Introduction to Software Engineering

CS 320

UMass Amherst, Spring 2013

320 staff

- Instructor:
  Prof. Yuriy Brun
  Office: CS 346
  Office hours: Thursday 3PM – 4PM
  brun@cs.umass.edu
- Teaching assistant:
  Demetre Lavigne
  Office: CS 345
  Office hours: Wednesday 1PM – 2PM
  demetre@cs.umass.edu

CS 529

- A group of six CS 320 alumni
- Experienced, seasoned, well-trained software engineers who have been to the dark side and returned back
- They are here to help lead you …
- … and to learn how to lead teams of developers toward success

Class website

http://www.cs.umass.edu/~brun/class/2013Spring/CS320

- The schedule
- All assignments
- News
- The best place to go for all info related to the class
- We will use Moodle to:
  - submit assignments
  - view grades
  - discussion forum

Course grading

50% Project	Product idea (2%) Requirements specification (6%) Software design (8%) Alpha release (5%) Beta release (10%) User and test report (4%) Final release (15%)
20%	First exam (March 14, evening)
20%	Second exam (Finals period)
10%	Class / lecture participation

Everyone in a group gets the same score on each project assignment.
Your scores on the project work may be adjusted, based on your contribution.
Peer evaluations will occur several times in the semester.

Course goal

To build a large, complex, significant software system

– teams of ~8 students + a leader

– 13 weeks

– more creativity and hard work than you can imagine

Why bother with 320?

care about your craft

Many of you will become software engineers. Why spend your life developing software unless you care about doing it well?

Why bother writing software well?

Why bother writing software well?

- Software is important: It runs our lives!
  - medical devices
  - cars, airplanes, factories
  - try living a day without software
- Software is complex, which leads to poor quality systems (e.g., bugs).

Making SW is hard! Pitfalls to avoid

People	Process	Product	Technology
• Undermined motivation • Weak personnel • Uncontrolled problem employees • Heroics • Adding people to a late software project • Noisy, crowded offices • Friction between developers and customers • Unrealistic expectations • Lack of effective project sponsorship • Lack of stakeholder buy-in • Lack of user input • Politics placed over substance • Wishful thinking	• Overly optimistic schedules • Insufficient risk management • Contractor failure • Insufficient planning • Abandonment of planning under pressure • Wasted time during the "fuzzy front end" • Shortchanged upstream activities • Inadequate design • Shortchanged quality assurance • Insufficient management controls • Premature or overly frequent convergence • Omitting necessary tasks from estimates • Planning to catch up later • Code-like-hell programming	• Requirements gold-plating • Feature creep • Developer gold-plating • Push-me, pull-me negotiation • Research-oriented development	• Silver-bullet syndrome • Overestimated savings from new tools or methods • Switching tools in the middle of a project • Lack of automated source-code control

What will you learn in this class?

Let's brainstorm

Prerequisite

- You must already know how to program!
- Grade of C or better in 220.
- If you got below a B in 220, be prepared to work very hard to stay up to speed.
- Programming is not something you can wing in this class.
  - You can't really do well in Algebra if you can't add.

What is software engineering?

- The process of developing software systems
- From eliciting requirements to producing a software system that meets those requirements
- May involve (among other activities)
  - eliciting and formalizing requirements
  - designing the system architecture
  - developing prototypes
  - working in teams
  - testing
  - implementation
  - debugging
  - validation
  - verification
  - maintenance

What activities comprise software engineering?

- How to organize projects and work with a team
- Developing features based on customer needs
- Evaluating the competition
- Responding to customers' evolving needs in an iterative development process
- Testing and making sure the deliverable is what the customer wanted

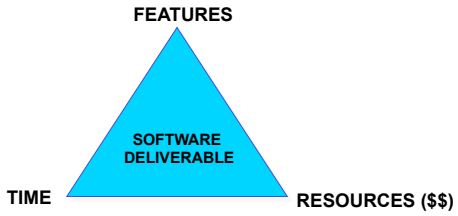
Study of software engineering

Software engineering involves:

1. Processes 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

What is a software project?

