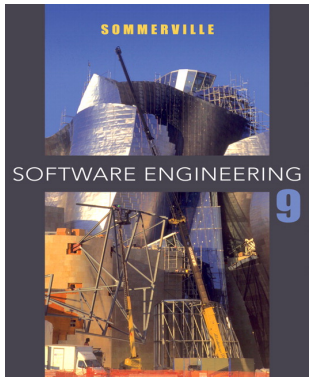
RUP, Agile, XP

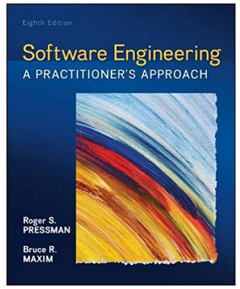


Sommerville Book:

Chapter 2.1.2, 2.3.3, 2.4, 3.1, 3.3

Pressman & Maxim Book:

Chapter 4.1.2, 4.1.3, 4.3, 5.1—5.4



Administrative stuff

- TA office hours
  - Lefan, 2—3:30pm T @ CSIL1
  - Yuxi, ??pm Mon/Wed @ CSIL? ??
- Class enrollment
- Warmup project

Software Development Process

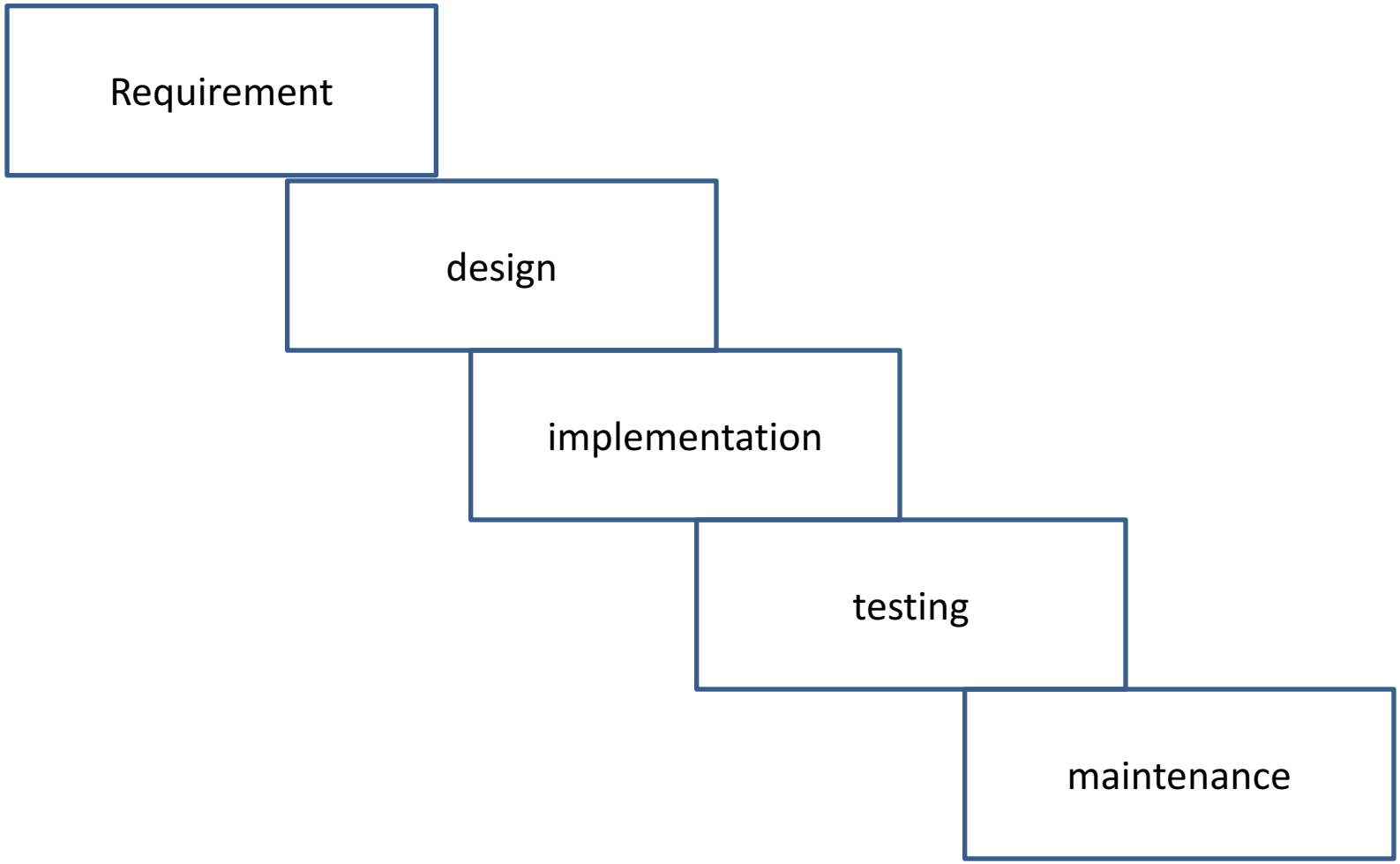
Software Development Process

- In software engineering, a **software development process** is the process of