Software should not make wasteful use of system resources;
- Acceptability
  - Software must be accepted by the users for which it was designed. This means it must be understandable, usable and compatible with other systems.

What are the key challenges facing software engineering?

- Heterogeneity
  - Developing techniques for building software that can cope with heterogeneous platforms and execution environments;
- Delivery
  - Developing techniques that lead to faster delivery of software;
- Trust
  - Developing techniques that demonstrate that software can be trusted by its users.

Professional and ethical responsibility

- Software engineering involves wider responsibilities than simply the application of technical skills.
- Software engineers must behave in an honest and ethically responsible way if they are to be respected as professionals.
- Ethical behaviour is more than simply upholding the law.

Issues of professional responsibility

- Confidentiality
  - Engineers should normally respect the confidentiality of their employers or clients irrespective of whether or not a formal confidentiality agreement has been signed.
- Competence
  - Engineers should not misrepresent their level of competence. They should not knowingly accept work which is out with their competence.
- Intellectual property rights
  - Engineers should be aware of local laws governing the use of intellectual property such as patents, copyright, etc. They should be careful to ensure that the intellectual property of employers and clients is protected.
- Computer misuse
  - Software engineers should not use their technical skills to misuse other people's computers, e.g. game playing on an employer's machine to dissemination of viruses.

ACM/IEEE Code of Ethics

- The professional societies in the US have cooperated to produce a code of ethical practice.
- Members of these organisations sign up to the code of practice when they join.
- The Code contains eight principles related to the behaviour of and decisions made by professional software engineers, including practitioners, educators, managers, supervisors and policy makers, as well as trainees and students of the profession.

Code of ethics - preamble

- The short version of the code summarizes aspirations at a high level of the abstraction; the clauses that are included in the full version give examples and details of how these aspirations change the way we act as software engineering professionals. Without the aspirations, the details can become legalistic and tedious; without the details, the aspirations can become high sounding but empty; together, the aspirations and the details form a cohesive code.

- Software engineers shall commit themselves to making the analysis, specification, design, development, testing and maintenance of software a beneficial and respected profession. In accordance with their commitment to the health, safety and welfare of the public, software engineers shall adhere to the following Eight Principles:

Code of ethics - principles

- PUBLIC
  - Software engineers shall act consistently with the public interest.
- CLIENT AND EMPLOYER
  - Software engineers shall act in a manner that is in the best interests of their client and employer consistent with the public interest.
- PRODUCT
  - Software engineers shall ensure that their products and related modifications meet the highest professional standards possible.
- JUDGMENT
  - Software engineers shall maintain integrity and independence in their professional judgment.

Code of ethics - principles

- MANAGEMENT
  - Software engineering managers and leaders shall subscribe to and promote an ethical approach to the management of software development and maintenance.
- PROFESSION
  - Software engineers shall advance the integrity and reputation of the profession consistent with the public interest.
- COLLEAGUES
  - Software engineers shall be fair to and supportive of their colleagues
- SELF
  - Software engineers shall participate in lifelong learning regarding the practice of their profession and shall promote an ethical approach to the practice of the profession.

Ethical dilemmas (Groups)

- Disagreement in principle with the policies of senior management???

- Your employer acts in an unethical way and releases a ***safety-critical system*** **without** finishing the testing of the system???

- Participation in the development of military weapons systems or nuclear systems???