Announcements, 3/2/2023

Today: Software lifecycle

Break around 11:15am

2 minute project pitch

Reading discussion

- Anything surprise you?
- Anything you find particularly interesting?
- Anything you disagreed with?

Outline

- Discuss software lifecycle
- (if time) Overview software development models

Acknowledgements

Some of these slides are from Prof. Alex Kuhn, and some are based on the lecture notes from Prof. Frank Tip and Prof. Michael Weintraub at Northeastern University and Prof. Emina Torlak at University of Washington and Prof. Scott Stoller at Stony Brook University.

Software development lifecycle

Software development lifecycle

- Projects will go through a series of phases:
  - Requirements analysis
  - Design
  - Implementation
  - Testing
  - Maintenance

Software development lifecycle

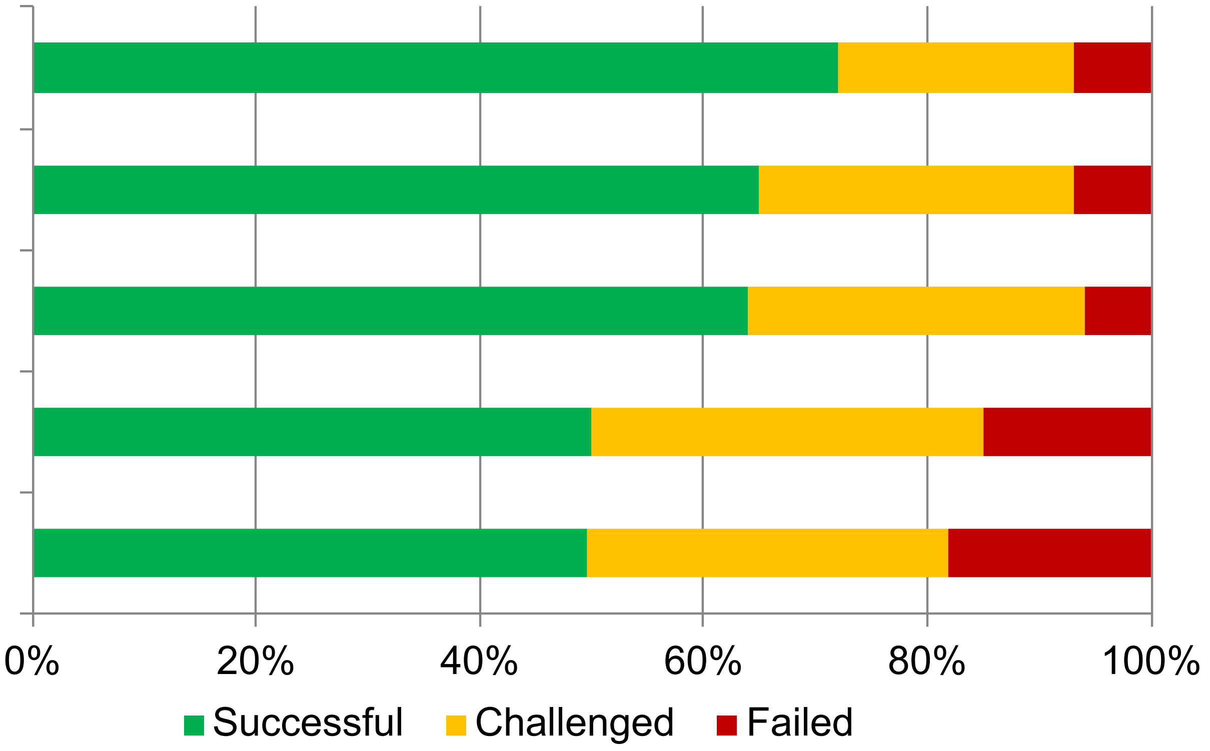
- Each phase has goals:
  - Determine the set of work to be done
  - Produce something that can be reviewed
  - Determine next steps
- Can spend days, months, or years in each phase

Why use software development models?

- Software is complex!
  - Windows XP had 45 million lines of code
  - Google’s total codebase was over **2 billion** lines of code in 2015

Why use software development models?

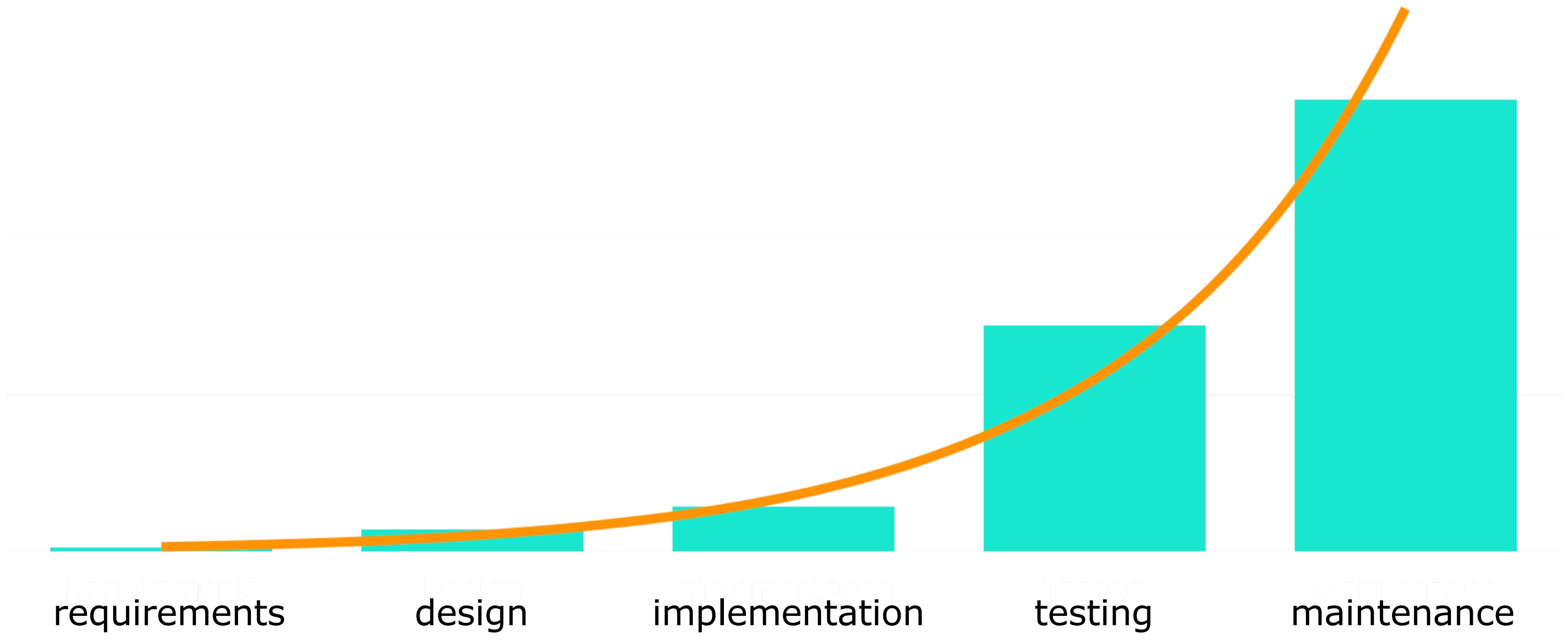
Many software projects fail or go over budget and time estimates