dividing software development work into distinct phases to improve design, product management, and project management. It is also known as a **software development life cycle**. ---- Wikipedia

Outline

- The problems of waterfall
  - How to improve waterfall?
- RUP
  - Phases
  - (iterative) Activities
  - UML
- Agile
  - XP

Waterfall model



What are the problems?

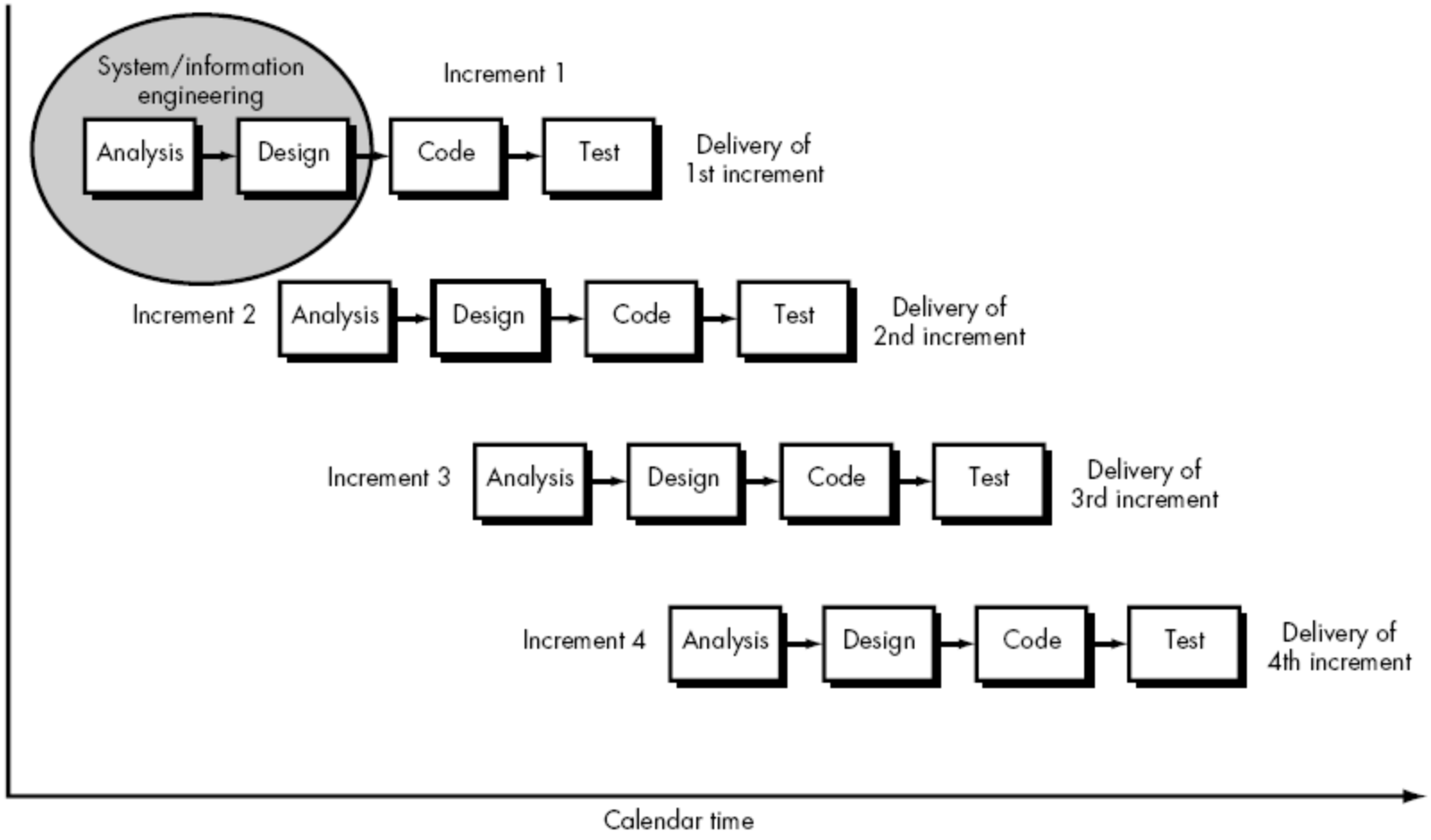
- 1. difficult to handle changes
- 2. take long time to deliver
- 3. expensive to fix errors
- 4. difficult to estimate/planning

