
CMSC 435: Software Engineering

Course overview

CMSC 435 - 1

Topics covered today

- Course requirements
- FAQs about software engineering
- Professional and ethical responsibility

CMSC 435 - 2

Course Objectives

- To introduce software engineering and to explain its importance in building large programs
- To understand the process of developing new technology and the role of experimentation
- To set out the answers to key questions about software engineering
- To introduce ethical and professional issues and to explain why they are of concern to software engineers

CMSC 435 - 3

CMSC 435 requirements

- Exam 1 (20%) and Exam 2 (20%)
- Project (40%)
- Report (10%)
- Other (homework, presentations, ...) (10%)
- Project goal is to take a large system and add several features to it:
 - System is a prototype of an FAA air traffic control system called TSAFE.
 - You have to make decisions on the implementation and test your results
 - Projects are group activities; system written in Java.
 - For EACH phase of the project you will be in a different group, randomly assigned
- Use class website.
(www.cs.umd.edu/~mvz/cmsc435-s09/)

CMSC 435 - 4

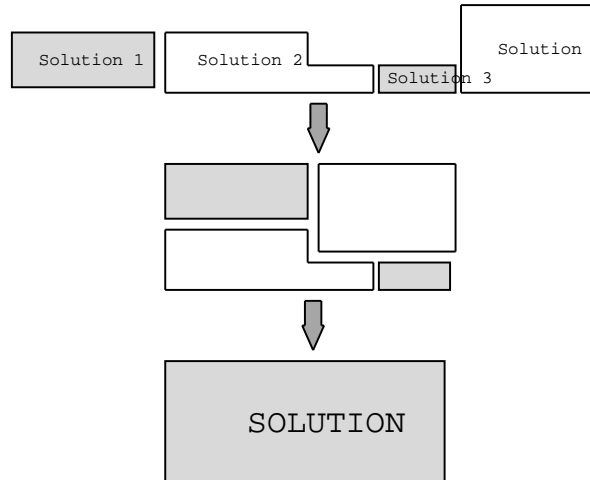
Software engineering

- 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.

What is software engineering?

"... the systematic and regular application of scientific and mathematical knowledge to the design, construction, and operation of machines, systems, and so on of practical use and, hence, of economic value. Particular characteristic of engineers is that they take seriously their responsibility for correctness, suitability, and safety of the results of their efforts. In this regard they consider themselves to be responsible to their customer (including their employers where relevant), to the users of their machines and systems, and to the public at large."

Two aspect of Software Engineering: Synthesis



CMSC 435 - 9

Software costs

- Software costs often dominate computer system costs. The costs of software on a PC are always 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.

CMSC 435 - 10

Why is it 435 and not 135?

- Need experience is building big programs. (435 is not a programming course.) Learn how to work in groups.
- Every student starts Computer Science 1 "knowing" everything. Previous generations had problems only because they were too sloppy or not smart enough.
- You have to experience the 4am panic to finish a project due 8am to really understand the problems with software development.
- By now you should understand that there **MUST** be a better way to do this.

CMSC 435 - 11

FAQs about software engineering

- What is software?
- What is software engineering?
- What is the difference between software engineering and computer science?
- What is the difference between software engineering and system engineering?
- What is a software process?
- What is a software process model?

CMSC 435 - 12

FAQs about software engineering

- What are the costs of software engineering?
- 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.
 - ❑ 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**. (Reuse is emphasis this semester)

What is software engineering?

- 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.

What is the difference between software engineering and computer science?

- Computer science is concerned with theory and fundamentals; software engineering is concerned with the practicalities of developing and delivering useful software.
- Computer science theories are still insufficient to act as a complete underpinning for software engineering (unlike e.g. physics and electrical engineering).

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.

CMSC 435 - 17

What is a software process?

- A set of activities whose goal is the development or evolution of software.
- Generic activities in all software processes are:
 - **Specification** - what the system should do and its development constraints
 - **Development** - production of the software system
 - **Validation** - checking that the software is what the customer wants
 - **Evolution** - changing the software in response to changing demands.

