CS 278: Principles of Software Engineering

Dr. Jules White
Magnum Research Group Director
Assistant Professor
Institute for Software Integrated Systems
Dept. of Electrical Engineering and Computer Science
Vanderbilt University
jules.white@vanderbilt.edu

Magnum Research Group

Software Engineering is The American Dream

If your goal is to graduate, make a ton of money, and retire early, networked apps are the right route:

1. Mark Zuckerberg – Facebook
2. Eric Schmidt – Google
3. Sergey Brin – Google
4. Shawn Fanning – Napster
5. Larry Ellison - Oracle
6. Steve Jobs – Apple
7. ….too many others to list…..

7 of the top 20 billionaires are from software backgrounds

Magnum Research Group

Unfortunately…

Of all software development projects, 53% are considered unsuccessful

Of all patients with the Bubonic plague, 11% don't recover

Software projects are riskier than the Bubonic plague.

Core Topics

- Software Lifecycle Models
- Testing
- Version Control
- Development Automation
- Refactoring
- Software Design Patterns
- Frameworks
- Mobile Computing
- Cloud Computing
- Secure Coding
- Maintenance / Deployment
- Requirements / Design

There Are No Rails

- When you graduate, there will be no more shells, clear tasks, etc.
- In the real-world, software engineering involves dealing with ambiguous and changing requirements, challenging schedules, and best guesses about how to make things work
- This class will expose you to that world and quickly throw away the hand-rails

Many Assignments will be Open-Ended

Magnum Research Group

Do Not Panic

Magnum Research Group

Embrace the Open Ocean of SE

Magnum Research Group

Learn By Doing

1. You can't really understand the principles behind software engineering without large-scale software systems
2. Facing real problems is how you learn
3. The hardest problems are not the ones that I will give you but the ones you will need to solve to realize your ideas
4. You will build large-scale software systems as part of this class
5. You will have architectural and design freedom over how the systems are implemented

Course Structure

1. Each week, there will be an assignment due
2. Almost all assignments will require demonstrating key concepts in software
3. Assignments will be due on Sunday night
4. On Monday morning, the TA will assign each person to code review another person's assignment
5. Code reviews will be due before class on Tues
6. The bulk of class on Tues/Thurs will be discussing everyone's code and the application of the course concepts
7. During class, one or more of the refactorings suggested by the reviewer will be "accepted"
8. You must turn in a revised version of your assignment with any refactorings before the end of the course

Code Reviews

To be a leader in a software project, you need to be able to both apply and communicate to others how to apply key software engineering principles

1. Your code reviews should be well thought out
2. Each week, you will present your code review to the class and lead the discussion of the code base that you were assigned
3. Your grade will be based on both your ability to apply the course concepts and to see how to improve others' code with those principles
4. You will be required to implement any suggested refactorings that your reviewer suggests and that the class agrees with
5. The reviewer and reviewee should be prepared to have a meaningful discussion about the application's design decisions
6. All code reviews will be turned into Piazza by posting them as questions that are tagged with the appropriate folder name for the assignment (e.g., hw1 for asgn1)

Code Refactoring

Being thorough in your code reviews of other people's work will reduce the amount refactoring you have to do in your own code base.

1. Suggested refactorings of your code must be turned in before the end of the semester (and are required before your assignment is graded)
2. All code reviews will be posted as questions in Piazza – feel free to post follow-ups or clarifications to reviews that other people post of your code
3. In class we will discuss the code reviews
4. You will be required to implement "accepted" suggestions for improvement as follows:
   - If you have K of your suggestions for improvement accepted on your review of someone else's code base
   - If you have N suggestions for improvement accepted for your code base
   - Then you must implement N-(K-1) of the suggested improvements in your code

The course material will be in ~~OAK~~ Git

The course is designed to mimic real world software engineering wherever possible. All course materials will be exchanged through Git.

1. Each student will have their own public GitHub repo where they will commit their assignments
2. Assignments are turned in by committing them and pushing to your git repo
3. The course material will be stored in Git
4. The first 6 assignments and the reading schedule for the class are already in the class repo
5. It is essential that you quickly get up to speed on Git if you are not familiar with it
   - Start the course reading today

Note: Do not check graded material into Git!

You should only check ungraded material into Git!

