

# CSE 403: Software Engineering, Fall 2016

[courses.cs.washington.edu/courses/cse403/16au/](https://courses.cs.washington.edu/courses/cse403/16au/)

# Introduction

**Emina Torlak**

[emina@cs.washington.edu](mailto:emina@cs.washington.edu)

# outline

**Staff, About, Format, Advice, Goals**

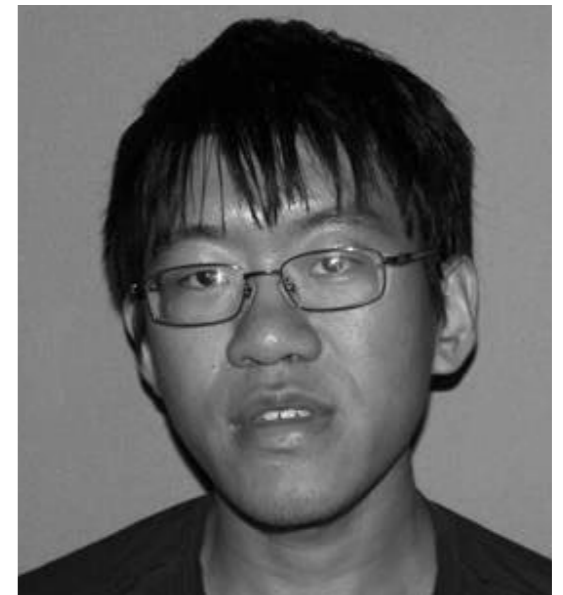
**Course staff: [cse403-staff@cs.washington.edu](mailto:cse403-staff@cs.washington.edu)**



**Emina Torlak**  
(emina)



**Spencer Pearson**  
(suspense)



**Yue Zhang**  
(yjzhang)

**focus**  
**This course is about *engineering* software.**

# What is software engineering?

software engineering  $\neq$  programming

**Software engineering**, broadly defined: creating and maintaining software applications by applying technologies and practices from computer science, project management, and other fields.

Software engineering is about people working in teams under constraints to create value for their customers.

# What is software engineering?

“The first step toward the management of disease was replacement of demon theories and humours theories by the germ theory. That very step, the beginning of hope, in itself dashed all hopes of magical solutions. It told workers that progress would be made stepwise, at great effort, and that a persistent, unremitting care would have to be paid to a discipline of cleanliness. So it is with software engineering today.”



Fred Brooks

# Aspects of software engineering

1. Processes, methods, and techniques necessary to turn a concept into a robust deliverable that can evolve over time
2. Working with limited time and resources
3. Satisfying a customer
4. Managing risk
5. Teamwork and communication

# Ties to many fields

- computer science (algorithms, data structures, languages, tools)



# Ties to many fields

- computer science (algorithms, data structures, languages, tools)
- business/management (project mgmt, scheduling)
- economics/marketing (selling, niche markets, monopolies)
- communication (managing relations with stakeholders: customers, management, developers, testers, sales)
- law (patents, licenses, copyrights, reverse engineering)
- sociology (modern trends in societies, localization, ethics)
- political science (negotiations; topics at the intersection of law, economics, and global societal trends; public safety)
- psychology (personalities, styles, usability, what is fun)
- art (GUI design, what is appealing to users)

# Ties to many fields

- computer science (algorithms, data structures, languages, tools)
- business/management (project mgmt, scheduling)
- economics/marketing (selling, niche markets, monopolies)
- communication (negotiations, interactions with customers, management, etc.)
- law (patents, contracts, intellectual property)
- sociology (modern trends in societies, localization, ethics)
- political science (negotiations; topics at the intersection of law, economics, and global societal trends; public safety)
- psychology (personalities, styles, usability, what is fun)
- art (GUI design, what is appealing to users)

**Necessarily “softer” than other parts of CS;  
fewer clearly right/wrong answers**

# Roles of people in software

- **customer / client:** wants software built
- **managers:** make plans, coordinate team
- **developers:** design and write code
- **testers:** perform quality assurance (QA)
- **users:** purchase and use software product



# Roles of people in software

- **customer / client:** wants software built
  - often doesn't know what he/she wants
- **managers:** make plans, coordinate team
  - hard to foresee all problems in advance
- **developers:** design and write code
  - hard to write complex code for large systems
- **testers:** perform quality assurance (QA)
  - impossible to test every combination of actions
- **users:** purchase and use software product
  - can be fickle and can misunderstand the product