How to deliver faster?

Incremental process

- Produce core products first
- Produce further refinements in follow-up releases

Incremental process



Example

- Text editor

- Class registration system

How to handle changes better?

Evolutionary process

- Spiral model

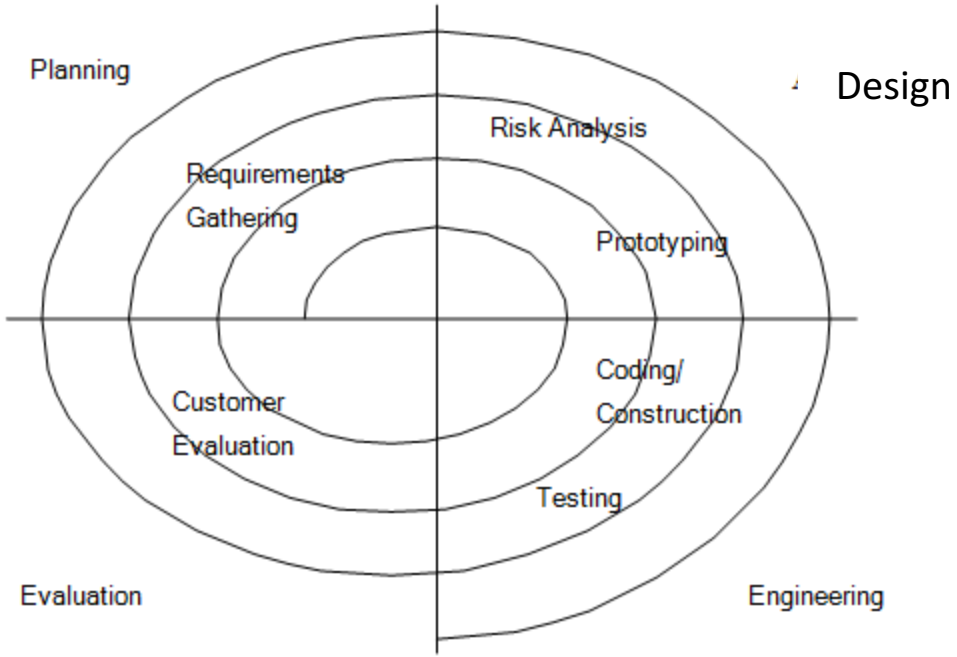


Fig. Spiral Model

Rational Unified Process

1990’

