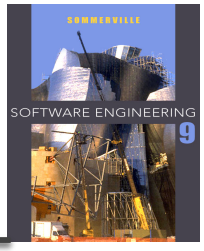
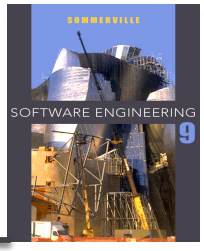


Software Engineering

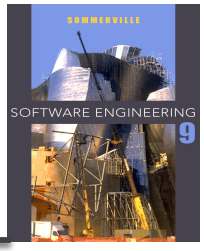
Mohsen Amini Salehi

Introduction



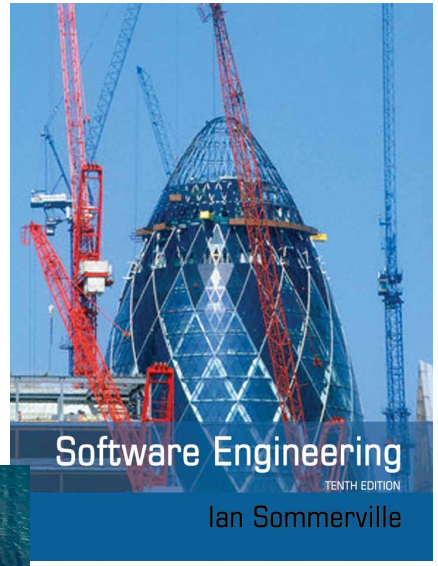
- ✧ Mohsen Amini Salehi
  - Born in 1980
  - Associate Professor
  - Prof at UL Lafayette from 2014 — 2023
  - Professor at UNT from 2023 — Now!
  - PhD from Melbourne University, Australia (2008-2012)
    - Director of the High Performance Cloud Computing (HPCC) lab.
    - Research interests:
      - Distributed and Cloud Computing

Introducing the Course: Text Books



✧ Course Title:

Software Engineering



✧ Reference Book:

Software Engineering (Edition 10)

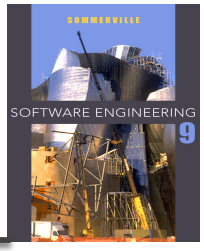
Author: Ian Sommerville

Engineering Software Products

Author: Ian Sommerville

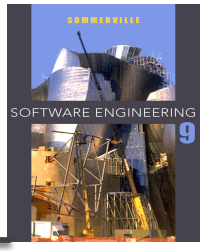
✧ Other online resources

Introducing the Course



- ✧ Recent research papers as another source for the class
- ✧ This is a practical course!
- ✧ Office Hours:
  - Office
    - Thursdays 3:30 – 5:30pm
    - Contact beforehand if the question will take more than 10min!
  - Email
    - 24/7
    - mohsen.aminisalehi@louisiana.edu

Introducing the Course

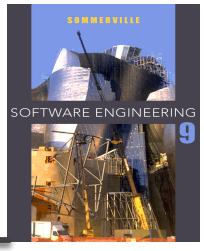


✧ I upload slides, samples, etc. through Canvas

✧ TAs:

TA Name	In-person office hours	Online office hours	Contact
Minseo Kim	Mon. 12:00 to 2:00pm	Wed 10:00am to 12:00 pm (link)	minseokim@my.unt.edu
Akiharu Esashi	Wed.12:00 to 2:00pm	Thur 12:00pm to 2:00pm (link)	akiharuesashi@my.unt.edu
Amina Firdouse	Fri. 2:00 to 4:00pm	Tues 2:00 to 4:00pm (link)	aminafirdousefirdouse@my.unt.edu

Course Plan

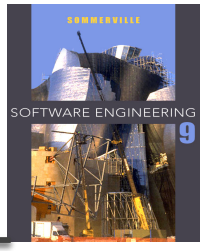


✧ Group-based Projects/Assignments

- **Groups of 5—8 students (start today!)**
- **60% of the final grade**
- Assignments can also be submitted within 24 hours after the due date unless otherwise advised (with 20% penalty!)
- Assignments will be announced through Canvas
- Just one week time to question any grade!