CMSC 435 - 18

Typical development processes

- Requirements engineering
- System analysis
- High-level design/architecture
- Low-level design
- Coding
- Integration
- Design and code reviews
- Testing
- Maintenance
- Project management
- Configuration management

CMSC 435 - 19

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;
 - Agile development;
 - Component-based software engineering.

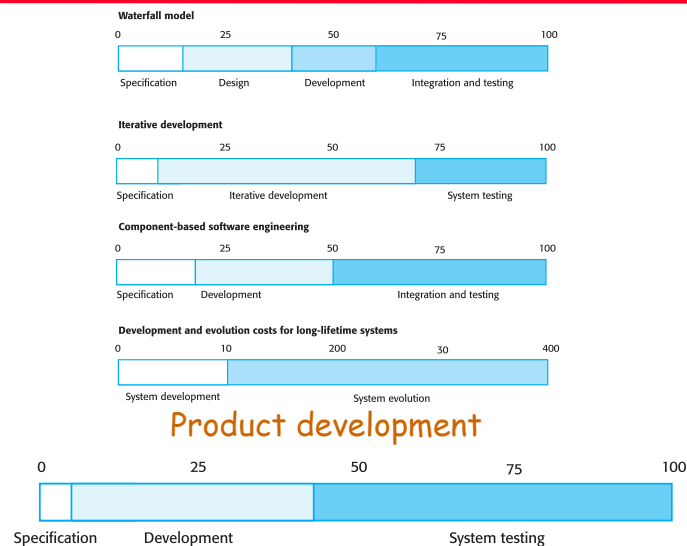
CMSC 435 - 20

What are the costs of software engineering?

- Roughly 60% of costs are development costs, 40% are testing costs. For custom software, evolution costs often exceed development costs.
- 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.

CMSC 435 - 21

Activity cost distribution



CMSC 435 - 22

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.

CMSC 435 - 23

What is CASE (Computer-Aided Software Engineering)

- Software systems that are intended to provide automated support for software process activities.
- CASE systems are often used for method support.
- Upper-CASE ("Upstream")
 - Tools to support the early process activities of requirements and design;
- Lower-CASE ("Downstream")
 - Tools to support later activities such as programming, debugging and testing.

CMSC 435 - 24

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; it must be understandable, usable and compatible with other systems.

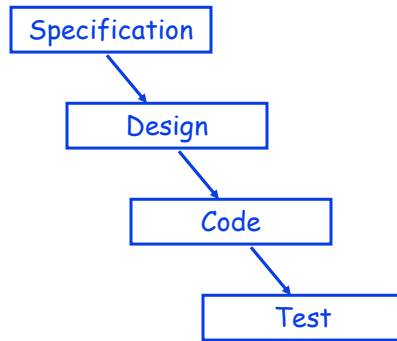
CMSC 435 - 25

What are the key challenges facing software engineering?

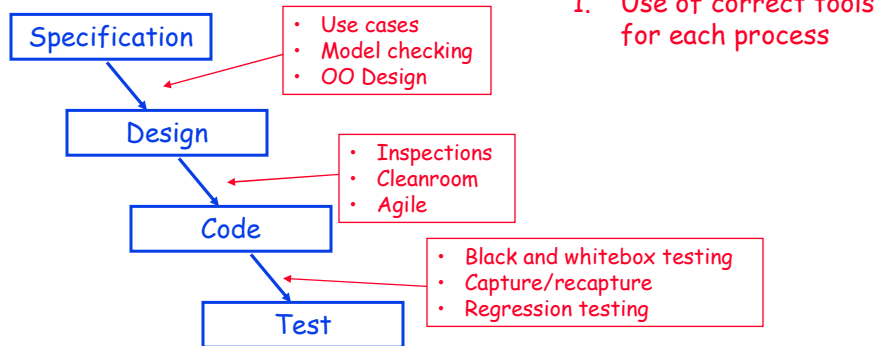
- Heterogeneity, delivery and trust.
- Heterogeneity
 - Developing techniques for building software that can cope with heterogeneous platforms and execution environments;
- Delivery
 - Developing techniques that lead to faster cost-effective delivery of software;
- Trust
 - Developing techniques that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