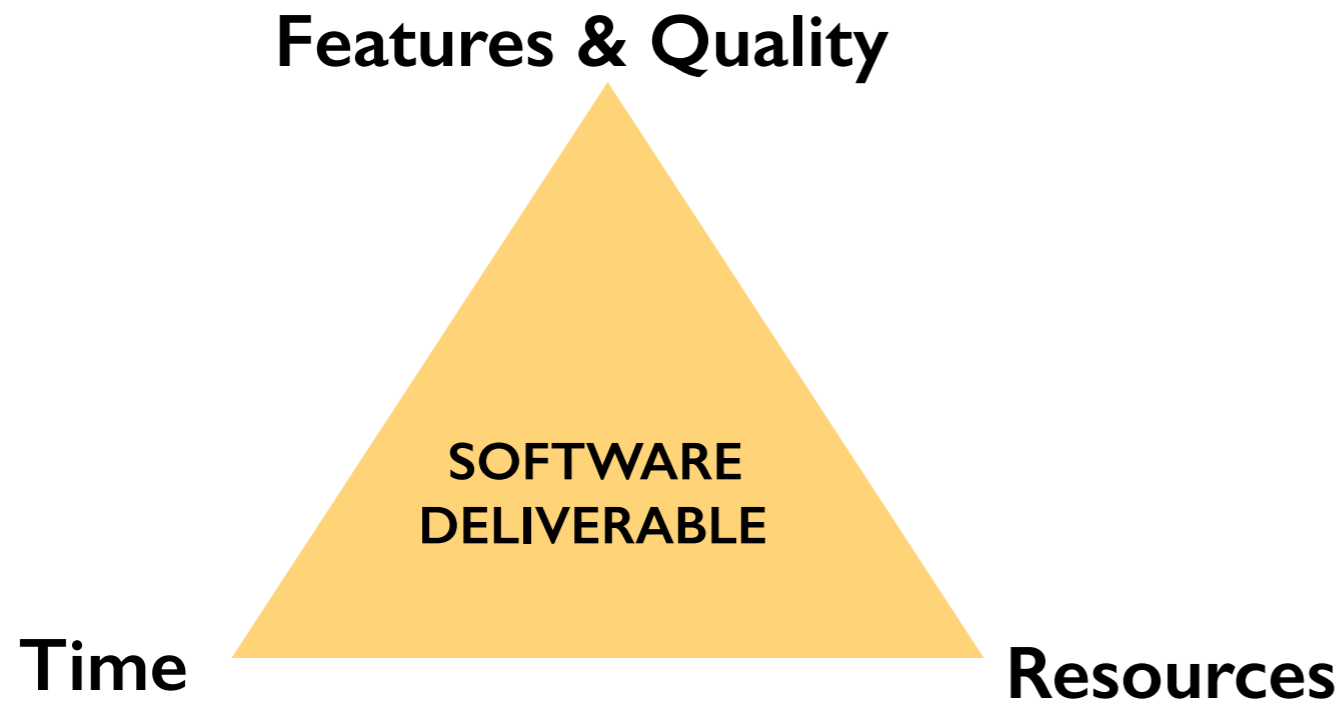
# format

**Course format**

# A typical 403 week

- **Class sessions** to discuss best practices (MWF)
- **Reading assignments** to reinforce the concepts
- **Group project:** to give you hands-on experience with the material
  - **Technical challenges** given the larger project
  - **Social challenges** given the team effort
  - **Frequent meetings:**
    - With your TA (mandatory, every Thursday during section)
    - With your teammates (at minimum, every Tuesday)

# What is a software project?



Projects are a balance of three dimensions, with the goal of producing a successful deliverable.

“Good, fast, cheap ... choose two”

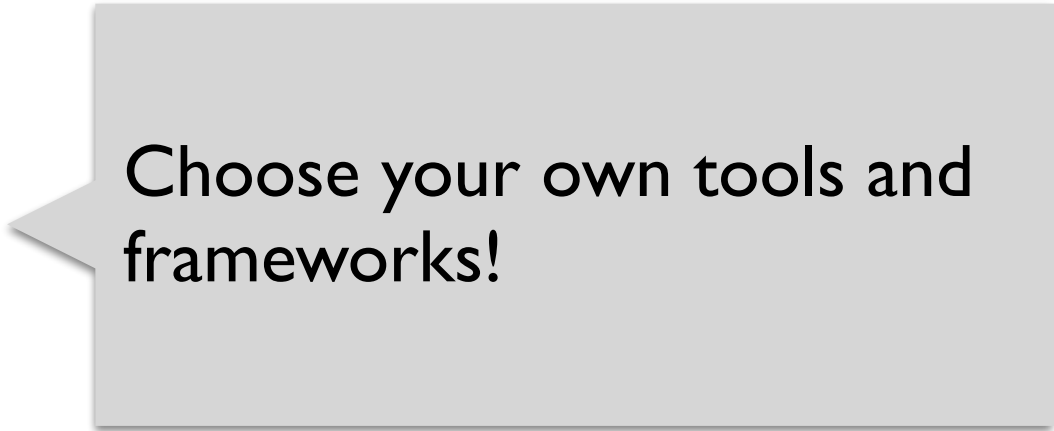
# The project

- You make product proposals
  - And then vote on which products to “fund”
- You’re divided into project teams of 6-8 students
  - Larger teams, larger projects
- You develop your deliverable in stages
  - Reflects modern methodologies for effective development
- A TA will act as your mentor and customer
  - A project is successful only if it satisfies its customer



# Project development stages

- Proposal
- Requirements
- Design
- Implementation
- Testing, validation, verification
- Documentation
- Customer exposure
- Final deliverable



Choose your own tools and frameworks!