CS 278 Course Information

- Me: Jules White – jules.white@vanderbilt.edu
- TA: Satabdi Basu
- satabdi.basu@vanderbilt.edu
- CS 278 class web page
  - https://piazza.com/vanderbilt/fall2013/cs278/home
  - https://github.com/juleswhite/cs278
- My office hours in FGH 229 are
  - Tues/Thurs after class until 2:15pm or by appointment

Please post all questions on Piazza:

https://piazza.com/vanderbilt/fall2013/cs278/home

TA office hours in FGH 314:

Mon/Wed from 10-12

No office hours today, instead they will be held tomorrow from 2:30-4pm

CS 278 Rules

- Assignments must be completed on time. No credit for assignments that are more than 8hrs late (8am Monday morning)
- No credit for code reviews that are late
- You are free and encouraged to use opensource libraries / code – as long as you *understand* them
- You are free to use Stack Overflow, Google, etc. as long as you understand the material you use (no blind cutting/pasting)
- You are free to talk with each other about how you plan to solve problems
  - However, you will be responsible for explaining your design rationale / code – if you can't explain it, you don't get credit for it
  - No copying code

Googling for Code

- I encourage everyone to Google for answers
- Google can probably get you an answer faster than I can email you back
- When you get stuck, before sending an email and waiting for an answer, take a few minutes to Google for a solution
- Stack Overflow has a wealth of information on implementation-specific details:
  - http://stackoverflow.com/
- In the real-world, you will be expected to be able to search the Internet for solutions on your own

Grading

Course Grading Breakdown:

- Weekly assignments* – 80%
- Final project* – 20%

Weekly Assignment Grading Breakdown:

  - Assignment code – 50%
    - Must compile and be a reasonable attempt for any credit
  - Code review – 25%
  - In-class discussion on design / review rationale – 25%

* I reserve the right to change the weighting of assignments

Magnum Research Group

Basic Requirements

- All of the following requirements must be met in order to receive partial credit on an assignment:
  - Everything compiles
  - You showed up and presented your code review
  - You made a reasonable attempt at completing the assignment
  - You came to see me in office hours to get help

I would rather…

- The most important aspect of this class is demonstrating that you know the concepts well enough to apply them
- The context that you demonstrate those concepts in is not important to me
- On any assignment, if you would prefer to demonstrate the concepts in an alternate context (e.g., I want to write XYZ really cool application), you are free to do so…if:
  - You get the alternate application approved by me before Thursday (e.g., if it is due on 8/25, you get approval by 8/22)
  - The alternate context will sufficiently demonstrate the concepts

~~Be Prepared for Weekly Quizzes~~

- Complete the assigned reading every week and be prepared to discuss it in class
- The reading list is already in Git

- It is possible that you might have a pop quiz on the reading
  - (one of the reasons that I reserve the right to change the grade weighting)

Assignment Progression

- The assignments are going to rapidly progress in difficulty
- I would ***strongly encourage*** you to start each assignment as soon as you finish the previous one
- All of the assignments are in GitHub, if you know that you are going to have some midterms, you are free to complete assignments ahead of schedule so that your midterm and the assignment don't overlap
- Start early
- Ask questions about the assignments in class
- Come to office hours

Assignment Discussion

- Please ask questions about the assignment in class
- I will start every class by asking if there are any questions about the next assignment
  - I will be liberal in giving away answers to questions that provoke good discussions in class
  - I will be conservative about giving away answers in office hours

Questions

- We will be using Piazza for questions (in and out of class)
- https://piazza.com/vanderbilt/fall2013/cs278/home

- We will try to keep as much of the in-class discussion in Piazza as possible – as long as it works as expected and doesn't impede discussion

What if I Just Can't Get X to Work?

- If you realize that something is much harder than expected to implement, don't panic
  - Ask about the assignment in class
  - Come see me in office hours
  - We will either figure out how to overcome the problem or come up with an exit strategy so you will still get a good grade for that assignment
- Start early so that you can predict if you aren't going to finish an assignment

