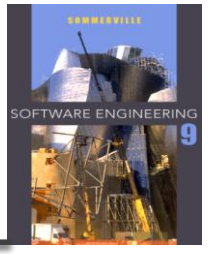
Chapter 1- Introduction

Lecture 1

Software engineering



- The economies of ALL developed nations are dependent on software.
- More and more systems are software controlled.
- Software engineering is concerned with theories, methods and tools for professional software development.
- Expenditure on software represents a significant fraction of GNP in all developed countries.

Software costs

- Software costs often dominate computer system costs. The costs of software on a PC are often greater than the hardware cost.
- Software costs more to maintain than it does to develop. For systems with a long life, maintenance costs may be several times development costs.
- Software engineering is concerned with cost-effective software development.

Software products

- Generic products
  - Stand-alone systems that are marketed and sold to any customer who wishes to buy them.
  - Examples – PC software such as graphics programs, project management tools; CAD software; software for specific markets such as appointments systems for dentists.
- Customized products
  - Software that is commissioned by a specific customer to meet their own needs.
  - Examples – embedded control systems, air traffic control software, traffic monitoring systems.

Product specification

- Generic products
  - The specification of what the software should do is owned by the software developer and decisions on software change are made by the developer.
- Customized products
  - The specification of what the software should do is owned by the customer for the software and they make decisions on software changes that are required.

Frequently asked questions about software engineering

Question	Answer
What is software?	Computer programs and associated documentation. Software products may be developed for a particular customer or may be developed for a general market.
What are the attributes of good software?	Good software should deliver the required functionality and performance to the user and should be maintainable, dependable and usable.
What is software engineering?	Software engineering is an engineering discipline that is concerned with all aspects of software production.
What are the fundamental software engineering activities?	Software specification, software design, software development, software validation and software evolution.
What is the difference between software engineering and computer science?	Computer science focuses on theory and fundamentals; software engineering is concerned with the practicalities of developing and delivering useful software.
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 more general process. Systems engineering also involves specifying, building, **maintaining and supporting technical infrastructure.**

Frequently asked questions about software engineering

Question	Answer
What are the key challenges facing software engineering?	Coping with increasing diversity, demands for reduced delivery times and developing trustworthy software.
What are the costs of software engineering?	Roughly 60% of software costs are development costs, 40% are testing costs. For custom software, evolution costs often exceed development costs.
What are the best software engineering techniques and methods?	While all software projects have to be professionally managed and developed, different techniques are appropriate for different types of system. For example, games should always be developed using a series of prototypes whereas safety critical control systems require a complete and analyzable specification to be developed. You can't, therefore, say that one method is better than another.
What differences has the web made to software engineering?	The web has led to the availability of software services and the possibility of developing highly distributed service-based systems. Web-based systems development has led to important advances in programming languages and software reuse.

Essential attributes of good software

Product characteristic	Description
Maintainability	Software should be written in such a way so that it can evolve to meet the changing needs of customers. This is a critical attribute because software change is an inevitable requirement of a changing business environment.
Dependability and security	Software dependability includes a range of characteristics including reliability, security and safety. Dependable software should not cause physical or economic damage in the event of system failure. Malicious users should not be able to access or damage the system.
Efficiency	Software should not make wasteful use of system resources such as memory and processor cycles. Efficiency therefore includes responsiveness, processing time, memory utilisation, etc.
Acceptability	Software must be acceptable to the type of users for which it is designed. This means that it must be understandable, usable and compatible with other systems that they use.

Software engineering

- Software engineering is an engineering discipline that is concerned with all aspects of software production from the early stages of system specification through to maintaining the system after it has gone into use.
- Engineering discipline
  - Using appropriate theories and methods to solve problems bearing in mind organizational and financial constraints.
- All aspects of software production
  - Not just technical process of development. Also project management and the development of tools, methods etc. to support software production.

Importance of software engineering

- More and more, individuals and society rely on advanced software systems. We need to be able to produce reliable and trustworthy systems economically and quickly.
- It is usually cheaper, in the long run, to use software engineering methods and techniques for software systems rather than just write the programs as if it was a personal programming project. For most types of system, the majority of costs are the costs of changing the software after it has gone into use.

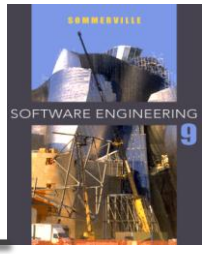
Software process activities

- Software specification, where customers and engineers define the software that is to be produced and the constraints on its operation.
- Software design and development, where the software is designed and programmed.
- Software validation, where the software is checked to ensure that it is what the customer requires.
- Software evolution, where the software is modified to reflect changing customer and market requirements.

