

CSE 403: Software Engineering, Winter 2016

courses.cs.washington.edu/courses/cse403/16wi/

Conclusion

Emina Torlak

emina@cs.washington.edu

Outline

- Final release, demo, exam
- A brief recap of CSE 403
- Beyond CSE 403



final release, demo, exam

Logistics and dates for the final release & demos

- Final release on **Tuesday, March 08**, at 11pm
 - Final version of your product!
 - SRS revision
 - Requirements & schedule postmortem
- Final product demos on **Wednesday, March 09**, and **Friday, March 11**, in class
 - Must include all team members who have not presented yet
- Individual reflections on **Friday, March 11**, at 11pm



Logistics and dates for the final exam

- Final exam on **Monday, March 14**, at 8:30am in **EEB 037**
 - 110 minutes.
 - Closed book, notes, laptop, phone, and neighbor.
 - Cumulative.
 - Anything discussed in Lectures 1-18 is fair game.
 - See prior years' exams for practice.

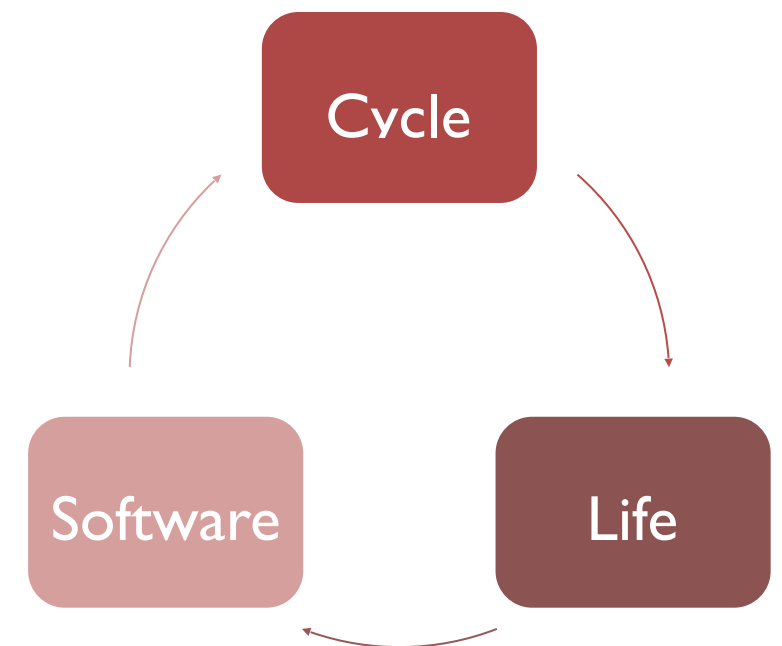


recap

a brief recap of CSE 403

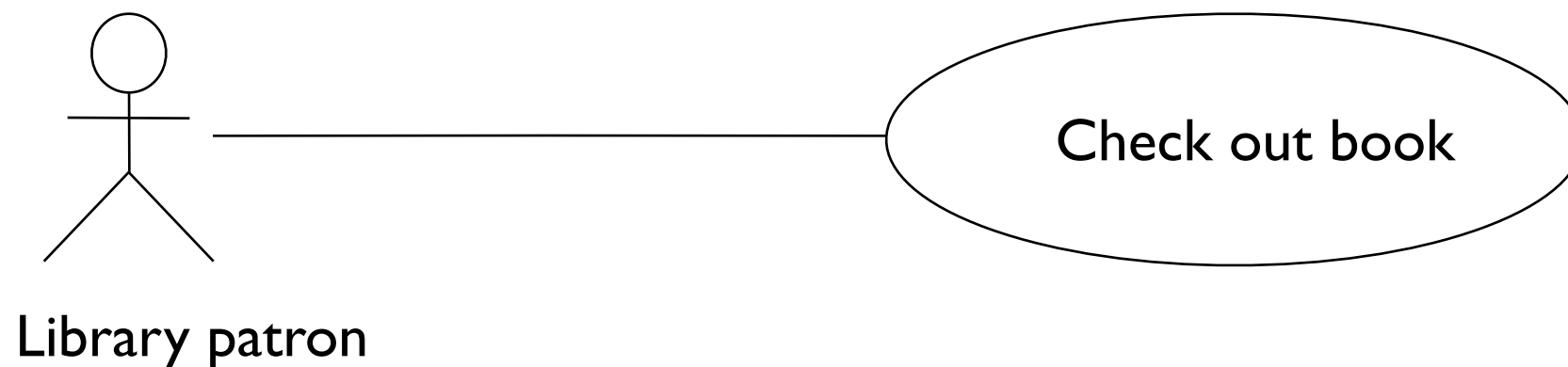
Software lifecycle

- Determines the order for Requirements, Design, Implementation, Testing, and Maintenance.
- Goal: Perform work as early as practical
 - Costly to discover bugs or make changes late
