Lecture 01

Introduction to the Software Engineering

1. Make friends

2. Don’t fall behind

3. Don’t skip lectures / labs / tutorials

4. Submit assignments on time

5. Study every day

6. Avoid memorizing material…focus on understanding concepts

7. Seek help if you are falling behind for any reason

8. NEVER CHEAT

Introduction to SE

- SE provides an introduction to some of the basic methods and principles used by engineers, including fundamentals of technical communication, measurement, analysis, and design
- Some aspects of the engineering profession, including standards, safety and intellectual property are also covered
- Special emphasis is placed on the analysis and design of software systems

What is Software Engineering?

- It is very difficult to define software engineering!

- Software engineering requires:
  - Classical engineering project-management skills
  - Specialized tools and knowledge
  - Designing, building, analyzing, and maintaining complex computer software
  - Broad understanding of the natural sciences
  - Ability to communicate and work with specialists in other engineering disciplines

- Computer programming is only a small fraction of the work performed by a typical software engineer

Definitions of Software Engineering

- **Software engineering** is a systematic approach to the development, operation, maintenance, and retirement of software

- **Software engineering** is the application of science and mathematics by which the capabilities of computer equipment are made useful to man via computer programs, procedures, and associated documentation[by Boehm ’81]

NOTE:

The key difference between a computer programmer and a software engineer is quite simple. A software engineer is a professional trained specifically to design software that safeguards life, health, property, economic interests, the public welfare and the environment.

Software vs. Software Engineering

- Is a collection of computer programs, procedures, rules, and associated documentation and data
- Generally complete itself, and generally used by the author of the program. Little documentation or other aids
- Program are tested by author only for getting actual outputs.
- Portability is not a key issue

- Is a programming systems product (is a systematic approach to the development, operation, maintenance, and retirement of software)
- Generally used largely by people other than the developers of the system of different background(user interface, sufficient documentation),
- Programs are thoroughly tested before put into operational use.
- Portability is a key issue(variety of environment, variety of hardware etc.)

Software vs. Software Engineering

- A program to solve a problem
- These programs are not designed with such issues as **protability, reliability, usability** in mind.
- The presence of "**bugs**" is not a major concern. If the bugs are presence, the author will fix the program and start using it again.

- A programming systems product to solve the same problem are two entirely different thing
- A software product is an entirely conceptual entity, so there is an "intellectual distance" between the software and the problem the software is solving
- Never wears out due to age, failure occur due to conceptual process (Design fails, other reasons)

SE = Computer Science + Engineering

Computer Science

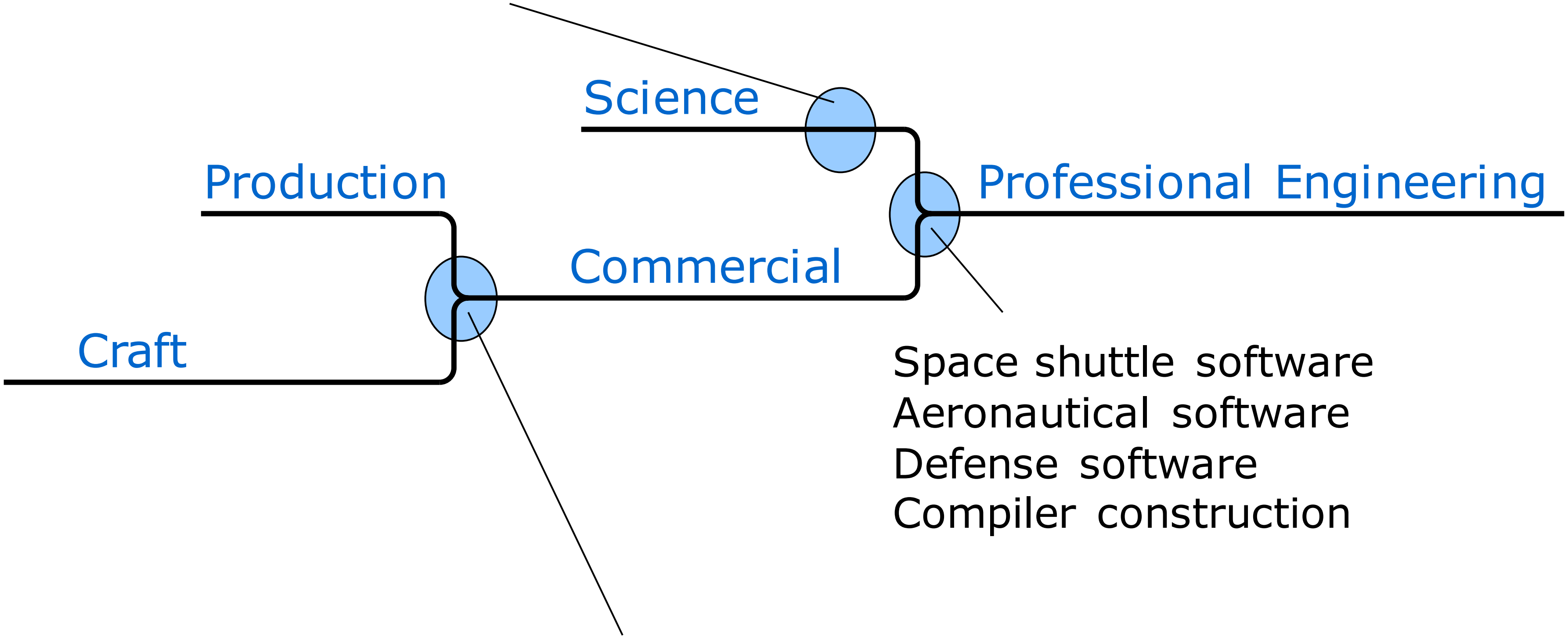
- Data management
- Data transformations
- Design patterns
- Algorithm paradigms
- Programming languages
- Human-computer interfaces

