Welcome to CS220
Software Construction

October 1st, 2019
Shan Lu

https://www.classes.cs.uchicago.edu/archive/2019/fall/22001-1/
Outline

• Technical stuff
  – What is software engineering
    • What are the goals & challenges
  – What is a software engineering process
    • Waterfall model

• Administrative stuff
  – Who I am
  – Components/tasks/schedule of this class

• A brief history of software engineering
My background

• Shan Lu
  • JCL 343, shanlu@cs.uchicago.edu
  • Office hours: after-class—11:50pm, Tu/Th

  – East China → Illinois → Wisconsin → Illinois

  – Research
    • Software reliability, software efficiency, etc.

  – Teaching
    • I enjoy discussion
Our TAs

- **Yuxi Chen**
  - chenyuxi@uchicago.edu
  - Office hour: 4—6pm Tu/Th @ ??

- **Chi Li**
  - lichi@uchicago.edu
  - Office hour: 4—6pm Tu/Th @ ???
Your background?

- How many programs have you written?
  - What are the sizes of your programs?
- What programming languages do you use?
- How familiar are you with O-O?
Software Construction
--- An engineering discipline about all aspects of software production
What do you do to produce software?
What are the aspects of Software production?

• Gathering requirements
• Design
• Development
• Testing & debugging
• Maintenance
What is the goal of S.E.?

• What are the criteria for good programmers?

• What are the criteria for good software?

• The goal of software engineering is ...
What is the goal of S.E.?

• What are the criteria for **good** programmers?
  – Write good software
  – Be on time

• What are the criteria for **good** software?
  – Reliable/correct (few bugs)
  – Efficient (run fast)
  – Maintainable
  – Good usability
  – Good security

• The goal of software engineering is
  – Produce good software, within time schedule, within resource budget
What are the challenges?
What are the challenges?

• Large code sizes
  – Linux Kernel 1.0.0 (1994) 100K+
  – Linux Kernel 2.2.0 (1999) ?
  – Hubble Space Telescope ?
  – Chrome? Firefox?
  – Boeing 787?
  – Mac OS X Tiger?
  – Car software
  – healthcare.gov

• Changing requirements
  – User, hardware, ...

• Large development team (at different geo locations)
Google

- 15000+ developers in 40+ offices
- 4000+ projects under active development
- 5500+ submissions per day on average
- Single monolithic code tree with mixed language code
- Development on one branch - submissions at head
- All builds from source
- 20+ sustained code changes per minute with 60+ peaks
- 50% of code changes monthly
- 75+ million test cases run per day
How to ...?

• Practices/disciplines

• Tools
Engineering
Software Construction
--- Practices and tools about design, development, and maintenance of software
S.E. process

• A sequence of activities that lead to the production of a software product

• There are many processes proposed
  – Waterfall
  – RUP (Rational Unified Process)
  – Agile
    • Extreme programming
Waterfall model

• Activities ➔ separate process phases
Waterfall model
Waterfall model phase I

• Requirement & analysis

• Where do we obtain the requirement?
• Should we modify or refine the requirements?
  – What should we consider?

• Output
Waterfall model phase II

- Design

- What need to be designed?

- Output
Waterfall model phase II

• Design

• What need to be designed?
  – UI
  – Module, API interface (architecture design)
  – Data structure (component design)

• Output
  – Design document
Waterfall model phase III

• Implementation

• Output
Waterfall model phase IV

• Testing

• Output
Waterfall model phase IV

- Testing

- Output
Waterfall model phase V

- Maintenance

- Ratio of cost among phases
Problems with waterfall model
Problems with waterfall model

• Difficult to handle changes (not in model, high cost)
• Error fixing expensive
• Hard to estimate time
More information at ...

- Chapter 2.0, 2.1, 2.2.0, 4.1.0, 4.1.1

- Chapter 1.1.1, 2.1.0, 2.1.1
Administrative Stuff
An overview of our schedule

<table>
<thead>
<tr>
<th>Date</th>
<th>Topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>10/01</td>
<td>Introduction, Software Processes [notes]</td>
</tr>
<tr>
<td>10/08</td>
<td>Project Discussion, Requirement Engineering &amp; System Modeling I [notes]</td>
</tr>
<tr>
<td>10/15</td>
<td>System Modeling II [notes]</td>
</tr>
<tr>
<td>10/22</td>
<td>Testing 1 [notes]</td>
</tr>
<tr>
<td>10/29</td>
<td><strong>Midterm</strong></td>
</tr>
<tr>
<td>11/05</td>
<td>Refactoring [notes]</td>
</tr>
<tr>
<td>11/12</td>
<td>Design Patterns II [notes]</td>
</tr>
<tr>
<td>11/19</td>
<td>Parallel Software Construction [notes]</td>
</tr>
<tr>
<td>11/26</td>
<td>No Class (Happy Thanksgiving)!</td>
</tr>
<tr>
<td>12/03</td>
<td><strong>Project Presentation</strong></td>
</tr>
<tr>
<td>10/03</td>
<td>Agile, Extreme Programming [notes]</td>
</tr>
<tr>
<td>10/10</td>
<td>Architectural Design [notes]</td>
</tr>
<tr>
<td>10/17</td>
<td>System Modeling III [notes]</td>
</tr>
<tr>
<td>10/24</td>
<td>Testing 2 [notes]</td>
</tr>
<tr>
<td>10/31</td>
<td>Code Smell [notes]</td>
</tr>
<tr>
<td>11/07</td>
<td>Software Maintenance and Design Patterns I (Observer, Composite, Interpreter) [notes]</td>
</tr>
<tr>
<td>11/14</td>
<td>Design Patterns III [notes]</td>
</tr>
<tr>
<td>11/21</td>
<td>Bugs and Bug Detection [notes]</td>
</tr>
<tr>
<td>11/28</td>
<td>No Class (Happy Thanksgiving)!</td>
</tr>
<tr>
<td>12/05</td>
<td>No Class (Reading Period)</td>
</tr>
</tbody>
</table>