❖ Bonus point for extra class/assignment activities

Course Plan



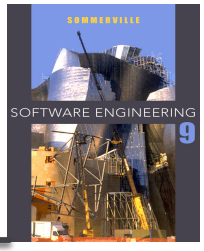
✧ Midterm and Final Exams

- In total 40% of the final grade
- ODS students should do the appropriate arrangements

✧ Grading standard

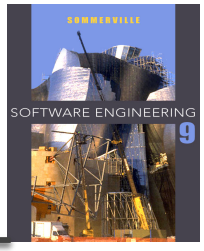
- A: 90-100
- B: 80-89
- C: 70-79
- D: 60-69
- F: 0-59

Academic Honesty



- Academic honesty is expected from all students in the class
- Any cheating and/or plagiarism will be treated based on the University academic honesty policy
- First violation zero in that activity; second violation F for the course and report to the University

Collaboration: A Tricky Issue!



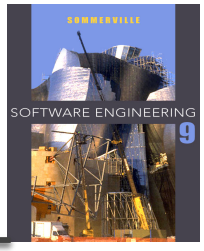
- It is encouraged! if it means: discussing the problem and solution with others

- It is **STRICTLY** prohibited if it means: inclusion of any code/text in the program/document that was not done by yourself!
  - This is in fact called cheating!

- Complete the integrity quiz on Canvas TODAY!

- Please include either of the following statements at the beginning of any assignment report :
  - *I certify this assignment is completely done by myself. However, I received help in the following senses. Explain the help here…*

Attendance

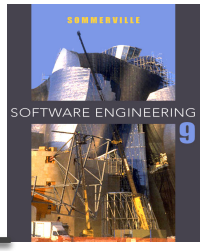


- Taking part in the classes is encouraged but not compulsory!

- However:
  - Students are responsible for all missed works, or any announcements in the class

- The absence reason could be anything (including) university sponsored events

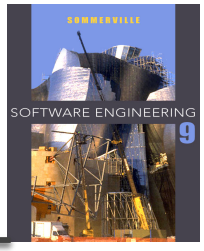
Course Goal



- My belief:
  - For the major part, Software Engineering organizes things you already know
  - Formalizes what we had been doing awkwardly!
- Understanding with the software lifecycle
- Obtaining the knowledge of software engineering for various scales and domains
- Learning responsibilities & ethics of a software engineer
- Ability to design a software
- *Don't forget: All companies are software companies!*

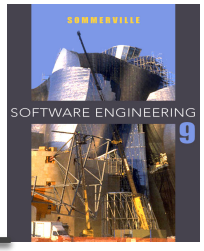
Course Goal

Activities During this Course



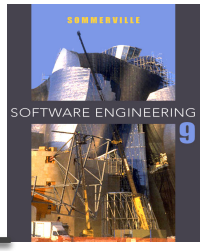
- ✧ Take part in the Academic Integrity quiz on Canvas
- ✧ Form your group
- ✧ Choose your project topic
- ✧ Perform requirement engineering and design
- ✧ Develop the project
- ✧ Test and deploy

Roadmap



- ✧ Evolution of Software Engineering Approaches
- ✧ Requirement Engineering
- ✧ Software Design
- ✧ Testing
- ✧ Software Evolution
- ✧ Cloud-based software deployment
- ✧ Dependability and Security
- ✧ Reusability and version management
- ✧ SOA
- ✧ Project Management

Software Engineering: A Retrospective View

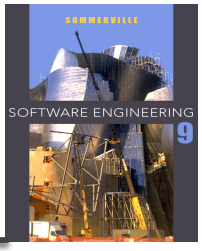


- Software "Engineering" emerged in 1960s
  - As part of Apollo project the essence of SE was felt
  - People were impressed by hardware development at that time
  - Software had to be designed for the specific hardware
- *Margaret Hamilton*, a lead engineer on Apollo program, responsible for on-board software development used the term around 1966.
- In **1968**,the first conference on software engineering was held, sponsored by NATO.

