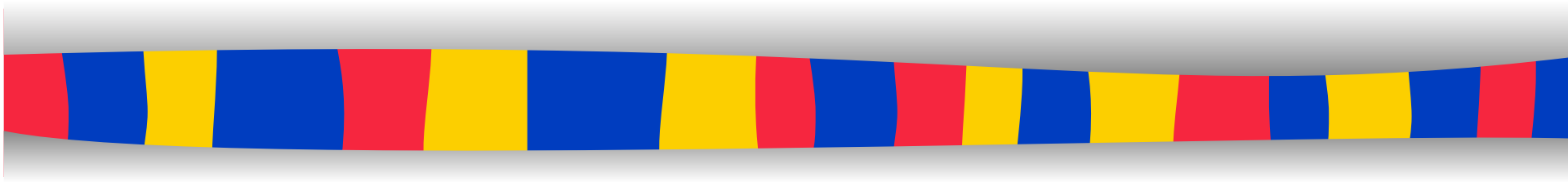
EECS 4314

Advanced Software Engineering



Topic 01:

Introduction and Admin

Zhen Ming (Jack) Jiang

Course Information

- Lecture Time
  - 13:00 to 14:30 pm every Mondays and Wednesdays in LSB 107
- Instructor:
  - Dr. Zhen Ming (Jack) Jiang, LAS 1012E, zmjiang@cse.yorku.ca
  - Office Hours: Mondays 2:30 - 3:30 pm or by appointment
- Course TA:
  - Minke Xiu
- Course Webpage:
  - https://wiki.eecs.yorku.ca/course_archive/2019-20/F/4314/

- Send emails from your York's email account, otherwise likely to be flagged as spam
- Put "**EECS4314**" in subject to go around spam filters

Calendar Description

- This course goes into more detail about some of the software engineering techniques and principles presented in earlier courses, as well as introduces advanced aspects of software engineering that are not addressed elsewhere:
  - Software process and its various models and standards (CMMI, ISO 9001).
  - Software architecture, i.e. the structure of data and program components that are required to build a software system. Examples include distributed and component-based architectures
  - Model Driven Engineering and the use of software description languages.
  - Software metrics, such as metrics for software quality, software design metrics, as well as testing and maintenance metrics.
  - Project management concepts on coordinating people and products.
  - Cost estimation and project scheduling for large software systems.
  - Risk management and mitigation.
  - Software configuration management (software evolution, change management, version and release management).
  - Emerging technologies, such as security engineering, service-oriented software engineering, and aspect-oriented software development.

What is Course is about

- This course goes into more detail about some of the software engineering techniques and principles presented in earlier courses, as well as introduces advanced aspects of software engineering that are not addressed elsewhere:
  - Software architecture
  - Software project management
  - Software cost estimation
  - Software metrics *(if time permits)*
  - Software performance *(if time permits)*

