

CS 501: Software Engineering

Fall 2000



Lecture 27

Software Engineering as Engineering

Administration



- Early examination: send email to:
rosemary@cs.cornell.edu

The Y2K Problem: Saving Memory



- In 1967 memory cost \$1 per byte

The Air Force used single digit dates

- If 2-digit dates saved 1% of memory...
savings over 20 years \$16 to \$24 million per gigabyte

The Y2K Problem: Saving Memory



- By 1980s, memory was much cheaper, but 2-digit dates were standard. Why incur the cost of changing standards?

1970 The mortgage industry

1990 The Social Security Industry

moved towards 4-digit dates

- **On January 1, 2000 2-digit dates stopped working!**

Where's the Problem?



- A simple bug:
 dates of the form 19xx have been encoded xx
- A simple fix:
 find every occurrence of the bug
 modify the code
 recompile

Where's the problem?

Find Every Occurrence ...



- What computers do we use?
 - data processing
 - control
 - embedded systems
 - personal devices
- What programs do they run?
 - in-house development
 - packages and libraries
 - firmware, microcode, hardware

Who wrote this program?

Where is the source code?

Where's the Problem?



Computers fail everyday. What's special about this bug?

- What if they all fail at the same time?
- What if we lose telephone, electricity, radio, etc.?
- Traffic signals, elevators,

The greatest worry was uncertainty.

Social Consequences



Worry creates its own problems:

- Wal-Mart forecast lower profits in Q1 2000
- Legislation to limit law suits
- Opportunities for computer fraud and sabotage
- Trading partners

Organizational Procedures



- Ostrich
 - => do nothing
 - => buy insurance
- Bureaucratic
 - => fill in forms that programs are compliant
- Subcontract
 - => hire Y2K specialists
- Do it yourself
 - => in-house computing department

Y2K Validation



Request from Library of Congress to confirm that our code is Y2K compliant:

Our code is fine

.... but it depends on ... which depends on ...

Yes. Our code is fine.

Request from DARPA to confirm that our code is Y2K compliant:

It's been validated by another part of the US government

Thank you!

Technical Strategies



- Replace noncompliant applications with compliant ones (e.g., new versions of packages)
- Repair noncompliant applications (e.g., in-house applications)
- Terminate noncompliant programs on an as-needed basis
- Mask the data exchange between applications
- Object code interception

New Bugs



If it's not broke don't fix it.

- 10 billion lines of code checked (often automatically)
- 10 million new bugs introduced accidentally
- ?? security holes, errors, etc. introduced accidentally or deliberately

Is all the Money Going to Y2K?



Y2K as a great excuse to have the computing budget increased:

- Upgrade the operating system
- Replace the old package
- Sell something to your customers

What boss will turn turn a request for Y2K funds?

What systems administrators will not install Y2K upgrades?

Profiteering



- Buy gold, wood stoves, bottled water
- Y2K specialists
- Pundits, consultants, writers
- Religious cranks

Final Thoughts on Y2K



We create computer systems that are more complex than our understanding of them:

- We over estimate our ability to validate systems
- We under estimate our ability to adapt and respond

Software engineering usually thinks of systems as independent.

Will the long-term benefit of the Y2K problem be a greater understanding of how software systems interact with each other and with our social systems?

The Need for Software Engineering



Software as a product:

=> Awkward to use

=> Full of errors

=> No chance to try it out

=> No guarantees

Not much of a product

What is Engineering?



What is Engineering?



The profession of:

... creating cost-effective solutions ...

... to practical problems ...

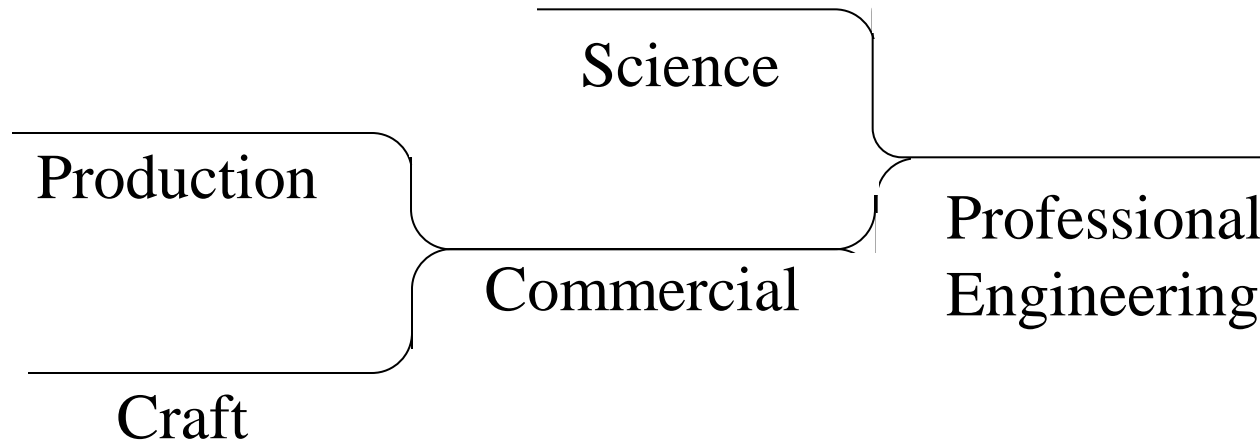
... by applying scientific knowledge ...

