CS 4530: Fundamentals of Software Engineering Module 1.1 Course Introduction

Adeel Bhutta, Jan Vitek and Mitch Wand

Khoury College of Computer Sciences

Instructors

Adeel Bhutta

Sections 1, 2, 3 & 7

Jan Vitek

Sections 4, 5

Mitch Wand

Section 6

Teaching Assistants

- We have around 400 students and 22 teaching assistants.
- Their pictures will be on the website as soon as we collect them

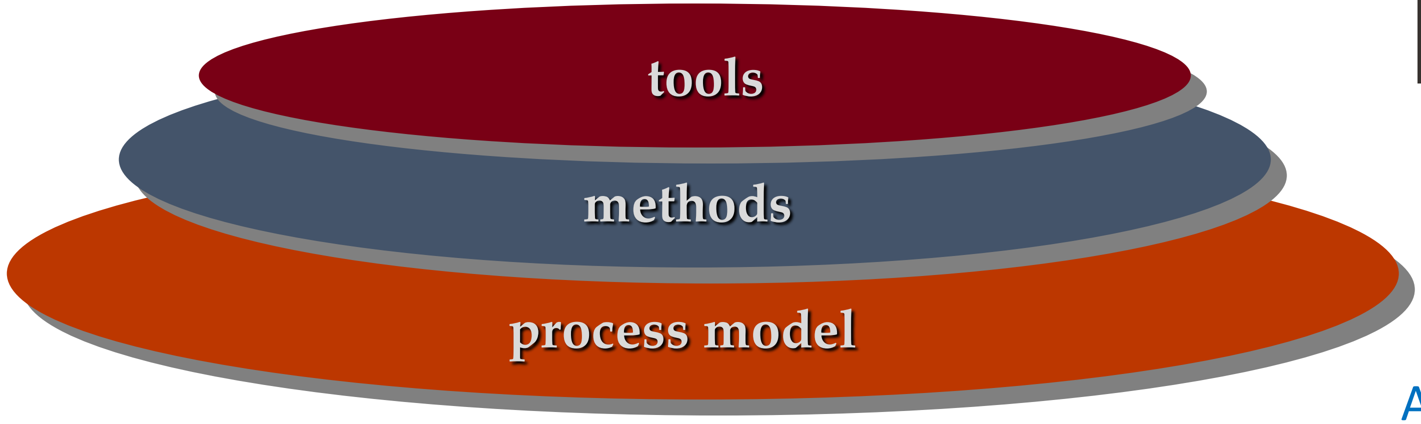
https://neu-se.github.io/CS4530-Spring-2023/staff/

Learning Objectives for this Lesson

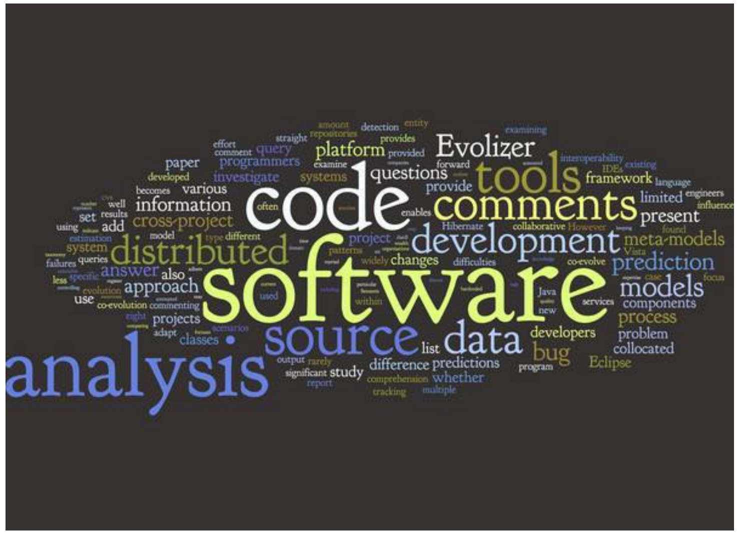
- By the end of this lesson, you should be able to:
  - Explain in general terms what software engineering is
  - List your weekly obligations as a student
  - List the requirements for completing the course

What is software engineering?