 - Costly to make decisions too early
 - Costly to do tasks multiple times
- In CSE 403, we followed an iterative process



Requirements

- “What” not “how.”
- Reflects user rather than developer view of the system.
- A common technique for expressing requirements: **use cases**.
- Get feedback early (example: paper prototype).



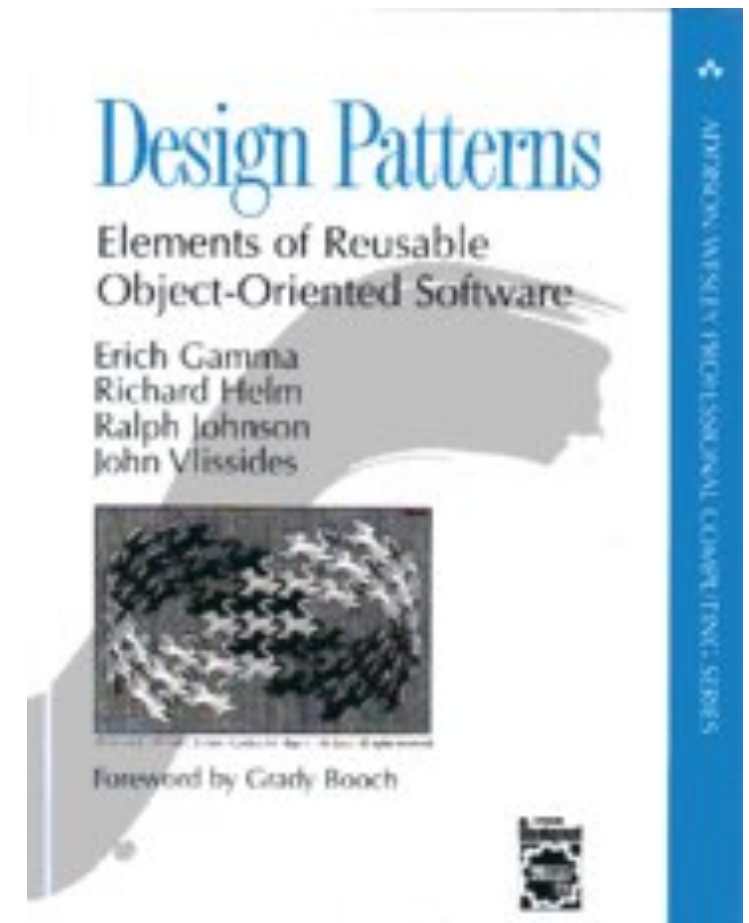
Architecture

- Provides a high-level framework to build and evolve a software system.
- Modules for logical units of computation
 - Minimize coupling, maximize cohesion.
- Draw it as a **UML class or sequence diagram**
 - Key purpose: to communicate to others
- Interactions are part of the architecture



Design patterns

- Vocabulary of program development:
 - A known solution to a known problem.
 - Don't reinvent the wheel!
- Many kinds of design patterns:
 - Creational
 - Structural
 - Behavioral
 - Concurrency
 - ...



Focus on modularity, abstraction, and specs

- No one person can understand all of a realistic system.
- Modularity permits focusing on just one part.
- Abstraction enables ignoring detail.
- Specifications and documentation formally describe behavior.
- Modularity, abstraction, and specifications help to understand/fix errors
 - Or to avoid them in the first place!