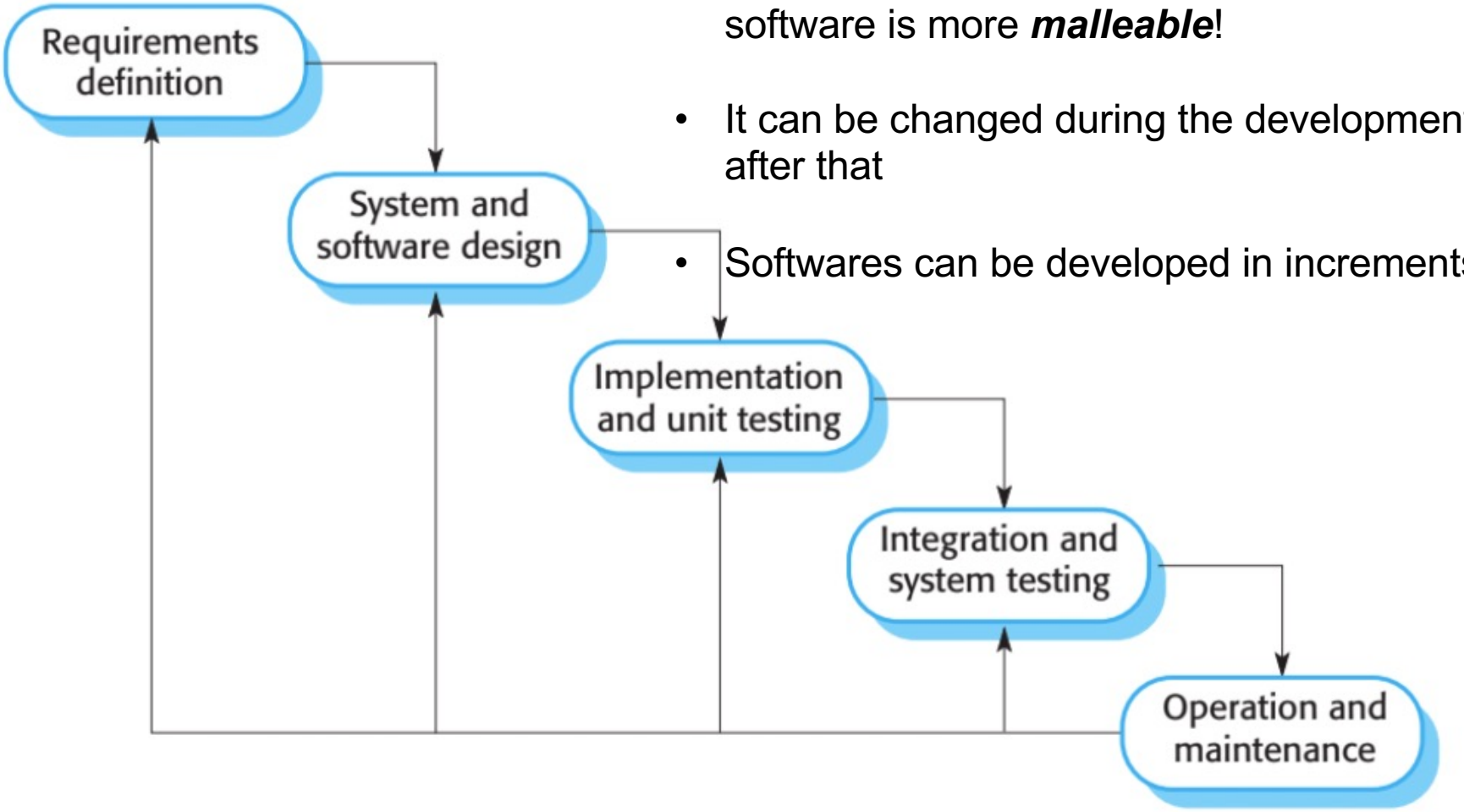
A discipline was born.

- The phased model of developing hardware was used as a basis for the so-called software life cycle model

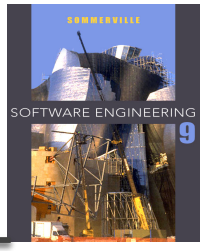
The software lifecycle: waterfall model!