- Software Engineering refers to the tools and processes that we use to
  - design,
  - construct, and
  - maintain programs
  - over time.

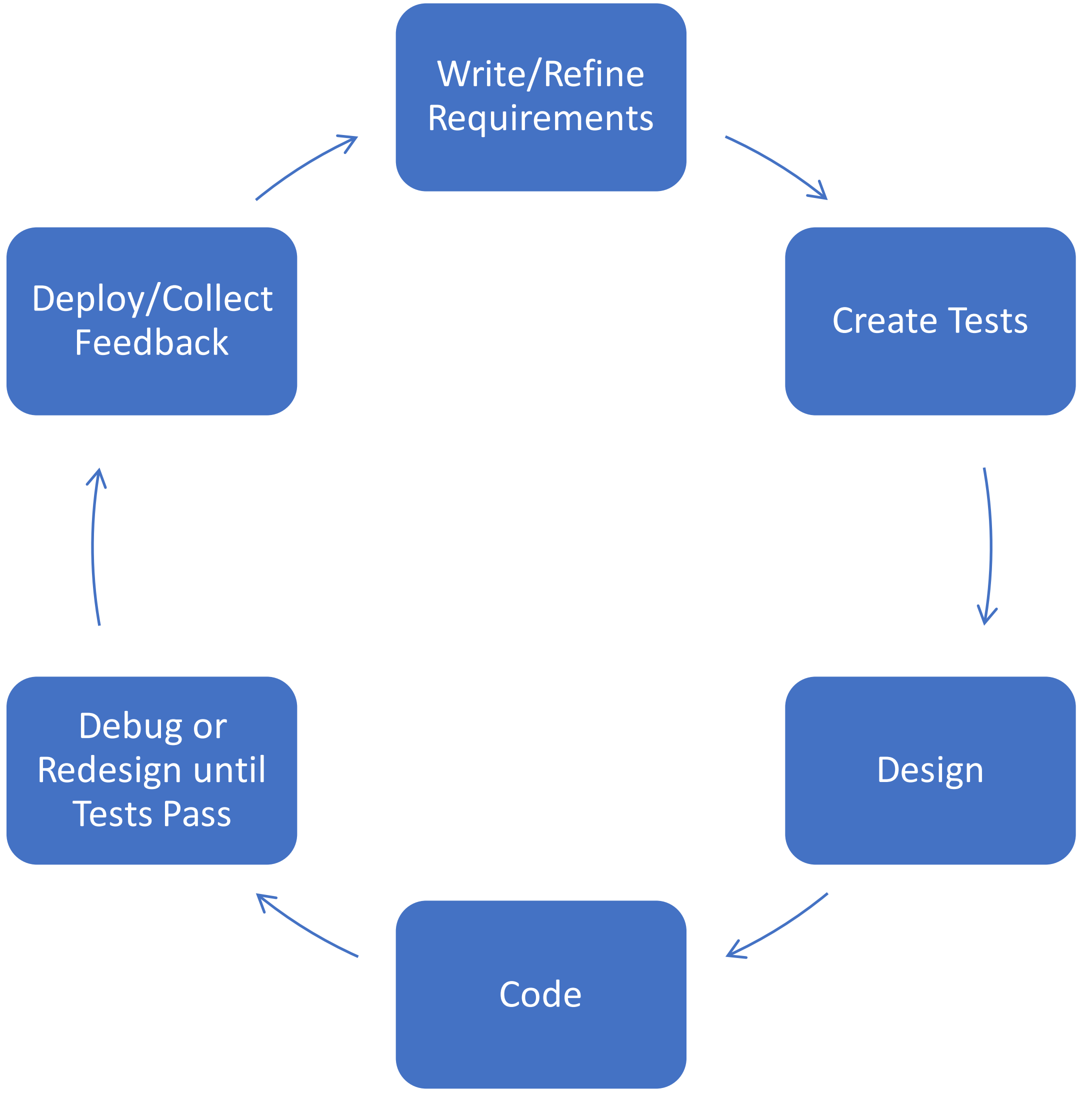


a "quality" focus



Applying "engineering" to Software!

