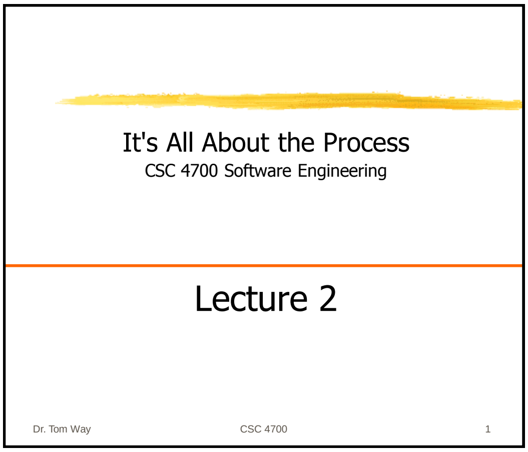
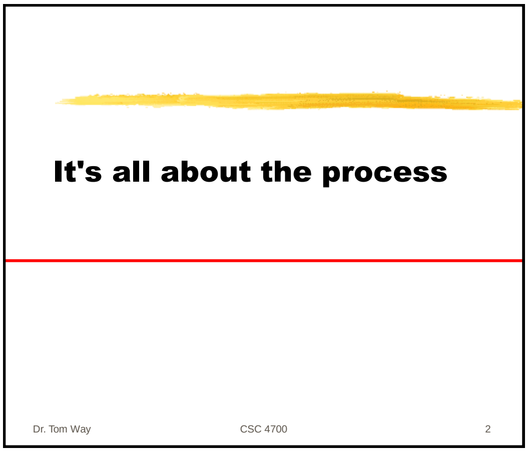


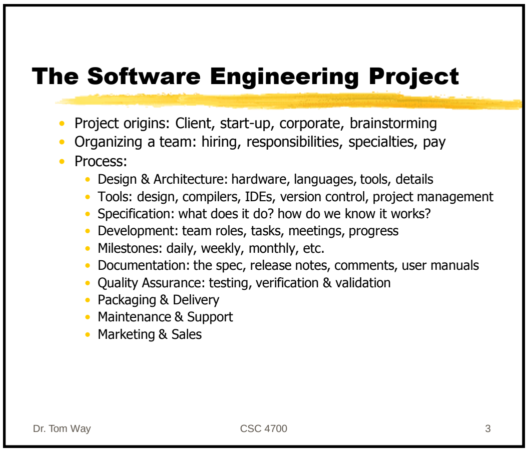
It's All About the Process

CSC 4700 Software Engineering

Lecture 2



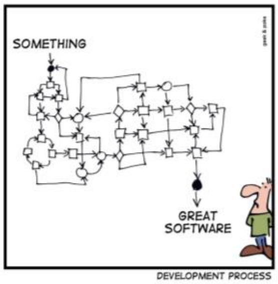
It's all about the process



The Software Engineering Project

- Project origins: Client, start-up, corporate, brainstorming
- Organizing a team: hiring, responsibilities, specialties, pay
- Process:
  - Design & Architecture: hardware, languages, tools, details
  - Tools: design, compilers, IDEs, version control, project management
  - Specification: what does it do? how do we know it works?
  - Development: team roles, tasks, meetings, progress
  - Milestones: daily, weekly, monthly, etc.
  - Documentation: the spec, release notes, comments, user manuals
  - Quality Assurance: testing, verification & validation
  - Packaging & Delivery
  - Maintenance & Support
  - Marketing & Sales

The Big Picture



What is Software Engineering?

- Not exactly a real term
- More than simply writing code
- Process of creating and maintaining software
  - No standards or requirements
  - Varies wildly from project to project
- Skills applicable beyond writing software

Facets of Software Engineering

- Requirements
- Design & Architecture
- Implementation
- Quality Assurance
- Documentation
- Packaging & Delivery
- Maintenance & Support

Team Process

Requirements

- What are we trying to build?
- Who wants us to build it?
- What do they want from the product?
- What do they *actually* want from the product?
- How do we verify we gave them what they asked for?

Design & Architecture

- How do we make it future-proof (as much as possible?)
- How do we make it maintainable for future developers?
- What changes do we know are coming?
- What changes do we *think* are coming?

Infrastructure: Tools

- What technologies should we use?
  - New hotness + lack of maturity
  - Old and boring + well-rounded feature set
- How do we make it quickly?
- Are there existing products that can be leveraged?
  - Are they actively maintained/supported?
  - Can we expand it later?
  - How do we not break it in the future?

Implementation Topics

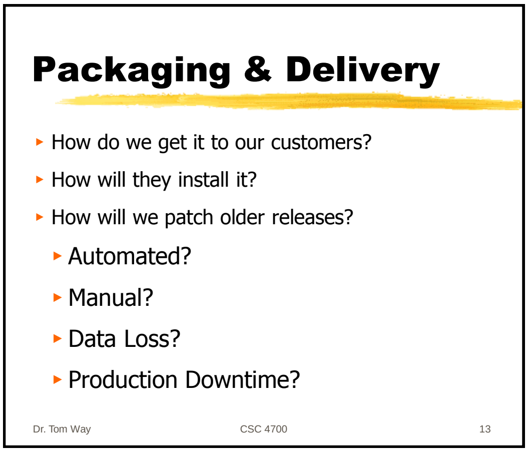
- Advanced topics covered in class
  - Configuration
  - Logging
  - Scalability
  - Concurrency
  - Technical Debt
  - Internationalization

Quality Assurance

- Does it work?
- No, really, does it actually work?
- How does it hold up under…
  - extended use?
  - large scale?
  - dumb users?
- How do we not kill our QA team in the process?

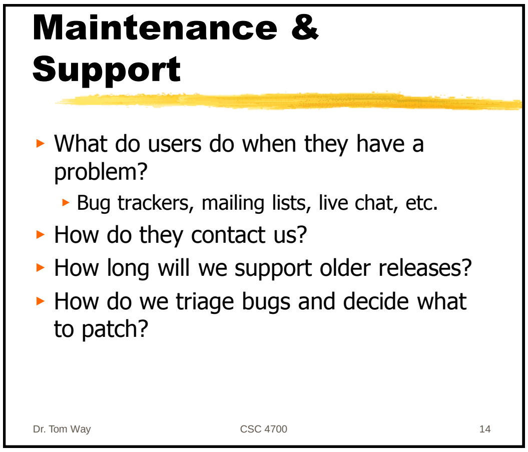
Documentation

- Will users know how to use it?
- Is the code documented so I can read it six months from now?
- Is the code documented so someone else can read it six hours from now?
- Are the APIs documented for integration partners?
- Is the documentation still accurate?
- Was it ever accurate in the first place?



Packaging & Delivery

- How do we get it to our customers?
- How will they install it?
- How will we patch older releases?
  - Automated?
  - Manual?
  - Data Loss?
  - Production Downtime?



Maintenance & Support

- What do users do when they have a problem?
  - Bug trackers, mailing lists, live chat, etc.
- How do they contact us?
- How long will we support older releases?
- How do we triage bugs and decide what to patch?