Ambysoft 2013 IT project success rate: from http://www.ambbysoft.com/surveys

Why use development models and processes?

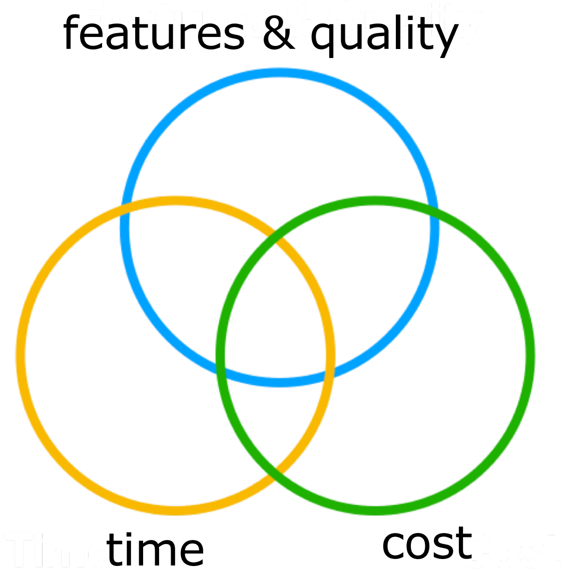
Fixing bugs later is more costly



Boehm’s first law

Errors are **most frequent** during the **requirements** and **design** activities and are **more expensive** the later they are removed.

Projects require balancing constraints



“Good, fast, cheap – Pick two”

No process — tradeoffs

Pros: Fast, work on whatever you want. Can be effective with small individual projects.

Cons: May ignore important tasks or requirements. Hard to review. Difficult to scale with more people. May develop less reusable code and more technical debt.

With process — also tradeoffs

Pros: Provides structure, easier to manage project and teams of people, forces keeping track of high level goals and ensuring they are met.

Cons: Takes time, can be frustrating and lead to artificial constraints and compromises. Sometimes people put the process before the product.

Project discussion

Project teams

- Team sizes are between 2-4 students
    - I expect far more from a 4 person group than a 2 person group
- If the class grows or shrinks in the next week, teams will need to be adjusted

Project review

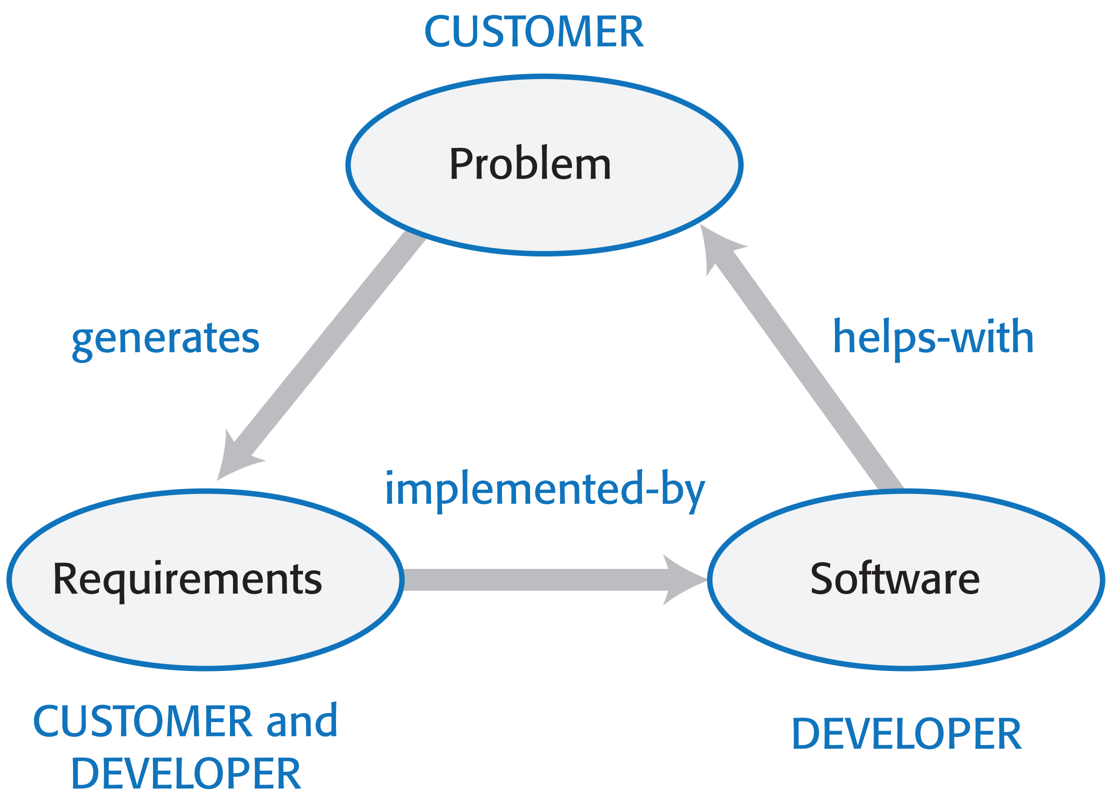
- Creating a substantial software project
- Have until next Thursday (3/9) to decide project
    - You pick project and technologies used
      (within the constraints / instructor approval)
- Opportunity to make a portfolio project – useful when applying for jobs!