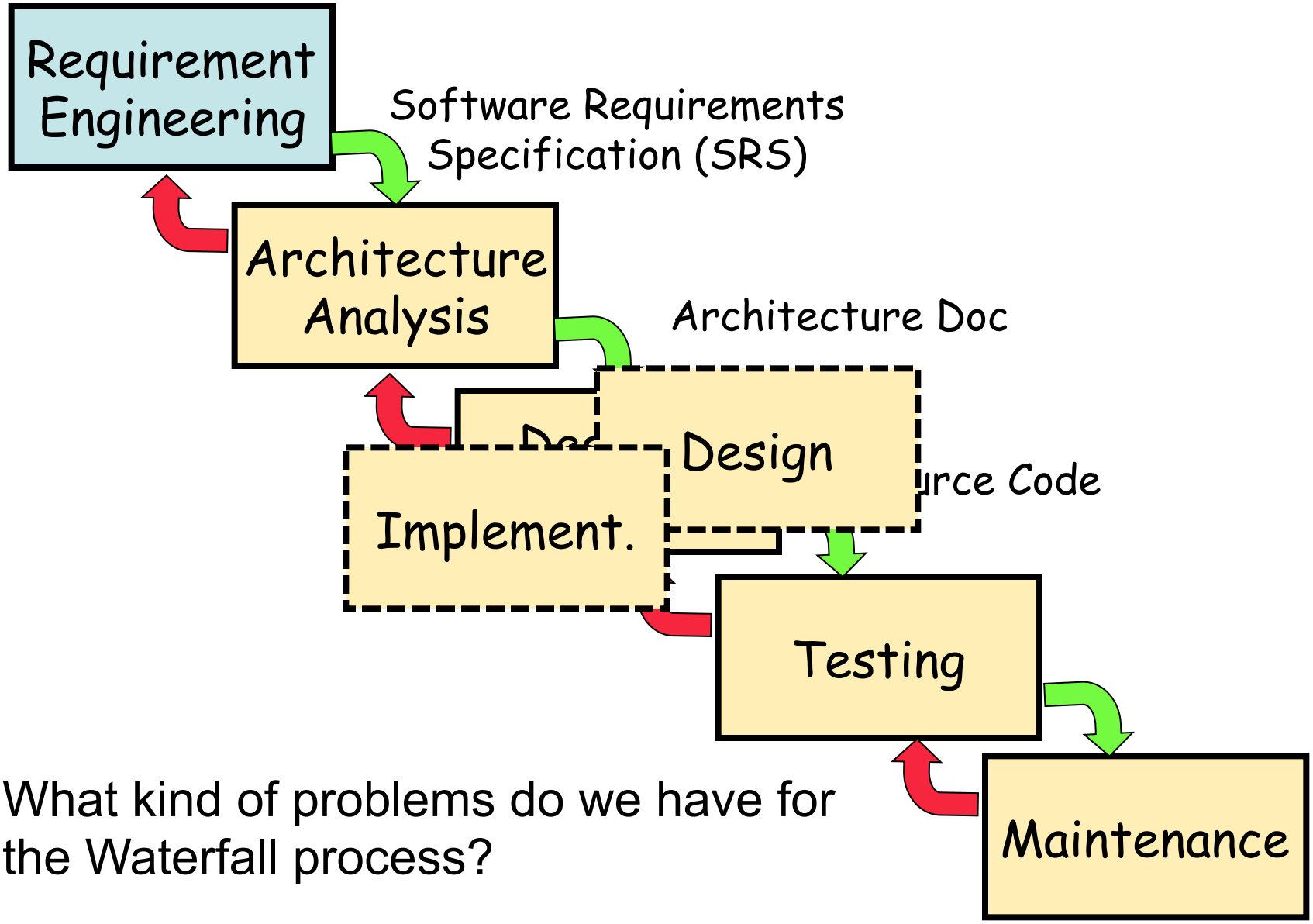
Learning Objectives

- Derive models of software systems and express them in a language such as UML
- Understand the differences between different types of software architecture
- Apply metrics that estimate the quality, maintainability, and test adequacy of a software system
- Derive cost estimation tables delineating the tasks to be performed, and the cost, effort, and time involved for each task
- Identify risks associated with a given software project, and develop plans to mitigate and manage these risks
- Manage software projects by identifying the sequence of tasks that will enable the project to complete in time, assigning responsibility for each task, and adapting the schedule as various risks become reality

On Software Engineering

- Software engineering is a pure intellectual activity
  - Output is documentation
  - Program text is a form of electronic documentation
- Difference with other engineering disciplines
  - Software has no physical characteristic
    - no mass, no heat produced
  - Software implements highly complex functions in a flexible way, making it an essential part of other systems

Waterfall Development Process



What kind of problems do we have for the Waterfall process?