Software Engineering encompasses the entire software development life cycle



But this raises many questions

- How big is each cycle?
  - In code to be written?
  - In time?
  - In person-power?
- Can you have multiple cycles going at once?
- What artifacts need to be produced at the end of each stage?
  - Need to prepare for the next time through the cycle.
  - Need to document what was done, so that others can build on your work.

The answers depend on many factors

- Depends on things like:
  - the size of the team
  - the size of the product
  - the longevity of the product
- There's no one "right" way; there are always tradeoffs.
- But there are best practices, which we will expect you to follow.

Software Engineering is about People

"Any fool can write code that a computer can understand. Good programmers write code that humans can understand"

- Martin Fowler

Learning Objectives for this course:

- By the end of this course, you will--
  - Be able to define and describe the phases of the software engineering lifecycle.
  - Be able to explain the role of key processes and technologies in modern software development.
  - Be able to productively apply instances of major tools used in elementary SE tasks.
  - Design and implement a portfolio-worthy software engineering project in a small team environment that can be showcased to recruiters.

Approach

- The course will mirror the steps of the software engineering life cycle
  - starting with requirements, through testing and deployment
- We will move some material forward to make sure that you have the learning you need when you need it

Technology

- We will use:
  - TypeScript as implementation language
  - Jest as Testing Framework
  - Visual Studio Code as our IDE
  - React for webapps
  - GitHub Projects for Project Management
  - GitHub Actions / Netlify for CI/CD
  - Also, other miscellaneous tools

Course Mechanics

- Our goal is to provide a productive learning environment to both remote and on-the-ground students
- 100% attendance is expected for both on-the-ground and remote sections
- Classes will include both lectures and in-class activities.
- Be sure to bring your laptop
- Weekly slides will be posted in advance, so be prepared to ask and answer questions about the material.

Course Mechanics: In-Class Exercises and Tutorials

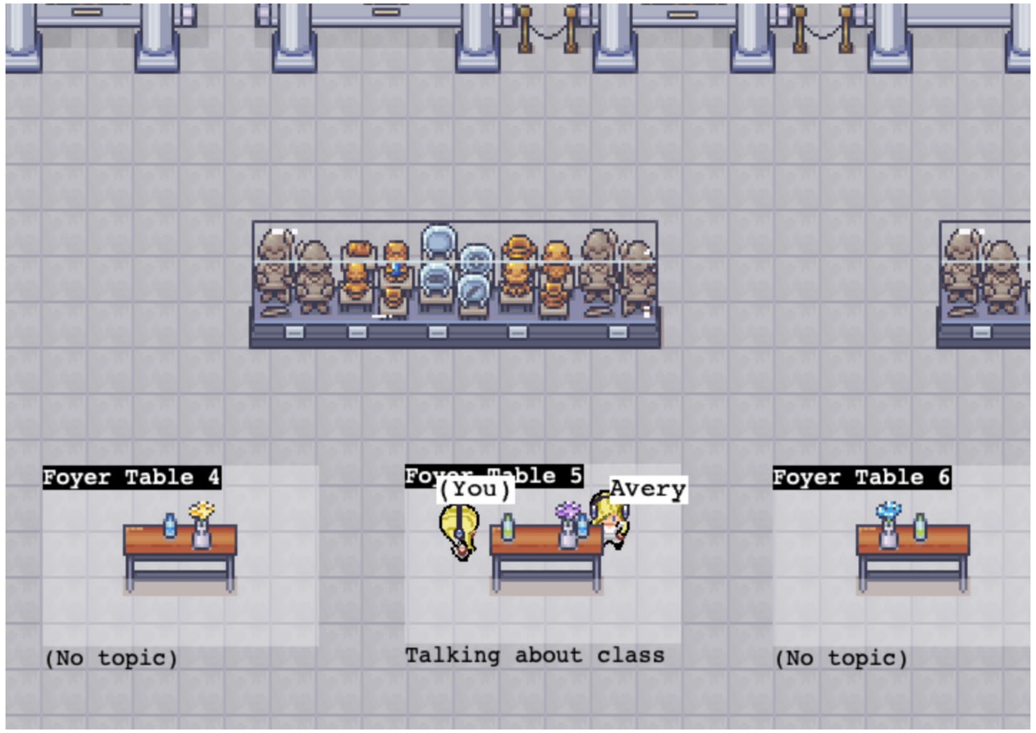
- There will often be in-class exercises to give you practice with the technologies we will use.
- Typically, will consist of structured steps that will guide you through a typical task
- Each instructor will use **individual approach** to grade in-class activities.
- In addition, there will be **tutorials** posted on the web.
- These will extend the in-class materials.