# We'll hit the ground running ...

- Your chance to turn a great idea into a product!
- Prepare a 3-slide, 3-minute product pitch in teams of 2
  - Vision and novelty
  - Architecture
  - Challenges and risks
- Schedule for this week and next week:
  - Pair up by **Thu (tomorrow) at noon**
  - Propose by **Fri at 11pm**
  - Pitch your idea on **Mon (lecture) and Tue (section)**
  - Vote by **Tue at 11pm** (rank your choices, self-select teams)



# Project culture

- This is a real project
  - We expect you to work to build a real system
  - To be used by real people
- This is real engineering
  - Take initiative
  - Find and solve problems yourselves
  - Coding is only part of the job
  - Good planning and design, hitting your market, and working well with your team, are all needed for success

# Grading and academic integrity

- Grading
  - Project: 70%
  - Reading assignments: 20%
  - Class participation: 10%
- Academic integrity
  - Simple: *do not cheat!*
  - Do individual work by yourself.
  - Do group work with your teammates only.

# advice

**Lessons from past students**

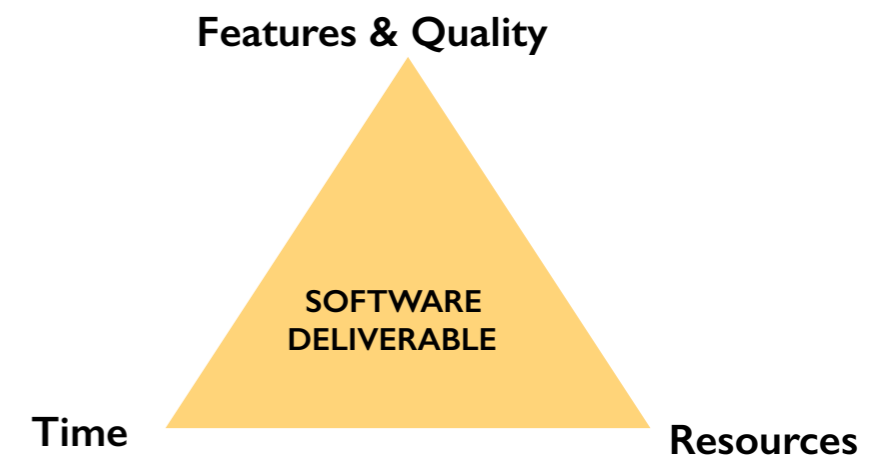
# Communication is key

- Foundation of the success of our team was communication
- Team communication and cooperation are all-important
- Working together (physically) was good
- Well-run and consistently scheduled meetings help a project a lot



# Scheduling

- We often underestimated tasks. If we had spent more time analyzing each task and breaking it down into more manageable chunks our estimated completion times would have been more accurate.
- Get things done early; don't cram at the end
- Remember you can cut features (triple constraint)
- Don't underestimate the difficulty of learning new programming languages, frameworks, and tools



# Testing and coordination

- Thoroughly testing your code and ensuring that your code passes all current tests before submitting is very helpful
- We needed a better upfront testing design
- We learned (through some pain) to ensure to do small, frequent updates and commits. Failing to do this results in merges that can be a nightmare.





**This sounds like a lot of work!  
Why take this course?**

# What's in it for you?

- See how software is produced, from idea to ship to maintenance
- Get exposure to software development practices in use today
- Get experience collaborating in a team toward a common goal
- Be able to articulate and understand technical ideas



# Unique aspects of CSE 403

- Cross-disciplinary nature of the subject
- Larger teams
- Propose and work on your own ideas
- Course staff in the "coach" role
- Mistakes along the way are encouraged, not penalized
- Few clearly right/wrong answers
- Plans always change
- Content: software design, testing, project management, etc.

# Isn't this just like an internship?

- It's not in that internships are
  - Focused on one role in the team (often dev. or test)
  - Requirements, arch, high-level design may be set
  - Less opportunity for reflection
  - Less generalization (such as from reading papers)
  - Mentor may be more focused on results than process and developing you as an engineer
- Internships are complementary to CSE 403
- People who have had internships learn different things in CSE 403, but no less

# CSE403

**[courses.cs.washington.edu/courses/cse403/16au/](https://courses.cs.washington.edu/courses/cse403/16au/)**