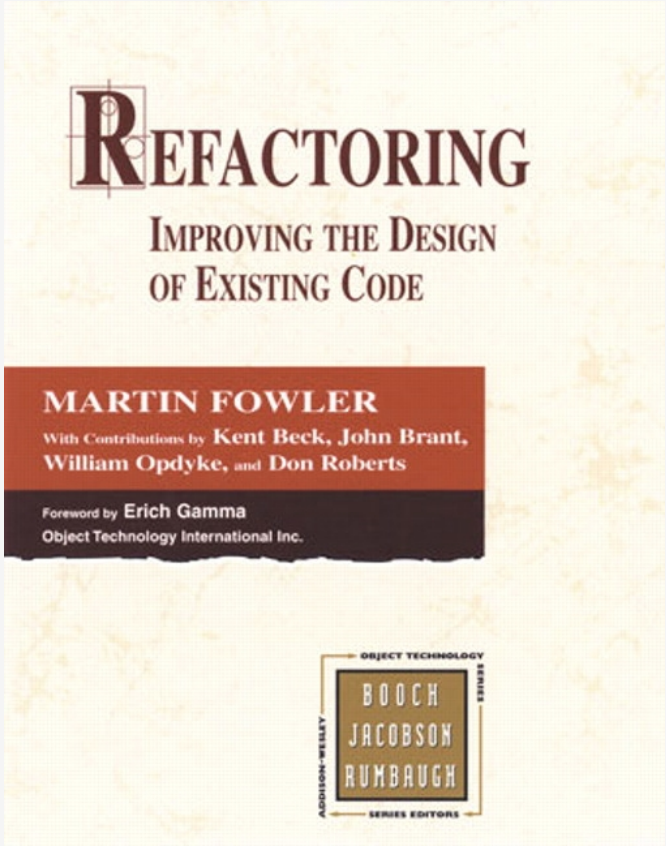
Do Overs (undergrads only)

- Undergrads can choose two assignments to skip.
- You must ask and get approval to skip an assignment by Thursday
- Only 3 people can skip any given assignment – skipping is first-come, first-serve
- You cannot skip two assignments in a row
- You cannot skip the final project
- You cannot wait until the last minute, realize that you forgot to do the assignment, and then email to get out of it
- Unused do overs will count as bonus points at the end of the course
- *You still must do a code review for any assignment you skip*

Reading Schedule

The reading schedule is in Git.

Yes, you need the book.



Assignment 0

Asgn0 is in the course Git repo

Due tomorrow at 5pm

Since part of the assignment is installing Git…here is a direct link to the assignment:

https://github.com/juleswhite/cs27x/tree/master/assignments/Asgn0

Assignment 1

Asgn1 is in the course Git repo

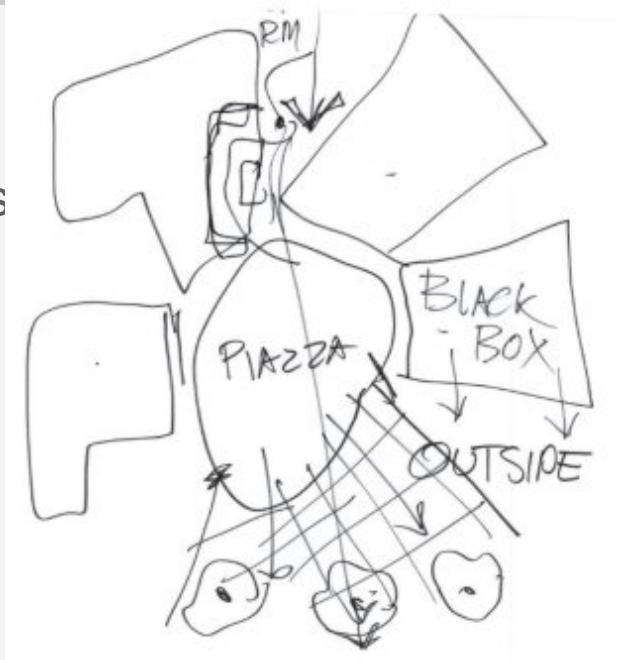
Due Sunday at 11:59pm

You will have a code review due before class on Tues

Demo…

Code Design

- Patterns should be used wherever possible
  - We will learn new patterns as needed in class
- Testing is critical, your code must be designed so that it can easily be tested
- Good application design *assumes that code will be refactored and extended*
  - Make sure that your code does not exhibit tight coupling



Coding Standards

- Basic coding standards:
  - The code format standard should be what you get when run the Eclipse automated code formatter (ctrl + shift + f)
  - You should decide on variable naming conventions and stick with them. I recommend all lowercase letters for local variables, all caps for static variables, and one of the following for member variables:
    - Foo myVariable; //All references to foo use “this”
    - this.myVariable = ….;

    - Foo myVariable_;
    - myVariable_ = ….;
  - Proper Java package naming
    - org.myprojectname.foo.bar

Quiz

- Not for credit