Course Scope

- Exposes you to the challenges in developing and maintaining large and ultra-large software systems
- Learn various concepts related to large scale software development
  - Architectural views
  - Architecture evaluation methods
  - Social architecture (Conway's law)
  - Effort estimation techniques
  - Etc.
- Study the architecture of a large software system (***IntelliJ IDEA***)

Feedback from the industries

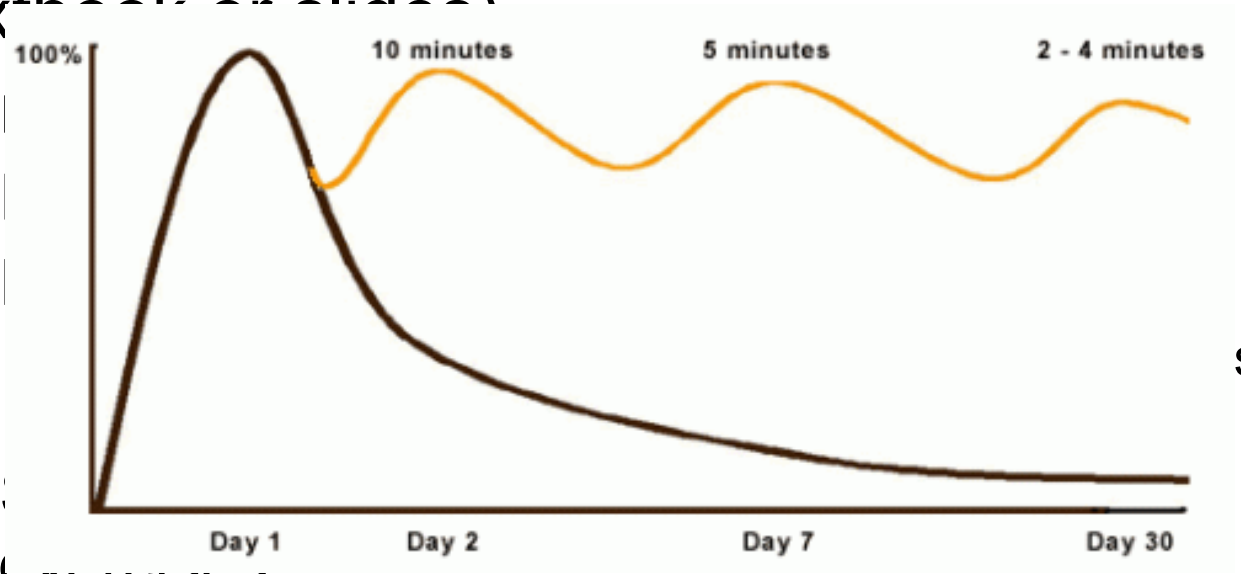
- Ask software companies for advice
  - Amazon, eBay, Google, Microsoft, Salesforce, VMware
- Students can write code, but lack basic software skills, especially:
  1. Dealing with legacy code (unanimous)
  2. Working as team with non-technical customer
  3. Automated testing

Course Expectations

- Read assigned readings
- Attend lectures and participate in discussions
- Bring your ideas and concerns to class
- Work effectively in a group setting (group members will evaluate each other)
- Learn how to use the tools and understand your project *very well*
- Hand in your deliverables on time

Study Strategy

- Don’t fall behind – Learning is work and repetition
- Attend classes (even though some materials are in tex
- Fo
  -
  -
  -



signments,

the

exercises

    - If you do not reflect on and use the material the same day you forget 50% within 24 hours and 80% within 48 hours

Evaluation

Assignments (4)	60%
Final Exam	40%

Course Assignments

- Form groups and create group webpages
- Describe the conceptual architecture
- Recover the concrete architecture and compare to the conceptual architecture
- Investigate system analysis techniques
- Look into ways to enhance the architecture

Team Website

- You need to update a group website throughout the term
- Website should be up by **Sept. 18, 2019**
  - Worth 0% of your mark
  - If not kept up-to-date you lose maximum 2%

Peer Reviews

- All members should receive same marks for project. However, to account for that individual effort, we have peer reviews
  - You can assign each member (including yourself) a grade
  - You have 5 * N + 1 marks, where N is size of group
- Peer reviews are sent 24 hours ***after*** each large deliverable:
  - For example, if Part1 is due on 9 am Oct. 21, 2016, the peer review will be due 9 am Oct. 22, 2016.
- **Reviews are submitted via email with subject: "Peer Review for Group ##"**

Lateness Policy for All Course Deliverables

- For all deliverables:
  - Hand in hard copy at the beginning of class or earlier to the instructor
  - Submit online via PRISM

NO LATE DELIVERABLES!!