Software engineering fundamentals

- Some fundamental principles apply to all types of software system, irrespective of the development techniques used:
  - Systems should be developed using a managed and understood development process. Of course, different processes are used for different types of software.
  - Dependability and performance are important for all types of system.
  - Understanding and managing the software specification and requirements (what the software should do) are important.
  - Where appropriate, you should reuse software that has already been developed rather than write new software.

Key points



- Software engineering is an engineering discipline that is concerned with all aspects of software production.
- Essential software product attributes are maintainability, dependability and security, efficiency and acceptability.
- The high-level activities of specification, development, validation and evolution are part of all software processes.
- The fundamental notions of software engineering are universally applicable to all types of system development.

Key points

- There are many different types of system and each requires appropriate software engineering tools and techniques for their development.
- The fundamental ideas of software engineering are applicable to all types of software system.

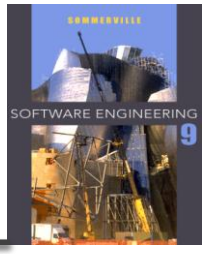
Chapter 1- Introduction

Lecture 2

Software engineering ethics

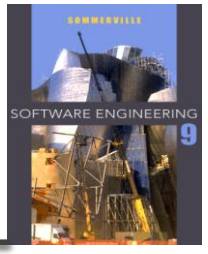
- Software engineering involves wider responsibilities than simply the application of technical skills.
- Software engineers must behave in an honest and ethically responsible way if they are to be respected as professionals.
- Ethical behaviour is more than simply upholding the law but involves following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

Issues of professional responsibility



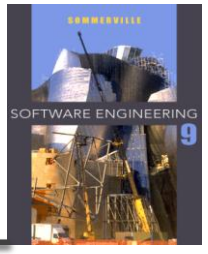
- Confidentiality
  - Engineers should normally respect the confidentiality of their employers or clients irrespective of whether or not a formal confidentiality agreement has been signed.
- Competence
  - Engineers should not misrepresent their level of competence. They should not knowingly accept work which is outwith their competence.

Issues of professional responsibility



- ✧ 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. Computer misuse ranges from relatively trivial (game playing on an employer's machine, say) to extremely serious (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.

Rationale for the code of ethics

- *Computers have a central and growing role in commerce, industry, government, medicine, education, entertainment and society at large. Software engineers are those who contribute by direct participation or by teaching, to the analysis, specification, design, development, certification, maintenance and testing of software systems.*
- *Because of their roles in developing software systems, software engineers have significant opportunities to do good or cause harm, to enable others to do good or cause harm, or to influence others to do good or cause harm. To ensure, as much as possible, that their efforts will be used for good, software engineers must commit themselves to making software engineering a beneficial and respected profession.*

The ACM/IEEE Code of Ethics

Software Engineering Code of Ethics and Professional Practice

ACM/IEEE-CS Joint Task Force on Software Engineering Ethics and Professional Practices

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:

Ethical principles

1. PUBLIC - Software engineers shall act consistently with the public interest.
2. 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.
3. PRODUCT - Software engineers shall ensure that their products and related modifications meet the highest professional standards possible.
4. JUDGMENT - Software engineers shall maintain integrity and independence in their professional judgment.
5. MANAGEMENT - Software engineering managers and leaders shall subscribe to and promote an ethical approach to the management of software development and maintenance.
6. PROFESSION - Software engineers shall advance the integrity and reputation of the profession consistent with the public interest.
7. COLLEAGUES - Software engineers shall be fair to and supportive of their colleagues.
8. 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

- 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.

PROJECT

Academic Integrity & Professional Practice

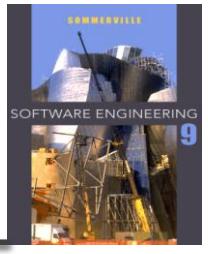
Software Engineering is a collaborative activity. You are encouraged to work together, but ...

- Some tasks may require individual work.
- Always give credit to your sources and collaborators.

Good professional practice: To make use of the expertise of others and to build on previous work, **with proper attribution**.

Unethical and academic plagiarism: To use the efforts of others **without attribution**.

Grading Proposal….



- Final exam 40 %
- Project 25%
- Midterm 30 %
- Lab: 5 %

Feedback about the Group Projects

Comments on the group projects

Your feedback about what is working well on the project and where you see difficulties: to help anticipate problems early.

Feedback about the contribution of team members

Your feedback about how each member of your team contributed to the work of the group: to identify those individuals who make extra effort or do not contribute fully.