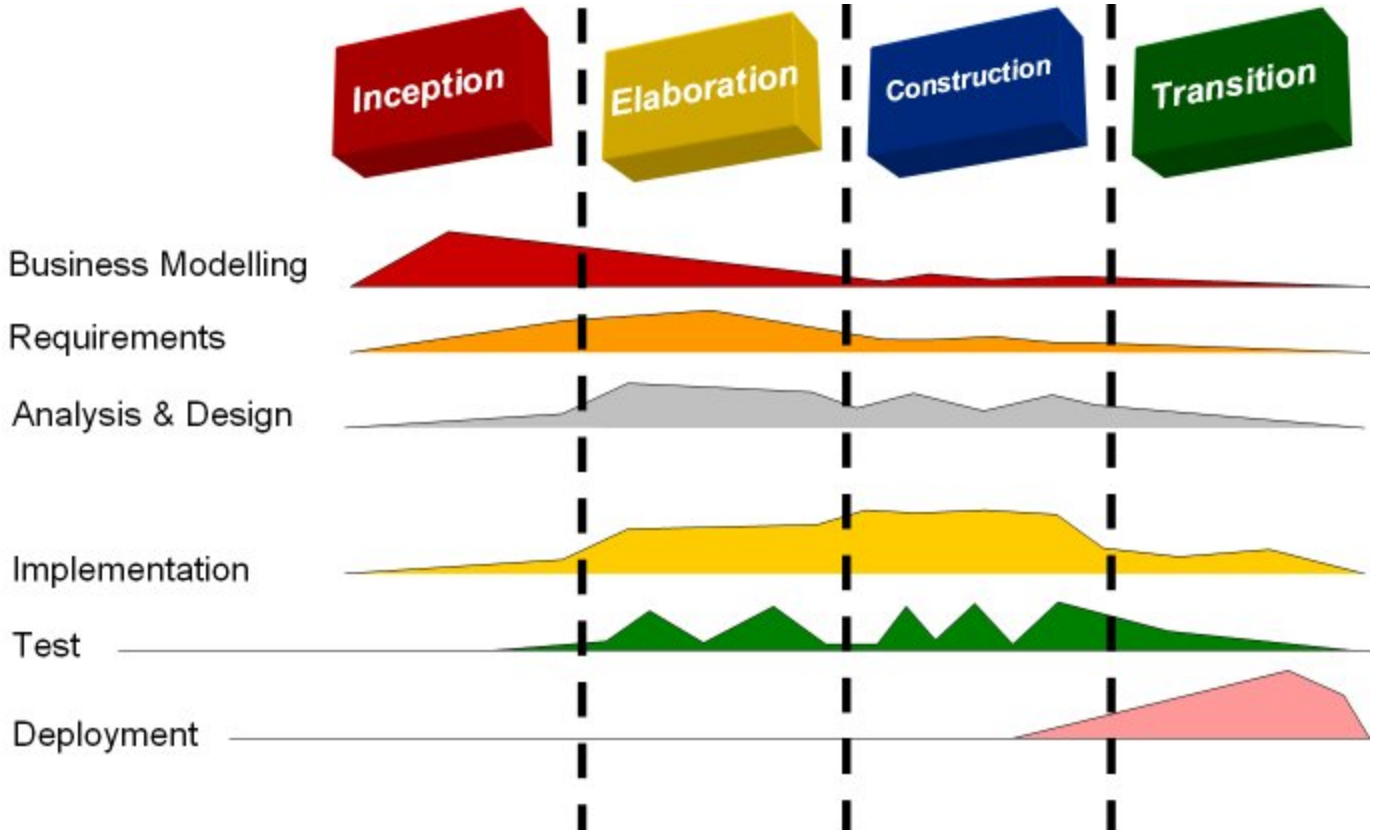
Rational Unified Process

- Basic idea: incremental + iterative
- Phases + workflows

Business modeling				
Req.				
Design				
Impl.				
Test				
Deployment				

Which workflow happens at which phase?

RUP


Inception
Elaboration
Construction
Transition
Business Modelling
Requirements
Analysis & Design
Implementation
Test
Deployment

RUP

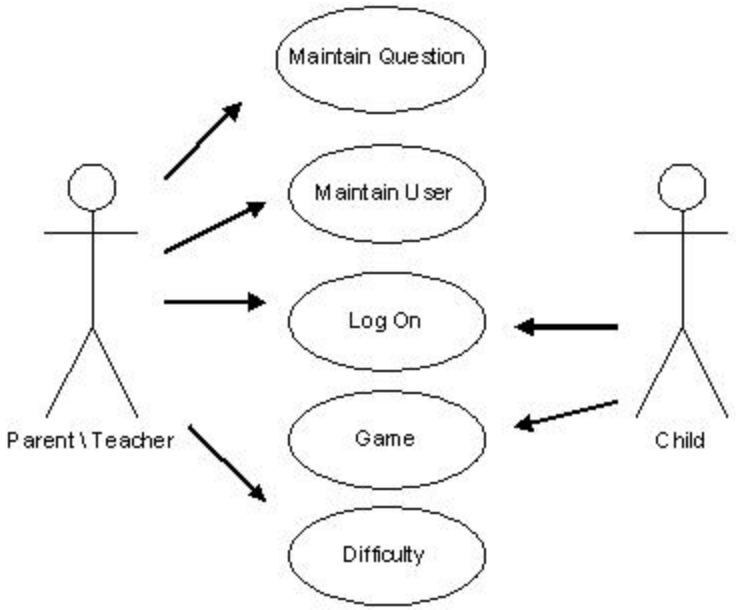
- What is the product of each workflow?
  - Unified Modeling Language
- Business modeling + requirement