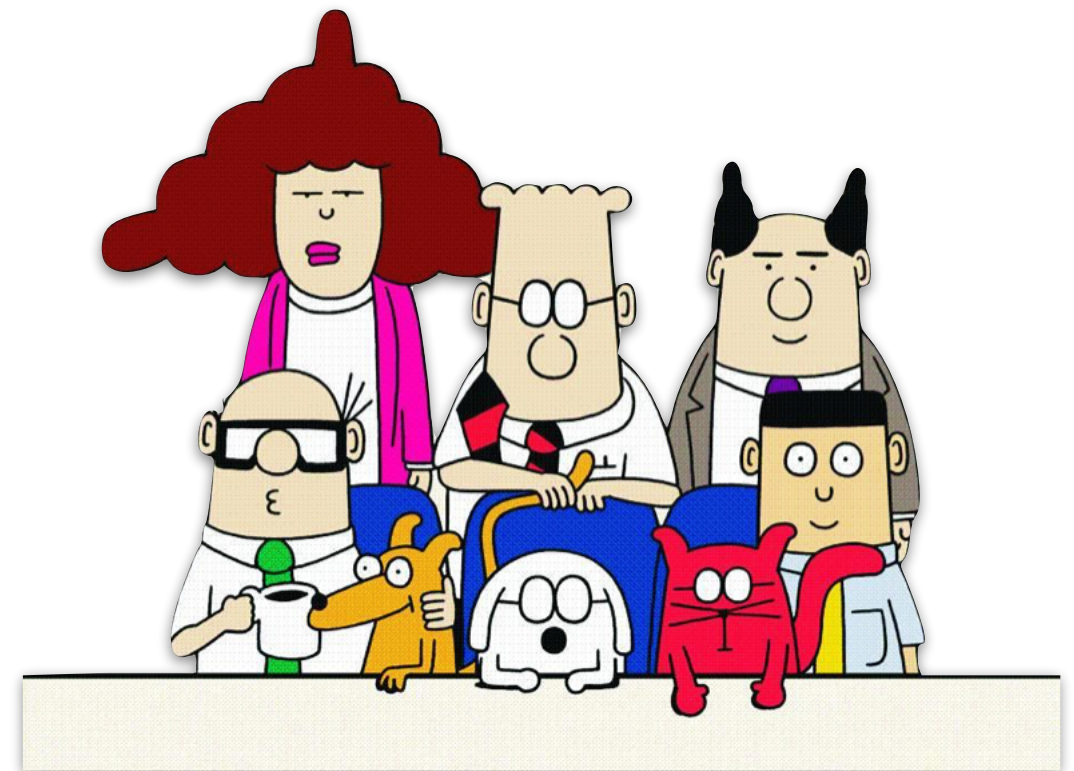
Process

- Needed to keep your project under control:
 - Specification
 - Schedule
 - Source control
 - Automated builds and test
 - Bug database
 - Bug fixes before features
 - Hallway usability testing



Teamwork

- Organization
 - Dominion vs communion
- Decision making
 - Write it all down, be specific, prioritize
- Motivation
 - Practice positive reinforcement
- Communication
 - Face-to-face is most effective



Testing, static analysis, and symbolic execution

- Increase software quality.
- Testing techniques
 - Unit and system testing
 - Black and white box testing
 - Integration and performance testing
- Static analysis
 - Soundness vs Completeness
 - Abstract values
 - Transfer functions
- Symbolic execution
 - Symbolic values
 - Path conditions
 - Tools



Code reviews and refactoring

- Code reviews improve code quality, teamwork, knowledge, and skills.
- Code reviews can also help identify opportunities for refactoring.
- Refactoring improves software's design
 - to make it more extensible, flexible, understandable, performant, ...
 - but every improvement has costs (and risks)



future

beyond CSE 403

What you have learned and will learn

- Compare your skills today to the beginning of the term
 - Bottom line: Your project would be **easy** for you
- Your next project can be much more ambitious
- You will continue to learn
 - Building interesting systems is never easy, like anything worth doing.
 - Practice is a good teacher
 - Requires thoughtful introspection
 - Don't learn **only** by trial and error!



Tell us what you think!

- Please complete the [course evaluation](#) form
 - Useful to future students
 - Useful to course staff
 - Useful to the department



Build amazing things!

- System building is fun!
 - It's even more fun when you build them successfully.
- Pay attention to what matters
 - Use techniques and tools of CSE 403 effectively.
 - Above all, use good taste and engineering judgement.