- This approach is hardware-inspired, however, software is more ***malleable***!
- It can be changed during the development and after that
- Softwares can be developed in increments

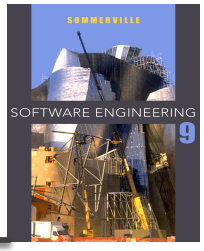


Software Development Ecosystem



- Waterfall does not prescribe any software tool
- However, over time many software tools emerged around this dominant software engineering paradigm
  - Structured programming in high-level languages (1960)
  - Graphical system modelling (1990)
  - Object-oriented development (1960 came back in 90's)
  - Programming environments (1970)
  - Parallel programming (practically in 80's)
  - Application program interfaces (API's) (1990)

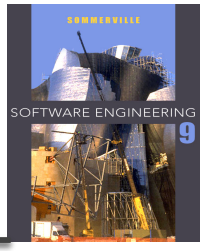
Conventional Software Engineering



- In conventional Software Engineering:
  - The customer has a problem
  - He/She specifies the characteristics of the software to be developed (software requirements)
  - Changes to the requirements are suggested by the customer and must be agreed with the developing company

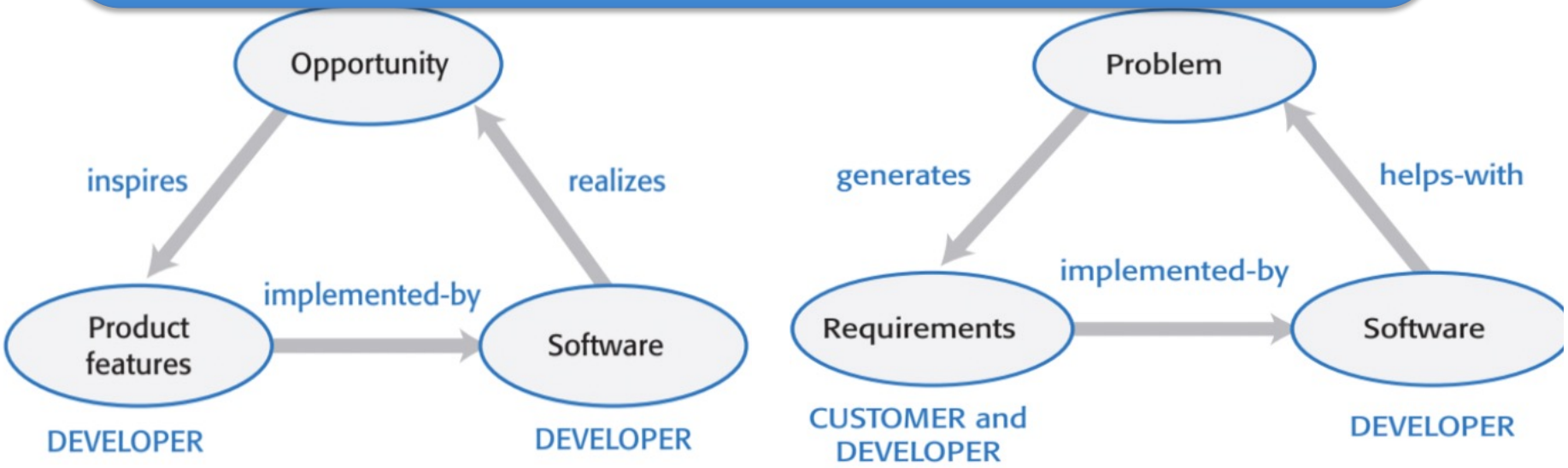
- Waterfall better fits software engineering, whereas agile fits to software products better

Software Product vs Software System Engineering

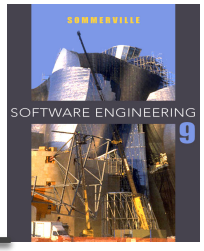


- SE designed to support the business activities of the purchaser of the software system

A paradigm shift in Software Industry:
Software product engineering is by far the biggest sector of the software market today!



Software Product vs Software System Engineering



- ✧ Both software products and systems have a long lifetime
- ✧ Example: banking systems and Ms. Excel (1985)
- ✧ Software systems still use the same code
  - Because owner should decide for the change
  - The non-tech owners generally do not understand when the software change is needed!
  - Sometimes they change the same system, and it becomes even more difficult to understand and maintain!
- ✧ Software products, however, rarely use the same code
  - They update more frequently