Project

The course is built around the project

- Select your own project, any branch of software development, if you have difficulties, contact me (on time!).
- **Project task 1: Form teams of 4-5 students.**
- Teams need many strengths -- organizational, technical, writing, etc.
- Consider appointing a leader to coordinate the effort, or a separate leader for each of the four assignments.
- **Project task 2: Project Proposal**
- **Project task 3: Project pitch:** Can be a ppt presentation, but not required. **Each team will have 8-10 min to present their idea.**
- Group presentations:

  project proposal, 1st iteration, (if time allows 2nd iteration), final presentations

We will be discussing projects in class regularly

Overall Aim of the Course

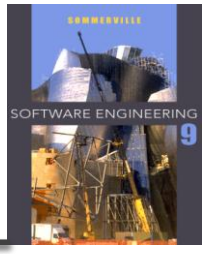
Assumption that you are technically proficient. You know a good deal about computing, can program reasonably, can learn more on the job.

When you leave IUS, you are going to work on production projects where success or failure may cost millions of dollars.

Soon you will be in charge. It may be your money.

Idea is to make your mistakes now and learn from your mistakes.

Variety of Software Products



Examples

Real time:	(air) traffic control
Embedded systems:	digital camera, GPS, iPod
Data processing:	telephone billing, pensions
Information systems:	web sites, digital libraries
Sensors:	weather data
System software:	operating systems, compilers
Communications:	routers, telephone switches
Offices:	word processing, video conferences
Scientific:	simulations, weather forecasting
Graphical:	film making, design

etc., etc., etc., ...

Software is Expensive

Software is expensive.

The major costs are salaries (your salaries)!

Every software project has a trade-off between:

- Functionality
- Resources (cost)
- Timeliness

Client

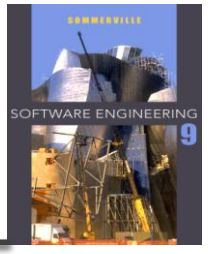
Client (a.k.a. Customer)

The client provides resources and expects some product in return.

The client is often a member of the organization that is providing the money. The client's job success depends on the success of the software project.

Client satisfaction is a primary measurement of success in a software project.

Software is Risky



How can you manage risks?

- ***Much of software is never used (perhaps 50%)***
- ***Most software development projects have major problems***

What is the penalty if software is:

late?
over budget?
does not work or full of bugs?

Examples:

car anti-lock brakes (no bugs allowed)
web browser in cell phone (no delays in release allowed)

Software is Risky

Most software projects fail because the software developers build the wrong software!

Developers start implementing the system:

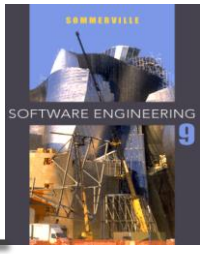
without determining whether they are building what the customer really wants.

It is important

- Learn requirements analysis and specification techniques thoroughly
- Understand what the client expects of the software
- Understand what the client's organization expects of the client
- Add technical insights and suggestions, but remember:

Client satisfaction is the primary measurement of success in a software project.

Teams



Most software development is by **teams**

- Effectiveness of team determines success

Most large software projects are **built on older ones**

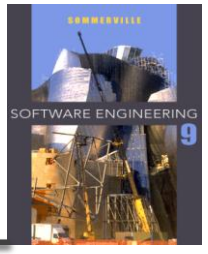
- It is rare to start a new suite of programs from scratch
- Building on the work of others is a fundamental skill of software development

Previous Experience (Yours)

Your background

- Biggest program that you have written?
- Biggest program that you have worked on?
- Biggest project team that you have been part of?
- Longest project that you have worked on?
- Most people who have used your work?
- Longest that your project has been in production?

Future Experience



What will you be doing one year from now?

Ten years from now?

Typical career paths in computing combine technical work with varying degrees of project management, marketing, entrepreneurship, etc.

Risk

Risk (as Seen by a Manager)

- **Problems**

  Over budget
  Late delivery
  Does not work as expected

- **Never used**

  Does the wrong thing
  Needs change
  Users dislike to use it
  etc.

Failures of software development projects can bankrupt companies

Visibility

Visibility (as Seen by a Manager)

- **Problem**

  Must rely on others for reports of progress or difficulties

- **Software Developers**

  Have difficulty evaluating progress
  Optimistic
  Consider reporting a waste time
  etc.

The people who take the responsibility must know what is happening

You will make regular progress reports on your projects

Due next week

Project proposal: Due 2rd March 2022. To be signed by all team members.
+
Project pitch (9th march 2022). Each team has 8 – 10 min to present their idea.

Project proposal should provide a description of the project.

•Describe a software product that you would like to order as a customer.
•Make sure that you include what the problem does, i.e. what is the software intended to solve, and why do you believe it would be important to build it (i.e. what real-world problem would it solve.)
• Also include, in bulleted list, list some functionalities the software will have (i.e. *what* must the developers implement to enable users to accomplish their tasks),
• as well as related work (briefly) including references.
•Preferred implementation technology
• The proposal should be min 350, max 700 words. Font size:12 , Font type: Times New Roman, Arial or Calibri, single spaced