Next steps

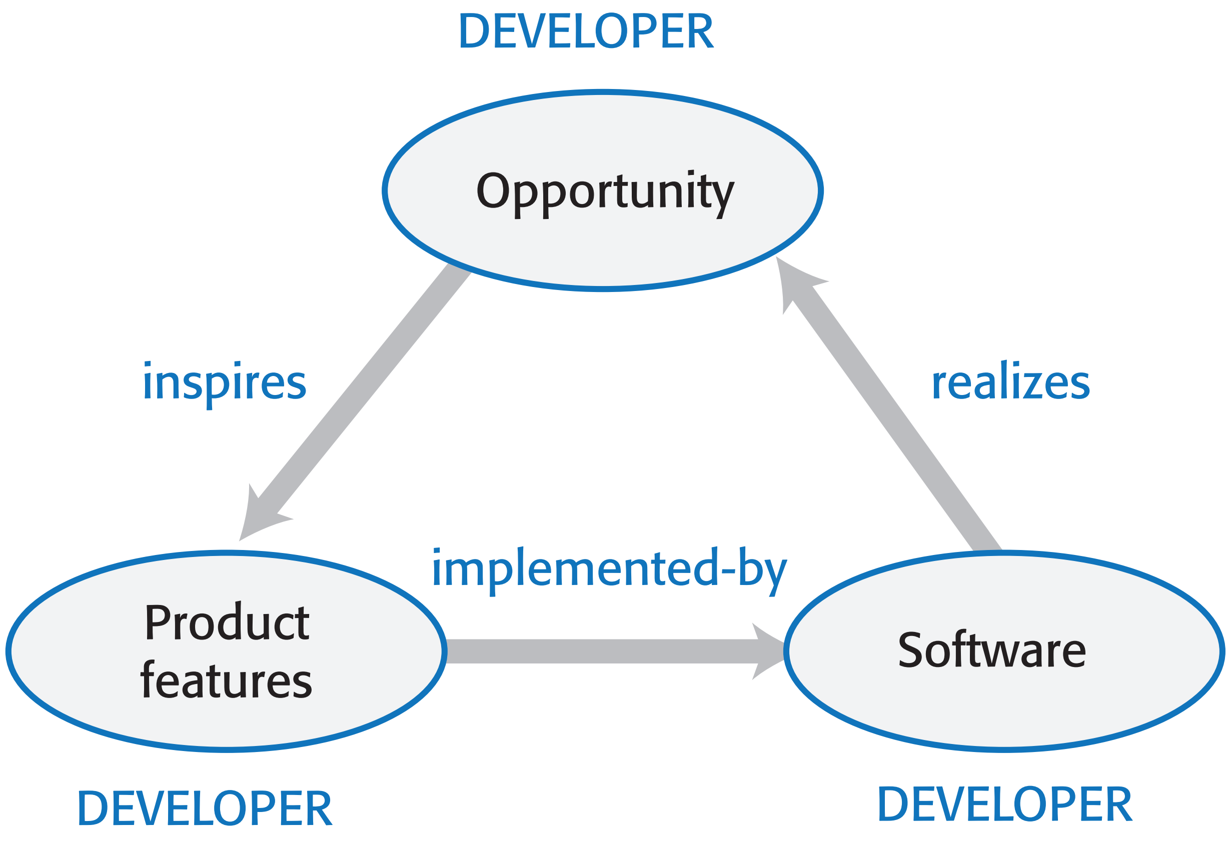
- "Create Teams" assignment posted on the class web
    - Decide on project
    - Create GitHub Repository with project description and team
    - Present your group project statement **on March 7**