Engineering

- Disciplined processes
- Scalable design principles
- Design evaluation
- Effective documentation
- Coordinated teams
- Non-functional metrics
  - Performance
  - Reliability
  - Maintainability
  - Ease of use

The Emergence of Software Engineering

1960s: data structures, algorithms
1990s: design patterns, architecture patterns



Space shuttle software
Aeronautical software
Defense software
Compiler construction

1980s: software development processes

Mary Shaw, "Prospects for an Engineering Discipline of Software", IEEE Software, November 1990

Why is Software Engineering Important?

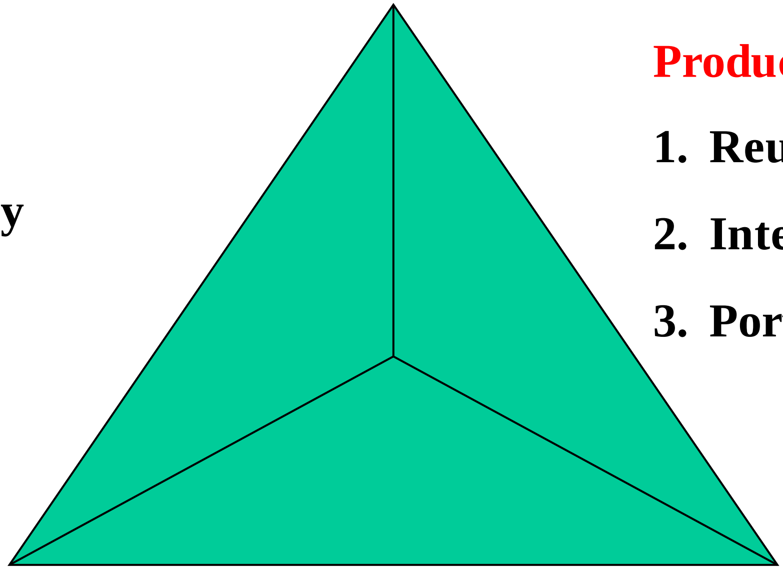
- Computers control most modern machinery:
  - Software is often responsible for maintaining the safe operation of complex machinery
- Software bugs have led to some serious consequences:
  - Therac-25: A computer-controlled radiation therapy machine massively overdosed six people
  - Patriot Missile: An automated anti-missile defense system failed to detect incoming missiles after running continuously for 15+ hours
  - Ariane-5: A European Space Agency rocket self-destructed 40 seconds into its maiden flight
  - Denver Airport: Software problems in the automated baggage-handling system kept airport closed for over a year

The Software Engineering Experience

- Software engineering requires dedication and hard work
- Workload is heavy but the rewards are numerous
  - Satisfying work-term placements
  - Exciting careers
  - Job security
- Software engineering is an emerging engineering discipline with a bright future ahead of it

Software Quality

- Quality metric – three dimensions of the product



Product Revision

1. **Maintainability**
2. **Flexibility**
3. **Testability**

Product Transition

1. **Reusability**
2. **Interoperability**
3. **Portability**

Product Operations

1. **Reliability**
2. **Correctness**
3. **Efficiency**
4. **Usability**
5. **Integrity**