CMSC 435 - 26

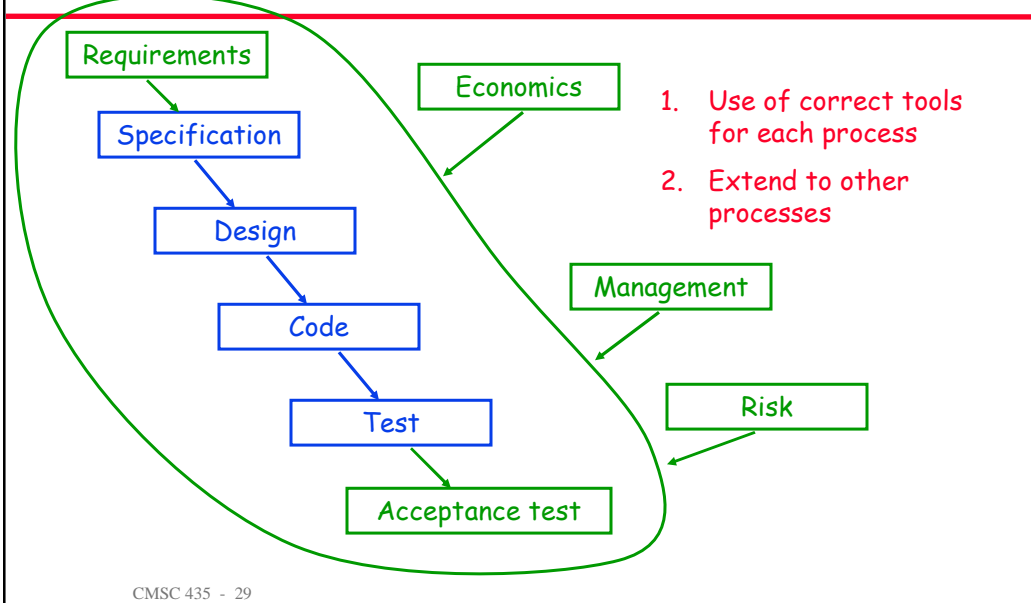
Programming



~~Programming~~ Software Engineering



~~Programming~~ Software Engineering



What attributes are new in 435?

- What is different about these attributes from projects developed in earlier courses?
 - ❑ Economics - Can I build it on time and within budget?
 - ❑ Risk - What can happen and how likely is it for project completion not on time or within budget?
 - ❑ Management - How to manage and schedule people for most efficient and effective utilization of them

Programming Systems extend to entire life cycle of product

- **Problem:** Program to read in a rational number and compute its square root.
- **Programming solution:** A relatively small program in some programming language.
- **Programming system:**
 - Source program in some language
 - Executable file for common platforms
 - Installation script to install on various platforms
 - Documentation of requirements
 - Documentation of design
 - Error messages and corrective action
 - Timing and performance data
 - Users guide
 - Test data
 - Help system to fix errors

Turn in this programming system on Thursday, February 5th

CMSC 435 - 31

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 behavior is more than simply upholding the law.

CMSC 435 - 32

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 beyond their competence.
- Issue: Should software engineers be licensed?
What does competence really mean here?

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).

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.

CMSC 435 goals

- Learn components of good software engineering processes
- Learn role of experimentation in the software life cycle
- Learn to work in groups to build a product
- Learn role of reuse and maintenance in the software life cycle

Key points

- Computer science is concerned with getting the computer to do what you want it to do, as efficiently as possible.
- Software engineers use their computer science skills to create products of practical use and economic value. Software engineers are ethically responsible for the correctness, suitability, and safety of their projects. When possible, software engineers apply scientific and mathematical knowledge to their work.
- A software development process is a process by which user needs are translated into a software product. Software development processes are comprised of specific software development practices.
- A software process model is a generalized abstraction of a family of software development processes.
- Plan-driven processes are best for projects with a low degree of change or those with critical safety and security needs.
- Software engineering is especially challenging because software is a tractable medium, requirements often change, and competitive pressures cause schedule pressure.