Different projects, different approaches

Project-based software engineering



Product software engineering

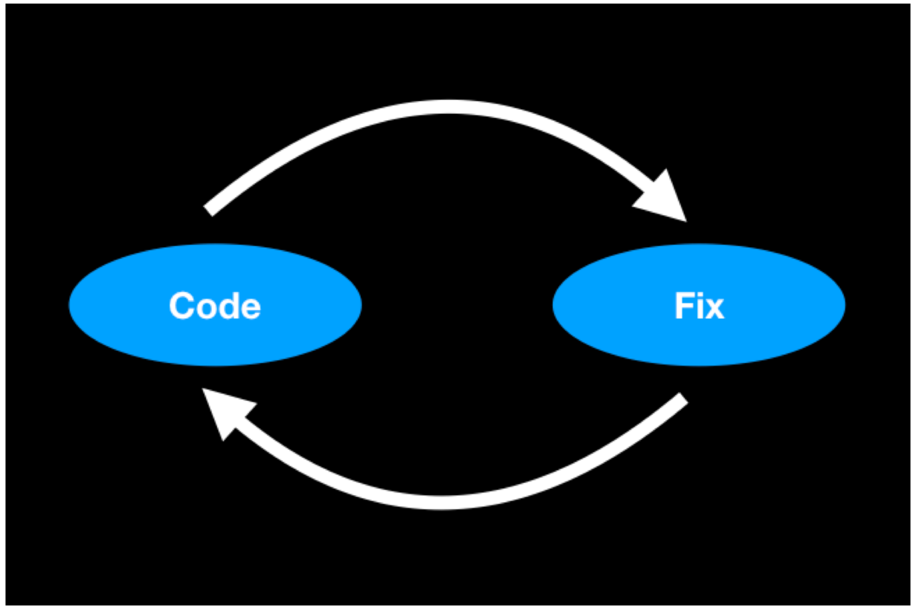


Note – the developer still should be constantly working with users

Software development models

Code and fix

Write code, fix it when it breaks or client is not satisfied

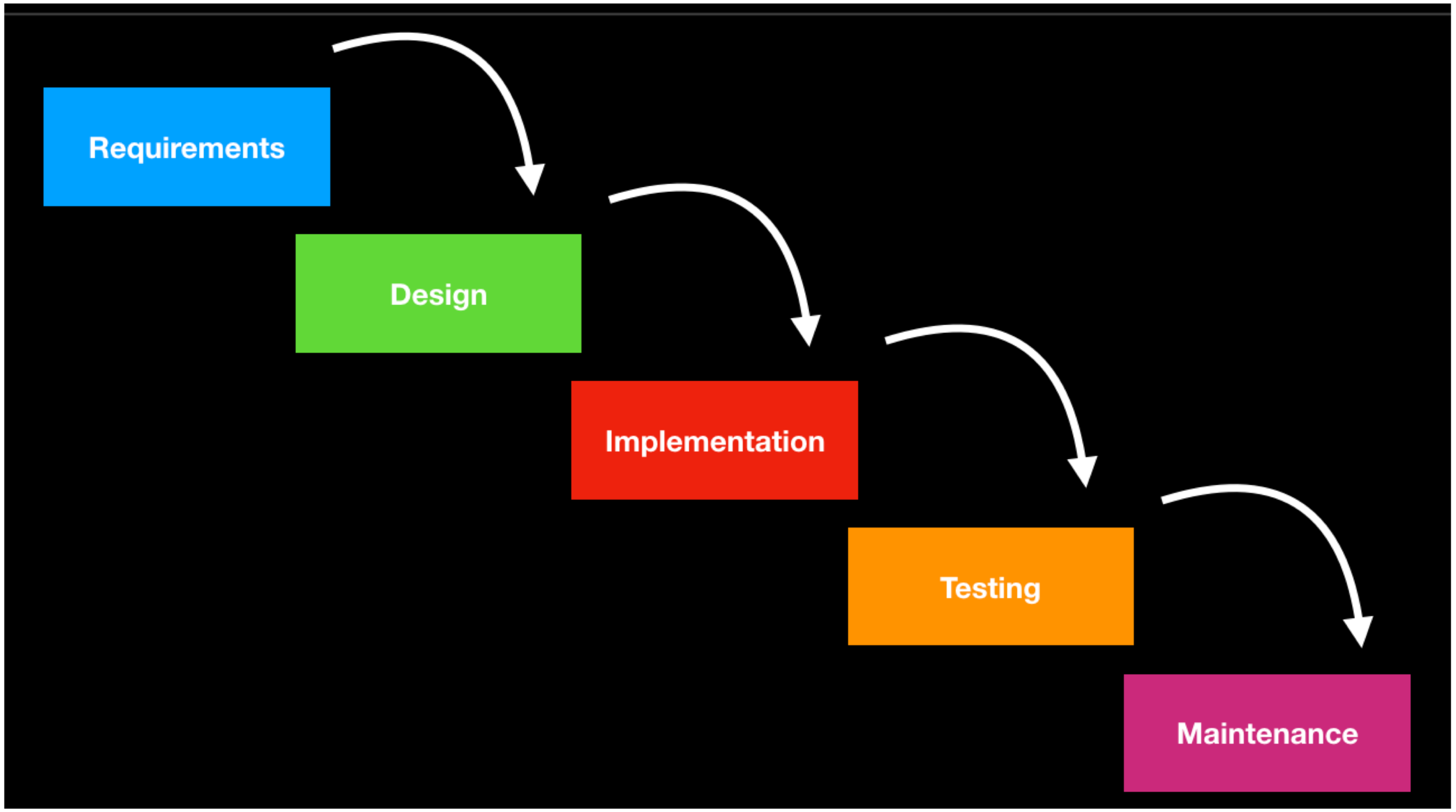


Code and fix

- Pros:
  - Little process overhead, can see progress quickly
  - Works for small projects or prototypes

- Cons:
  - Hard to distribute tasks / work as a team
  - Cannot deal with complexity
  - Unclear on what is being built and the schedule. Hard to assess progress or make good design decisions

