CS 410/510 - Software Engineering

What is Software Engineering?

Reference: Sommerville, Software Engineering, 10 ed., Chapter 1

In theory, there is no difference between theory and practice. But, in practice, there is.

Why do we need software engineering?

Software and software systems are everywhere. 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.

Applying software engineering principles helps address the two main factors of software failures:

Increasing demands
: As new software engineering techniques help us to build larger, more complex systems, the demands change. Systems have to be built and delivered more quickly; larger, even more complex systems are required; systems have to have new capabilities that were previously thought to be impossible.

Low expectations
: It is relatively easy to write computer programs without using software engineering methods and techniques. Computer programming is NOT software engineering.

FAQ about software engineering

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 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.

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

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: a definition

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. It is an engineering discipline because it uses appropriate theories and methods to solve problems bearing in mind organizational and financial constraints. Software engineering focuses on all aspects of software production and not just on the technical process of development; it includes project management and the development of tools, methods etc. to support software production.

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.

Any software process includes four types of activities:

- Software **specification**, where customers and engineers define the software that is to be produced and the constraints on its operation.
- Software **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.

General issues affecting most software

Heterogeneity

Increasingly, systems are required to operate as distributed systems across networks that include different types of computer and mobile devices.

Business and social change

Business and society are changing incredibly quickly as emerging economies develop and new technologies become available. They need to be able to change their existing software and to rapidly develop new software.

Security and trust

As software is intertwined with all aspects of our lives, it is essential that we can trust that software.

Software engineering fundamentals

These software engineering fundamentals that apply to all types of software system:

Software process

Systems should be developed using a managed and understood development process. The organization developing the software should plan the development process and have clear ideas of what will be produced and when it will be completed. Of course, different processes are used for different types of software.

Focus on reliability

Dependability and performance are important for all types of systems. Software should behave as expected, without failures and should be available for use when it is required. It should be safe in its operation and, as far as possible, should be secure against external attack. The system should perform efficiently and should not waste resources.

Importance of requirements

Understanding and managing the software specification and requirements (what the software should do) are important. You have to know what different customers and users of the system expect from it and you have to manage their expectations so that a useful system can be delivered within budget and to schedule.

Leverage software reuse

You should make as effective use as possible of existing resources. This means that, where appropriate, you should reuse software that has already been developed rather than write new software.

Software engineering ethics

Some issues of professional responsibility:

Confidentiality

You should normally respect the confidentiality of your employers or clients irrespective of whether or not a formal confidentiality agreement has been signed.

Competence

You should not misrepresent your level of competence. You should not knowingly accept work that is outside your competence.

Intellectual property rights

You should be aware of local laws governing the use of intellectual property such as patents and copyright. You should be careful to ensure that the intellectual property of employers and clients is protected.

Computer misuse

You should not use your 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 or other malware).

Software Engineering Code of Ethics and Professional Practice