Course Requirements

- We will start with an individual project, divided into 2 deliverables. This is to be done individually
- Then a group project, done in teams of about 4 people
- There will be an exam in Week 9. There will not be a final exam.
- The overall grading breakdown is:
  - 30% Individual Assignments (Individual Projects 1 and 2)
  - 40% Team Project
  - 10% Participation & In-class activities
  - 20% Week 9 Exam

We will use Covey.Town as the running codebase for the course

- Covey.Town is a virtual meeting space.
- Different groups of people can have simultaneous video calls, allowing participants to drift between different conversations, just like in real life.
- Inspired by existing products like Gather.Town , Sococo, and Gatherly.IO
- But it is an open source effort
- The features will be proposed and implemented by you!
- All implementation will take place in the TypeScript programming language, using React for the user interface.



Covey.Town and you

- The individual projects will help you become familiar with the codebase.
- The team project will be a new feature that you will propose.
- Further breakdown of team project grade (i.e., 40%) is:
  - Planning (20%)
  - Process (20%)
  - Product (40%)
  - Reports (20%)
- Peer evaluations (surveys) may be utilized, and individual contributions WILL impact your project grade.

Grade Appeal Policy

- If you have concerns regarding the grading of your work, please let us know right away.
  - All regrade requests must be made through Gradescope.
    - GradeScope provides an interface that allows us to review all regrade requests in one place.
  - Do not post on Piazza or email your TA or instructor
  - All regrade requests must be submitted within **7 days** from your receipt of the graded work.
  - If your regrade request is closed and you feel that the response was not satisfactory, you may appeal to the instructor via email within 48 hours

Late Policy

- Your work is **late** if it is not turned in by the deadline.
  - 10% will be deducted for late HW (individual assignments) turned in within 24 hours after the due date
  - Individual assignments submitted more than 24 hours late will receive a zero.
  - If you're worried about being busy around the time of a HW submission, please plan ahead and get started early.
  - No late submissions allowed for any **group work**
  - If you have an accommodation from Disability Resource Center, you must request it from the instructors separately for each assignment or exam.
    - DRC Accommodations are usually NOT available for Group Assignments

Academic Integrity (1)

- Students must work individually on all homework assignments.
- We encourage you to have high-level discussions with other students in the class about the assignments, however, we require that when you turn in an assignment, it is only your work. That is, copying any part of another student's assignment is strictly prohibited.
- If you steal someone else's work, you fail the class.
- You are responsible for protecting your work. If someone uses your work, with or without your permission, you fail the class.

Academic Integrity (2)

- You are free to reuse small snippets of example code found on the Internet (e.g., via StackOverflow) provided that it is attributed.
- If you are concerned that by reusing and attributing that copied code it may appear that you didn't complete the assignment yourself, then please raise a discussion with the instructor.
- If you are in doubt whether using others' work is allowed, you should assume that it is NOT allowed unless the instructors confirm otherwise.

Communication

- Canvas
- Course web page (https://neu-se.github.io/CS4530-Spring-2023)
  - Canvas and the course web site will mirror each other.
  - Assignments, important notices, etc., will appear in both places.
- Piazza (see Canvas for link)
  - for questions about assignments, projects, etc.

Review

- Now that you've studied this lesson, you should be able to:
  - Explain in general terms what software engineering is
  - List your weekly obligations as a student
  - List the requirements for completing the course

Lesson 1.1 Activity: Introductions

- Activity 1: Introduce yourself and ask students to introduce themselves
- Activity 2 {Optional}: **Welcome Survey**
  - Discuss survey responses and answer questions