**Any student graduating at the end of this quarter?**
There are a lot of work to do

- Class

- 1 mini project (due 10/14) 8%
- 1 big programming project 41%
  - Many milestones/checkpoints
  - Proposal due 10/16
- Weekly Quiz after mid-term 6%
  - 11/05, 11/12, 11/19
- Two exams 45%

If you are going to drop this course, do it soon.
What you need to do 1: lectures & reading

- Lectures
  - Tu/Th 2—9:30 am

Links in my slides
What you need to do 2: Quizzes

• ~15 minutes @ Tuesday lectures in November
• Close-book, close-note
• Cover lecture content

• 2 point for each quiz, 6% of your overall grades
  (don’t miss it; let me know to reschedule if ...)
What you need to do 3: Project

• Course project
  – 7—8 people a group
  – The whole process
  – 6+ milestones

<table>
<thead>
<tr>
<th>Date</th>
<th>#</th>
<th>Activity</th>
</tr>
</thead>
<tbody>
<tr>
<td>10/16</td>
<td>1</td>
<td>Proposal (2—3 students)</td>
</tr>
<tr>
<td>10/30</td>
<td>2</td>
<td>Planning (7—8 students)</td>
</tr>
<tr>
<td>11/06</td>
<td>3.a</td>
<td>Testing of 1(^{st}) iteration</td>
</tr>
<tr>
<td>11/13</td>
<td>3.b</td>
<td>End of 1(^{st}) iteration</td>
</tr>
<tr>
<td>11/20</td>
<td>4.a</td>
<td>Testing of 2(^{nd}) iteration</td>
</tr>
<tr>
<td>11/27</td>
<td>4.b</td>
<td>End of 2(^{nd}) iteration</td>
</tr>
<tr>
<td>12/03</td>
<td>5</td>
<td>System testing &amp; documentation</td>
</tr>
<tr>
<td>12/08</td>
<td>6</td>
<td>Acceptance testing &amp; debugging</td>
</tr>
</tbody>
</table>

– 41 % of your final grade
– Grading criteria: 30 pts group + 11 pts individual
– There will be peer evaluation
What you need to do 4: warm-up project

• One warm-up project
  – Will be released today or tomorrow
  – Do it in a group of two people

• It is due on 10/14th
What you need to do 5: Exams

• Midterm exam
  – In the lecture on 10/29
  – 20% of your final grades

• Final exam
  – During the exam week
  – 25% of your final grades

• Cover material from class and the projects
Overall Grade

• Curved

• 2018 Fall
  -- A* 17; B* 16; C* 1

• 2018 winter
  – A* 25; B* 5; C* 2

• 2017 winter
  – A* 19; B* 8; C* 5

• 2014 Fall
  – A* 22; B* 14; C* 4
Resources

• CSIL Labs

• TA
  – Yuxi Chen, chenyuxi@uchicago.edu
  – Chi Li, lichi@uchicago.edu

• Piazza!! (will start by the end of this week)
• Feel free to ask me questions in&off class
A brief history I

• The pioneering era
  – No S.E.
  – No way to estimate s/w development time
  – s.w. is free
• Starting 1960s
• The Software Crisis 1965--1985
  – Therac 25 1985—1987
  – Morris worm 1988
A brief history II

• 1985 – 2000
  – No silver bullet
  – **OO, design patterns**, formal methods, **process**

• 2000 – present
  – Agile
  – Model-driven design

  – **Tools, Program synthesis, verification, ...**
Current S.E. research

Belvin’s Law 3

YESSS!

If the code works the first time, that just means that the bug is more carefully hidden.

Gee, thanks for that.

© 2011 The Code Zone
Visit angriestprogrammer.com for daily cynicism
Summary

• What we discussed
  – What is software engineering
  – What is s.e. process
  – Waterfall model

• What you should do/prepare to do
  – Check course webpage
  – Check piazza
  – Mini-project to be released soon
  – Project proposal