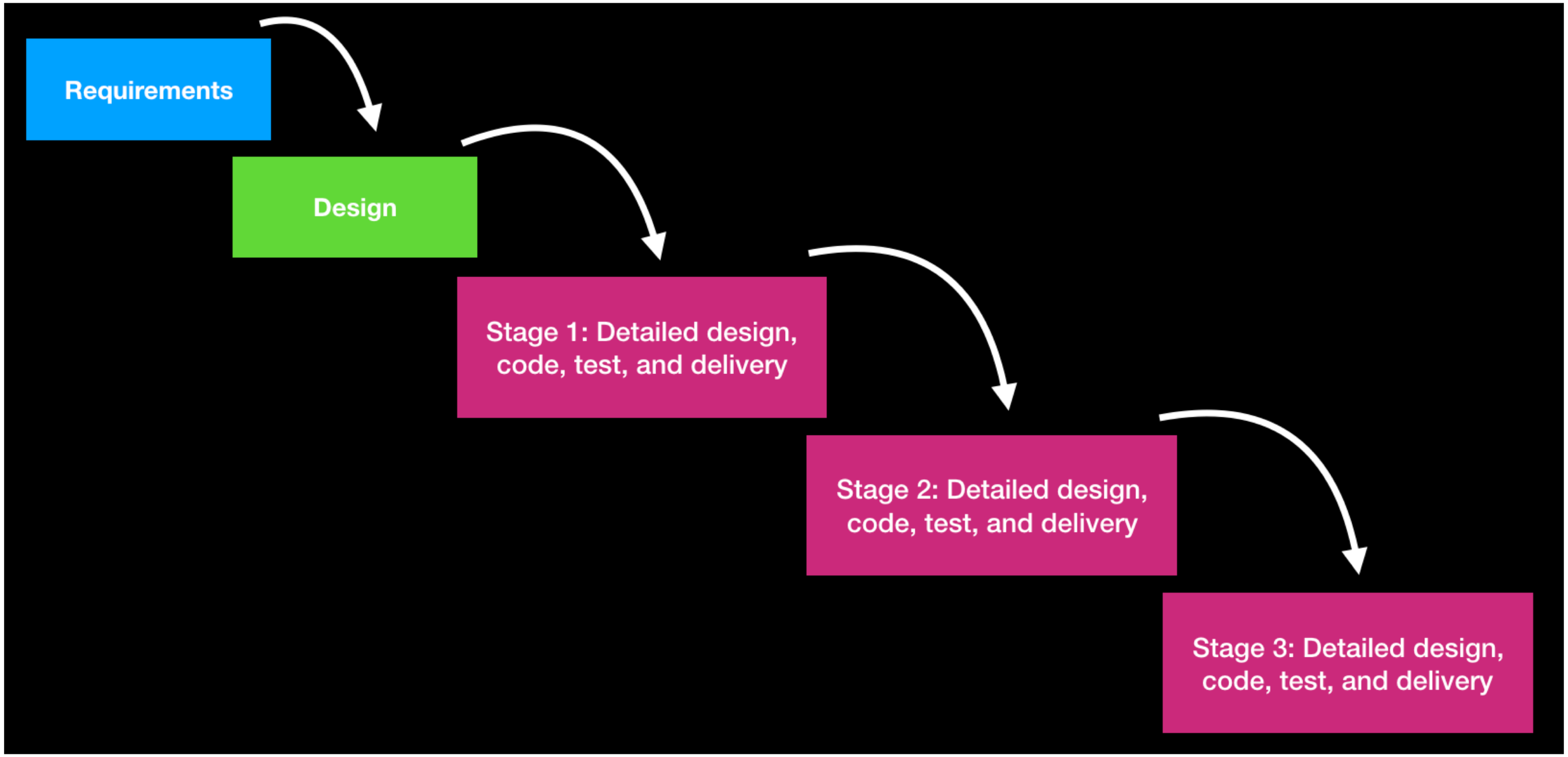
Waterfall model



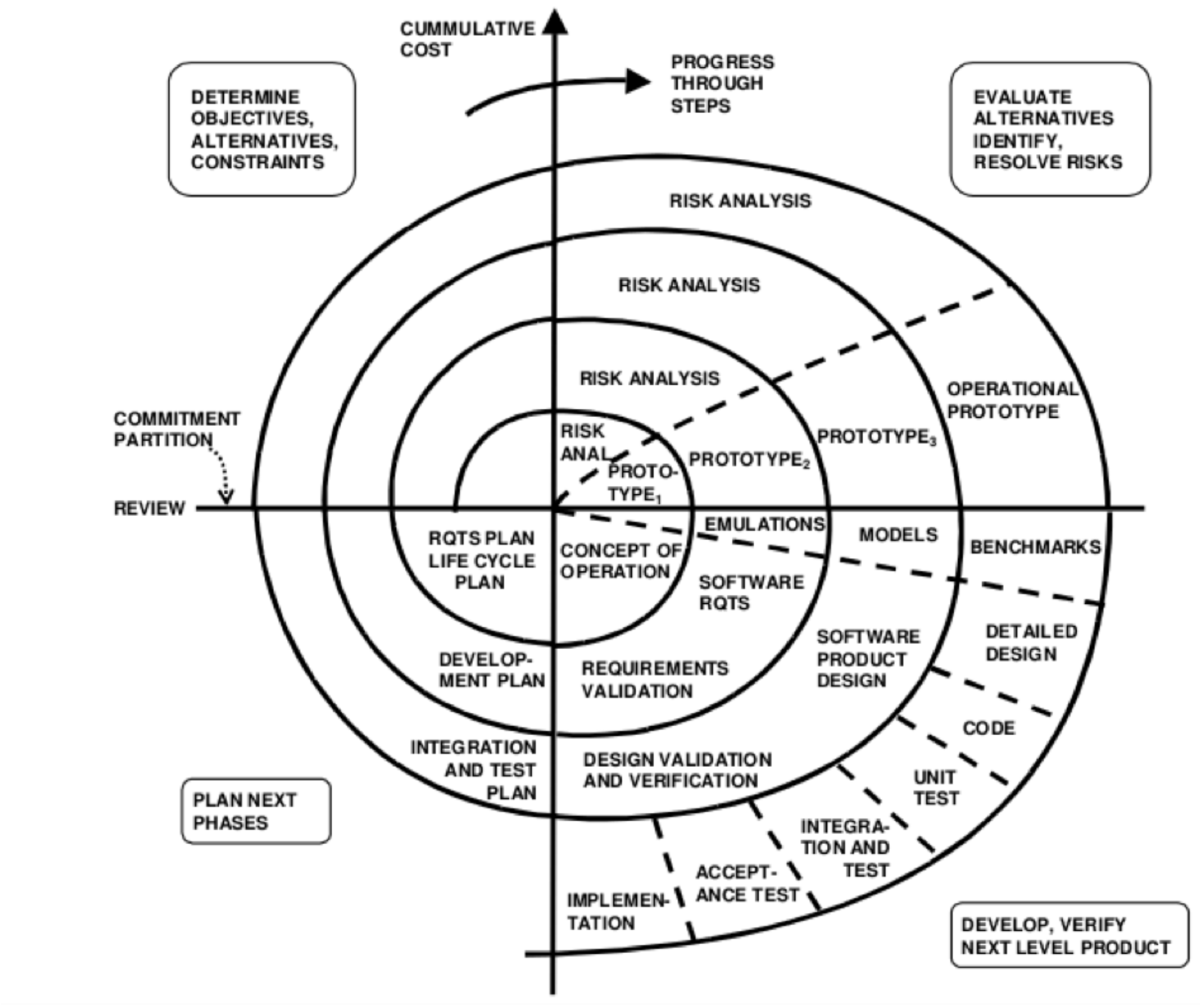
Waterfall model

- Pros
  - Suitable if requirements are well understood, complex, and will not change
  - Simple, easy to follow and understand model
  - Can be easier for larger and more inexperienced teams to work using this model
- Cons
  - Hard to specify all requirements upfront
  - No visible progress on the software until the end
  - Not flexible – cannot change requirements (or design) in the later phases
  - Potential to build products that may not solve the customer’s actual needs

Staged delivery


Requirements
Design
Stage 1: Detailed design, code, test, and delivery
Stage 2: Detailed design, code, test, and delivery
Stage 3: Detailed design, code, test, and delivery

Spiral model



From Cooperative Software Development by Andy Ko. Used with permission.

Spiral model

- Pros
  - Reveals unseen problems early and cheaply
  - Project risk decreases over time
  - Useful when requirements are changing or unknown

- Cons
  - Complicated. Requires lots of planning and management
  - Developers need to be able to assess risk
  - Customer or contract must be flexible

Agile software development

Manifesto for agile software development (2001)

Individuals and interactions	over	Process and tools
Working software	over	Comprehensive documentation
Customer collaboration	over	Contract negotiation
Responding to change	over	Following a plan

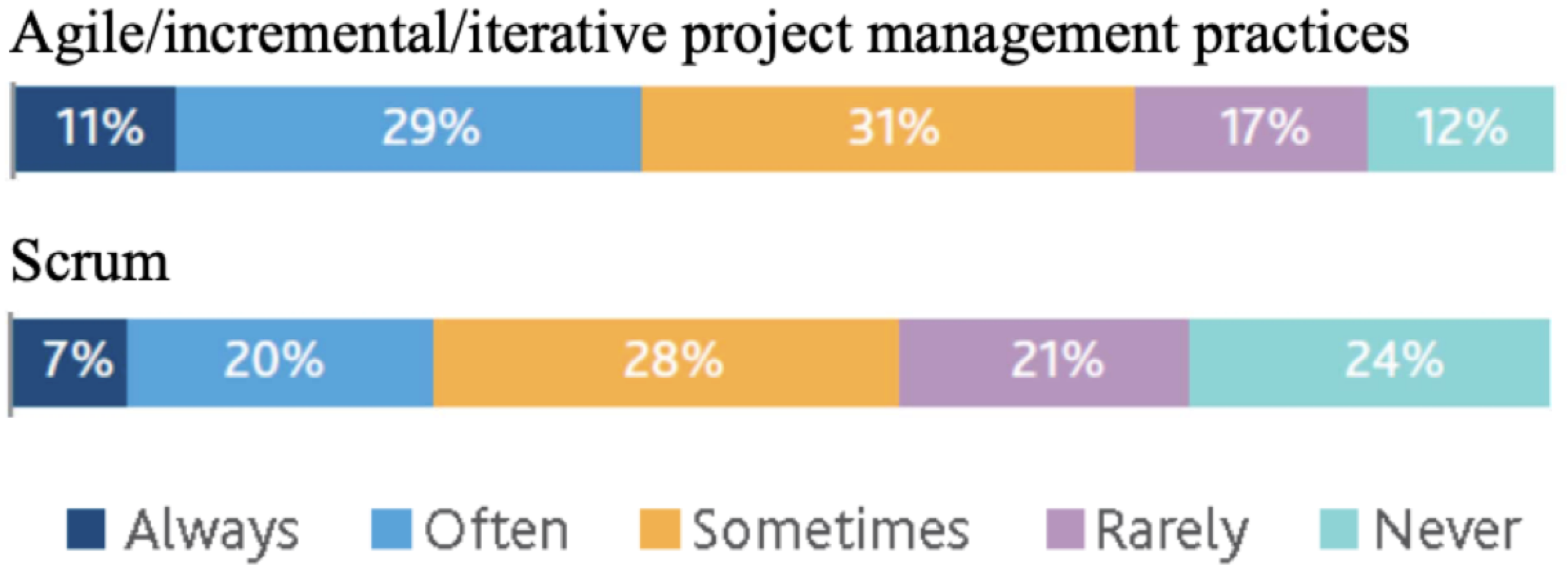
Agile software development

- Accept unpredictability and volatility (of requirements) as inevitable. Emphasize adaptability.
- Make a series of short-term plans, not a (detailed) long-term plan.
- Development proceeds in short iterations (1 to 4 weeks).
  - Plan one iteration at a time.
  - An iteration may involve a full lifecycle (requirements analysis, design, coding, testing) for some feature or a single step in the lifecycle for some feature.
  - Coding and testing (and design) are tightly interwoven.
- Solicit customer feedback after each iteration.
- Emphasize face-to-face interaction with customer and between developers. De-emphasize written documentation.

Adoption of agile methods

2017 Pulse of the Profession survey of 3,234 project management practitioners, by the Project Management Institute.

- This includes all kinds of projects, not just software projects
- How often does your organization use each of the following?



Source: https://www.pmi.org/-/media/pmi/documents/public/pdf/learning/thought-leadership/pulse/pulse-of-the-profession-2017.pdf

Scrum: agile framework

- Scrum is an iterative and incremental agile software development methodology for managing product development.
- Main roles:
  - Development team: About 3 to 9 software developers.
  - Scrum master: Project manager. Ensures that the team adheres to the Scrum process; handles most of the team's external interactions with management; etc.
  - Product owner: The client. Defines and prioritizes requirements in the product backlog.