  → Actor and use case diagram
- Analysis & design

  → class diagram, sequence diagram, state diagram
- Implementation
- Testing
- Deployment

  → deployment diagram

UML examples



Agile

2001

Background

- Planning planning planning
  - Airplane's control system needs 10 years to develop
- Problems
  - Too much document
  - Too late code delivery
  - Not easy to deal with changes
  - Too much bureaucracy
  - Hard to finalize design w/o implementation
  - Hard to estimate time before design & imp.
  - Hard to finish planning (prioritize) w/o estimating time

The Agile manifesto

- http://agilemanifesto.org/

12 key practices

- planning game
- small releases
- metaphor
- simple design
- testing (customer tests and tdd)
- Refactoring
- pair programming
- collective code ownership
- continuous integration
- 40 hour week
- onsite customer
- coding standards

The XP process

for each release/iteration (=2 weeks)

- review & planning
- design
- implementation

Planning

- ~~Requirement document~~
- User stories
  - What is it?
  - Customer provides …
  - Developers provide …

Planning

- ~~Requirement document~~
- User stories
  - What is it?
    - 3’’ X 5’’ card with text description
  - Customer provides: story, value
  - Developers provide: split a story to tasks, cost
  - Selection

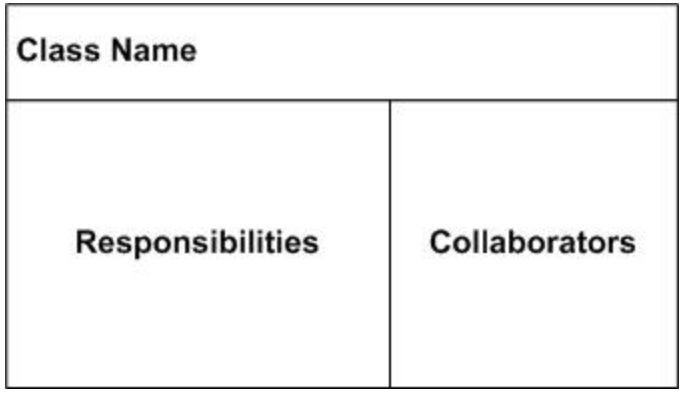
Example user stories

Design

- Principle – KIS (keep it simple)

- Output
  - CRC Card (Class-Respopnsibility-Collaboration)

Example (CRC Card)



Example (keep it simple)

Simplicity

```
int getSize (Vector v){
    return v.size();
}
```

Generality

```
int getSize (Container c) {
    Iterator i=c.iterator();
    int size =0;
    while(i.hasNext()){
        size++;
    }
    return size;
}
```

Design

- What is the problem of KIS?

- Solution

Design

- What is the problem of KIS?
  - Code difficult to maintain in the long term
- Solution
  - Code refactoring

refactoring

- What is refactoring?
  - **Code refactoring** is the process of restructuring existing computer code without changing its external behavior.

Implementation

- TDD (test-driven development)
  - Unit tests
  - Test suite
  - Regression testing & continuous integration
- Pair programming

Implementation

- TDD (test-driven development)
  - Unit tests (www.codehunt.com)
  - Test suite
    - The suite of many unit tests created and maintained over the time
  - Regression testing & continuous integration
    - Run the *whole* old/existing test suite at *every* code commit to make sure that new code does not violate old code assumption
    - Using test suite to replace documentation
- Pair programming

How to end an iteration?

12 key practices

- planning game
- small releases
- metaphor
- simple design
- testing (customer tests and tdd)
- Refactoring
- pair programming
- collective code ownership
- continuous integration
- 40 hour week
- onsite customer
- coding standards

Did Agile solve the problems?

Challenges for Agile

Summary

- Drawbacks of waterfall
- Good practices
  - Incremental, evolutionary
- RUP
  - Separating phases and work-flows
  - UML
- Agile, XP
  - …
  - tdd, small releases, …

Course Project

A few project example

- Proposal examples
  - …
- Repository examples
  - https://github.com/catherinemoresco/PDFProject
  - https://github.com/courageousillumination/deckr
  - https://github.com/dyxh/cs220
  - https://github.com/marlonliu/DivAssist

Course Project Grading

- Group performance
  - 75%
- Individual performance
  - Commit log
  - Self-evaluation + peer-evaluation
    - After milestone 3.b
    - After milestone 5