Academic Integrity and Cheating

- Cheating, plagiarism and other forms of academic fraud are taken very seriously by the University, the Faculty, and the teaching staff.
- Examples:
  - Submitting the work of another person as your original work
  - Incorporating others work in your work and not referencing it
  - It is permitted and encouraged to discuss assignments with your peers on the whiteboard but **NOT** permitted to copy their solutions as they talk to you. Both parties will be penalized

Course Text

- There are two optional textbooks for the course:
  - Ian Sommerville. Software Engineering (10th Edition). Software Engineering (10th Edition).
  - Martin Fowler, Kent Beck, John Brant, William Opdyke, Don Roberts. Refactoring: Improving the Design of Existing Code.
- Lecture slides, papers, online books
- Additional online readings assigned for case studies
- Exams will cover assigned readings and topics covered in class

Working in Groups and Choosing a Group

- Group Size : >= 5
- Understand the work habits and goals of your group members:
  - Night person
  - Start early
  - Laid back
  - Best project ever
  - Morning person
  - Start at last minute
  - Perfectionist
  - Reasonable mark
- Identify members with good communication skills

Asking Questions

- Ask in class
- Ask me (email, office hours)
- Discuss with your classmates or group members
- Ask on the discussion forum

The Software Pyramid

Software programming is the iconic job of the Information Age, but not all programmers are created equal. Here's the breakdown of software jobs and their prospects:

1 ARCHITECTS A few thousand tech visionaries sketch out entire systems to handle complex jobs. Adam Bosworth, for example, is the chief architect at BEA Systems.

PAY $150,000 to $250,000.

OUTLOOK Outsourcing is a nonissue.

2 RESEARCHERS They're key to innovation, which is crucial for the U.S. But there are only about 25,000 in the country, many in academia, where tenure trumps pay.

PAY $50,000 in academia to $195,000 in private sector.

OUTLOOK Prospects should brighten somewhat with the economy, but these jobs can move offshore, too.

3 CONSULTANTS Business-savvy consultants advise corporations about their technology needs, help them install new software, and create new applications from scratch.

PAY $72,000 to $200,000.

OUTLOOK Still bright for Americans. U.S. customers want face time with consultants.

4 PROJECT MANAGERS Crucial cogs in global software factories. They coordinate the work of teams in different countries and time zones and provide dependable products on schedule.

PAY $96,000 to $130,000.

OUTLOOK Good managers can write their own tickets. Pay has jumped 14.3% in the past two years.

5 BUSINESS ANALYSTS Go-betweens. About 100,000 analysts figure out what a business needs and turn it into a spec sheet for programmers. It's a key role now since the company and its programmers are often apart.

PAY $52,000 to $90,000.

OUTLOOK A relatively safe haven for programmers—if they have communications skills and a grip on business.

6 BASIC PROGRAMMERS The foot soldiers in the information economy, they write the code for applications and update and test them. Numbering about 1 million, they are one-third of all U.S. software engineers and programmers.

PAY Has tumbled 15% since 2002. Now $52,000 to $81,000.

OUTLOOK Watch out. Many of these jobs can be done anywhere. Forrester predicts 18% of them will be offshore within six years.

Data: Forrester Research, Foote Partners, Kennedy Information Inc., *BusinessWeek*

BusinessWeek March, 2004

Tell Me a Little About You

- Background
  - SE, ECE, CS, CE, others?
- Any industrial experience (e.g., co-op or internship)?
- Schedule
  - Any classes after this?
- Plan to graduate in next summer?
  - Grad schools or jobs?