Read more at: https://en.wikipedia.org/wiki/Scrum_(software_development)

Scrum: product backlog

- Product backlog: a list of tasks, with estimated durations.
- Some tasks correspond to required functionality of the product.
  - Functionality may be described as a user story, use case, or other format.
- Other tasks correspond to non-functional requirements, changes to existing features, bug fixes, etc.

User story

- User story: a brief informal user-oriented description of a feature, usually with acceptance criteria (conditions it must satisfy) and story points (estimate of implementation effort).
- Examples: (sub-items are acceptance criteria)
  - A student can view a list of all colleges in a selected city, with information about each one.
    - The listed information should include the college name, address, and admissions office email address.
  - A new user can register for an account by providing his/her name, username, and email address.
    - Password should be entered in two fields and compared.
    - Password must contain both letters and digits.
    - If the username is already taken, the system reports this, and the new user must select a different username.

Scrum: sprints

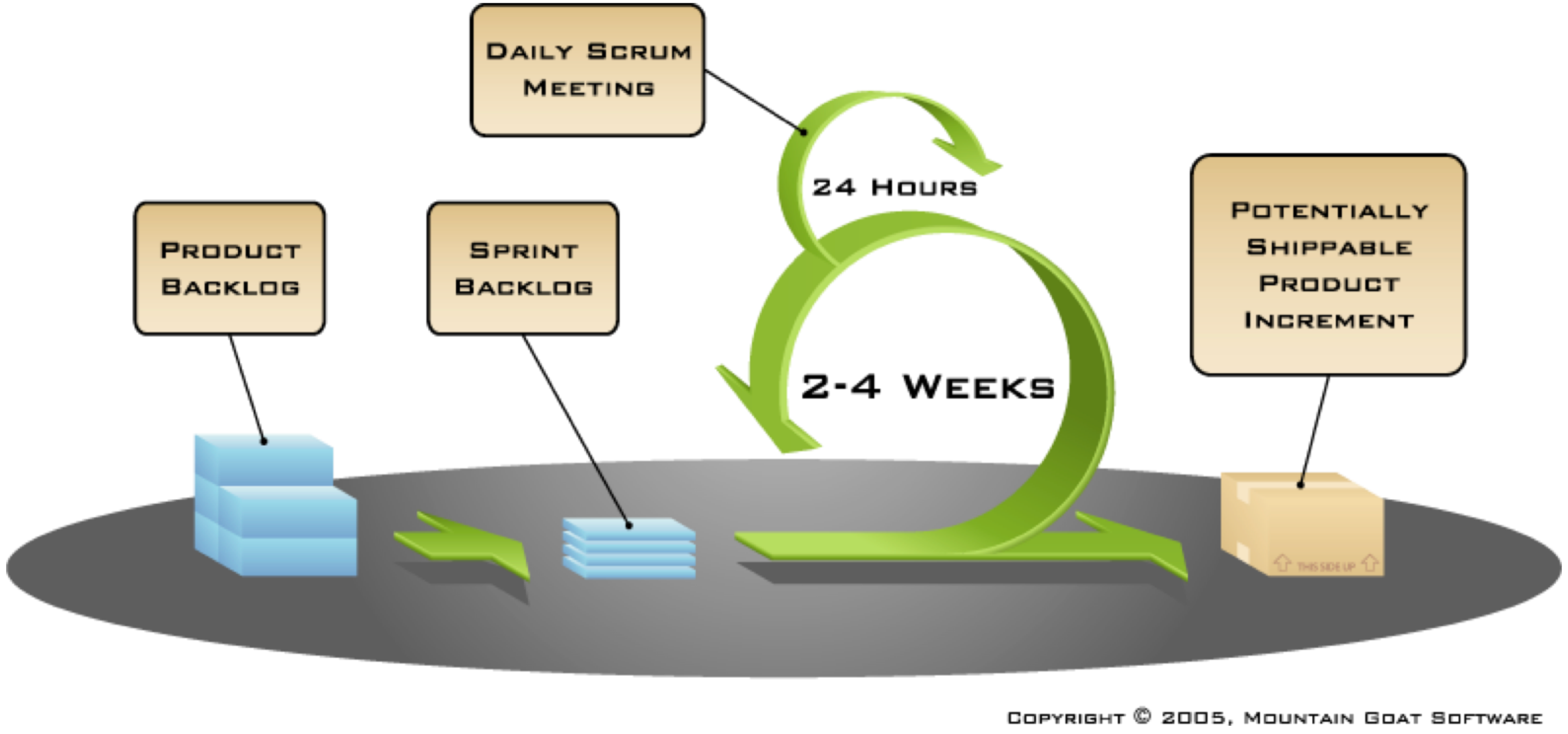
- Development is done in sprints (increments).
- Sprint planning meeting is held at the start of each sprint.
- Sprint backlog contains tasks selected from product backlog for one team to work on during a given sprint.
  - Tasks in the product backlog may get decomposed into smaller, more manageable chunks (e.g., 8 to 16 hour tasks) that actually get added to the sprint backlog.

Scrum: sprints

- Daily scrum meeting is held each day of the sprint. Every team member answers the following questions:
  - What did you do yesterday?
  - What do you plan to do today?
  - Anything keeping you from working / blocking you?

- Sprint review (to present results) and sprint retrospective (to reflect on and improve the process) are meetings held at the end of each sprint.