... and established practices ...

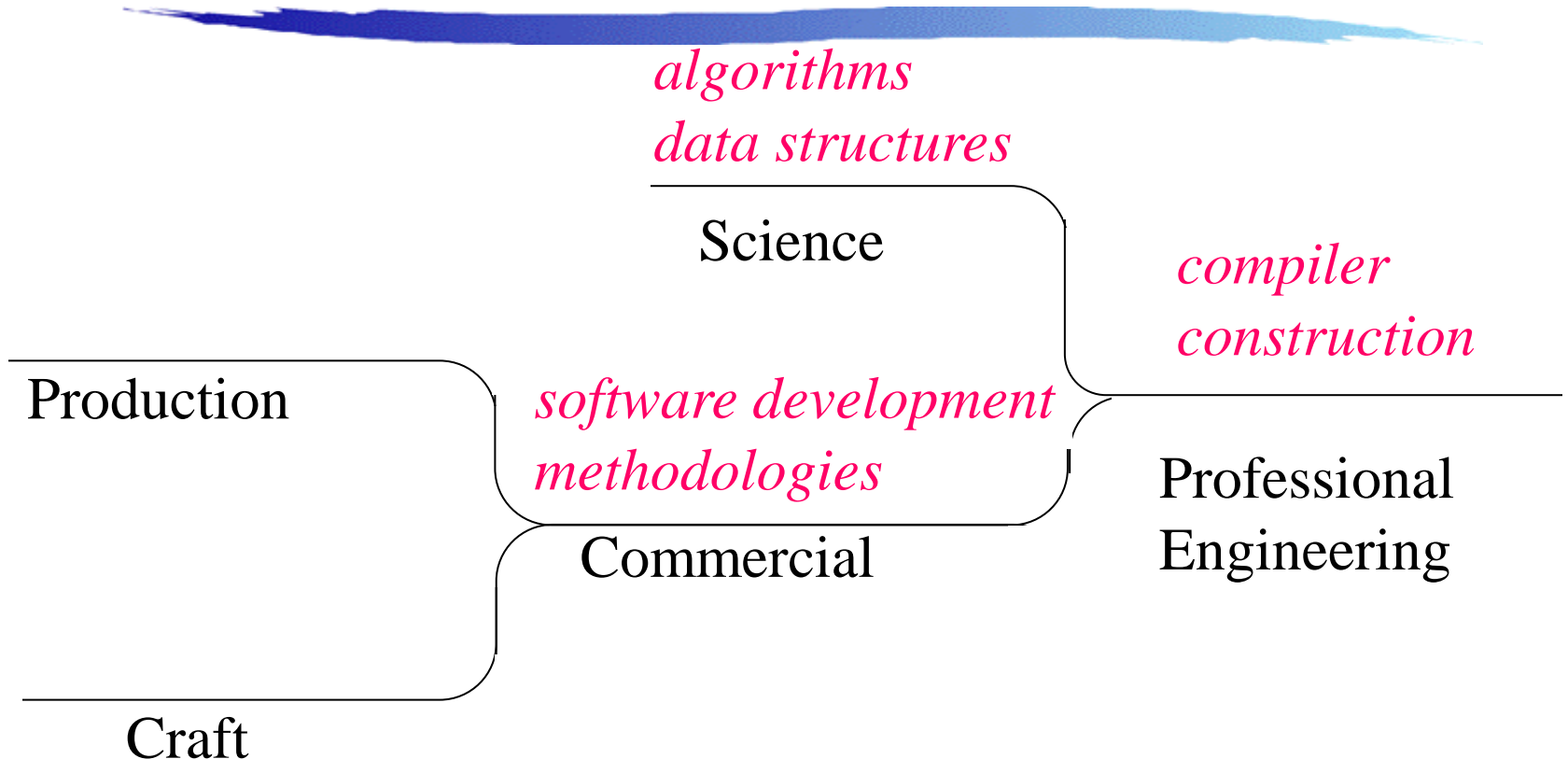
... building things ...

and taking responsibility for them!

Crafts, Science, Engineering



Crafts, Science, Engineering



Software Engineering as Engineering?



- Part craft -- part engineering
- Embryonic scientific basis
- Evolving body of knowledge
- Too much flux for the apparatus of a profession (e.g., accreditation)

Example: Texas and the ACM

The End



- Good process leads to good software:
the limits of heroic efforts
- Minimize risks:
visible process
function v. time v. cost
- The importance of people

Requirements, requirements, requirements!