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



A typical 320 week

1. Class sessions to discuss best practices.
2. Meetings with your team to make progress on your project and apply what you've learned.
3. Group meetings with the TA (30 min per week) to report progress, get feedback, be guided.
4. Project assignments to solidify what you learn, apply it to the real world, and ensure you're on track with your milestones.

Some discussion sections will be used for meetings, others for presentations.

- You'll meet *technical challenges* given the larger project
- You'll meet *social challenges* given the team effort

The Project

- You make product proposals: next week
  - and then vote on which products you want to do
- We'll divide you into project teams of ~8 students
  - We choose the teams, to mimic the real world
  - Larger teams, larger projects, like industry
- You develop your deliverable in stages
  - Reflects modern methodologies for effective software project development
  - From requirement development through delivery
- The TA act as your customer
  - Ultimately, a project will be successful only if it satisfies its customer

Project culture

- This is a real project
  - We expect you to work to build a real system
  - To be used by real people
- Take responsibility
  - 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

Product idea proposal

- First assignment: Due at noon, Jan 29

http://www.cs.umass.edu/~brun/class/2013Spring/CS320/productIdea.pdf

- Groups of 1 or 2
  - get into groups after class or use the Moodle class discussion forum
- Submit 4 slides:
  - title + names
  - vision slide
  - software architecture slide
  - challenges and risks slide
- 3-minute presentations in class next week

Product idea proposal constrains

1. It may not be a game.
2. It must be based on a client/server networked architecture.
   - The product must have some sort of a server (that may, for example, do computation or store data), and clients that communicate with the server to deliver the functionality to users.
3. It must be installable and runnable by 320 staff working on typical personal computer (think laptop).
4. It must be of suitable size and scope to be feasible in the time allowed, with a team of 8 software engineers.

WikiMap

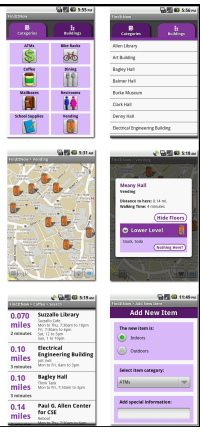
- A visualization interface for Wikipedia
  a GPS for Wikipedia surfing
- Server creates the graph of what pages link to what other pages, weighs the pages and edges with PageRank-like measures
- Clients connect to the map and display a live view of "where you are" as you surf
- Integrates into the Wikipedia interface

Gotta Go Now!

- Android project to find the closest, gender-specific bathroom on University of Washington's campus (built by UW students)
- Uses GPS for location
- Annotated campus maps
- Some algorithms + user input for floor location

Gotta Go Now updated

http://finditnow.googlecode.com



Your Kitchen Manager

- Track your pantry's inventory
- Suggest recipes
- Integrates with Amazon Fresh to buy products
- Predicts shortages
- Integrates with Android to create shopping lists and meal suggestions on the go

Lessons from past students

- 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
- 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. The improvement in quality is unreal.

More lessons

- 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.
- Thoroughly testing your code and ensuring that your code passes all current tests before submitting is very helpful.
- Need better upfront testing design.
- Remember you can cut features (triple constraint).
- It's important not to underestimate the difficulty of learning new programming languages, frameworks, and tools

Your goals of 320

- Be exposed to some of the best software development practices in use today
- Learn how to more effectively collaborate with others toward a common goal
- Understand how software is produced – from conception to shipping and subsequent maintenance
- Develop skills to articulate your ideas and progress
- Understand the issues and tradeoffs involved in making decisions as software engineers and project managers

Get started on the Product idea proposal!

- First assignment: Due at noon, Jan 29

http://www.cs.umass.edu/~brun/class/2013Spring/CS320/productIdea.pdf

- Groups of 1 or 2
  - get into groups after class or use the Moodle class discussion forum
- Submit 4 slides:
  - title + names
  - vision slide
  - software architecture slide
  - challenges and risks slide
- 3-minute presentations in class next week