Scrum: Sprint Diagram


Daily Scrum Meeting
24 Hours
Product Backlog
Sprint Backlog
2-4 Weeks
Potentially Shippable Product Increment
This side up
Copyright © 2005, Mountain Goat Software

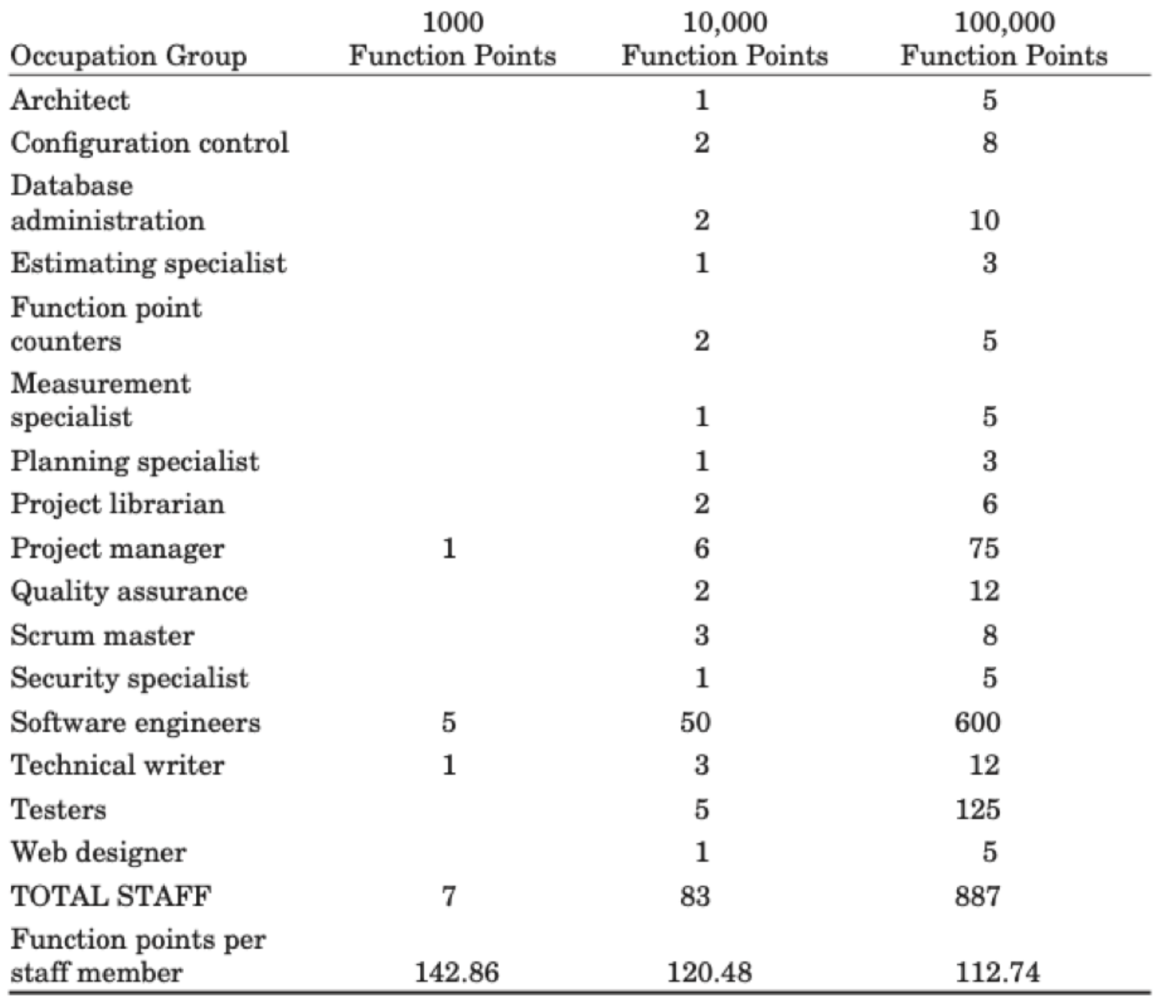
Source: https://www.mountaingoatsoftware.com/agile/s

Agile models tradeoffs

- Pros
  - High customer interaction – handles unknown or changing requirements
  - Constantly see progress
  - Risks and issues can be revealed quickly

- Cons
  - Less predictable – works well in a culture that can handle some chaos and uncertainty
  - Can be harder to execute with inexperienced developers
  - Less suitable for critical or long-lived systems due to minimal documentation
  - Harder to scale across large teams

Project staffing varies by project size



Occupation Group	1000 Function Points	10,000 Function Points	100,000 Function Points
Architect		1	5
Configuration control		2	8
Database administration		2	10
Estimating specialist		1	3
Function point counters		2	5
Measurement specialist		1	5
Planning specialist		1	3
Project librarian		2	6
Project manager	1	6	75
Quality assurance		2	12
Scrum master		3	8
Security specialist		1	5
Software engineers	5	50	600
Technical writer	1	3	12
Testers		5	125
Web designer		1	5
TOTAL STAFF	7	83	887
Function points per staff member	142.86	120.48	112.74

From Software Engineering Best Practices by Capers Jones, p. 88.

Choosing a development model

- Depends on project and requirements
  - Productivity and success will vary depending on model and processes used
- Some criteria to consider:
  - Project goals and scope, quality desired, budget & schedule constraints, customer involvement.
  - Further affected by needs to see visible progress, predictability, and handling risk management.
- Often people mix and match approaches – no “right” answer

Which model would you use?

- Mobile photo filter app for consumers
- System for launching a satellite
- New government tax system that replaces existing system

This class

- Using aspects of different models:
    - Initially define requirements and design
    - Requirements and design can change (within reason) as progress
    - Weekly milestones and demos to show progress
    - Weekly scrums with status updates

Questions?

Group team building assignment

- Team building project to get familiar with your team and technologies

- I emailed it to you – Due March 9
  - I recommend to start on it now

- Also posted a brief reading on tips for working successfully as a team

Next reading

- Read Mythical Man Month Prefaces, Introduction, & Chapter 1.
- As you read, consider:
  - What do you feel were the 1-2 key takeaways?
  - What 1-2 points were confusing or would you like to learn more about in class?
- Tips on reading:
  - Spend a minute to reflect and try to understand the opening image and quote for each chapter – the author refers to these ideas in the chapter, so understanding it will be helpful
  - The author uses the word “desiderata” to mean what they desire to build (it is defined as “something that is needed or wanted”)