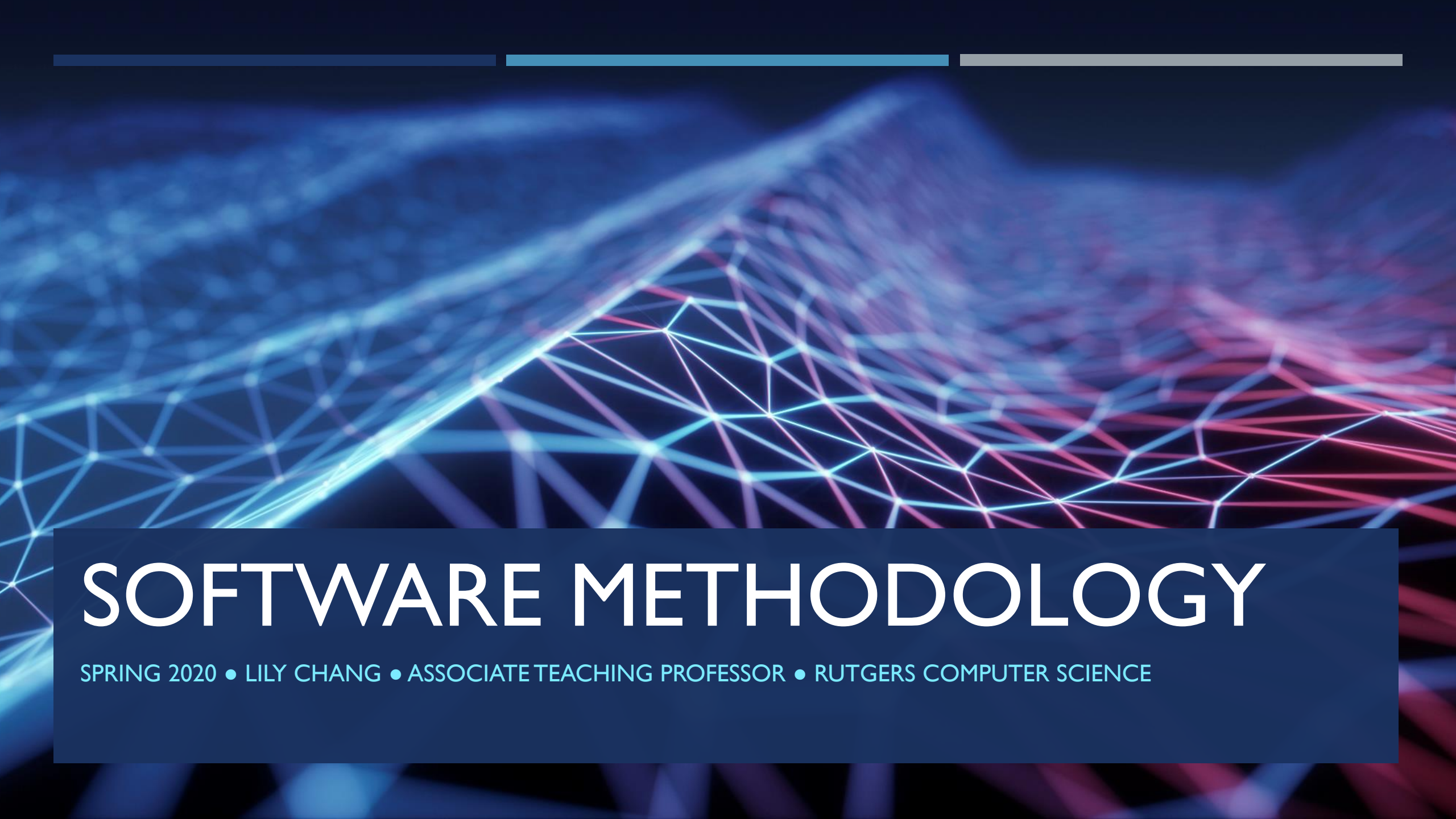


SOFTWARE METHODOLOGY

SPRING 2020 • LILY CHANG • ASSOCIATE TEACHING PROFESSOR • RUTGERS COMPUTER SCIENCE

OVERVIEW OF SOFTWARE DEVELOPMENT

WHAT IS SOFTWARE?

- Computer programs

AND

- Associated documentations
  - Describe the structure of the software
  - User documentation

AND

- Configuration data
  - To make the programs operate correctly

ESSENCE OF SOFTWARE

- Abstract and intangible
- No physical limitations
  - Good or bad?
- Complex
- Changeable

SOFTWARE IS COMPLEX

- Involve a group of people from different disciplines / application domains with different perspectives
- Comprise many interacting components, oftentimes distributed
  - Service-oriented cloud computing
  - Internet of Things (IoT)
  - Cyber Physical Systems (CPS)
    - Human in the loop

What do you see?

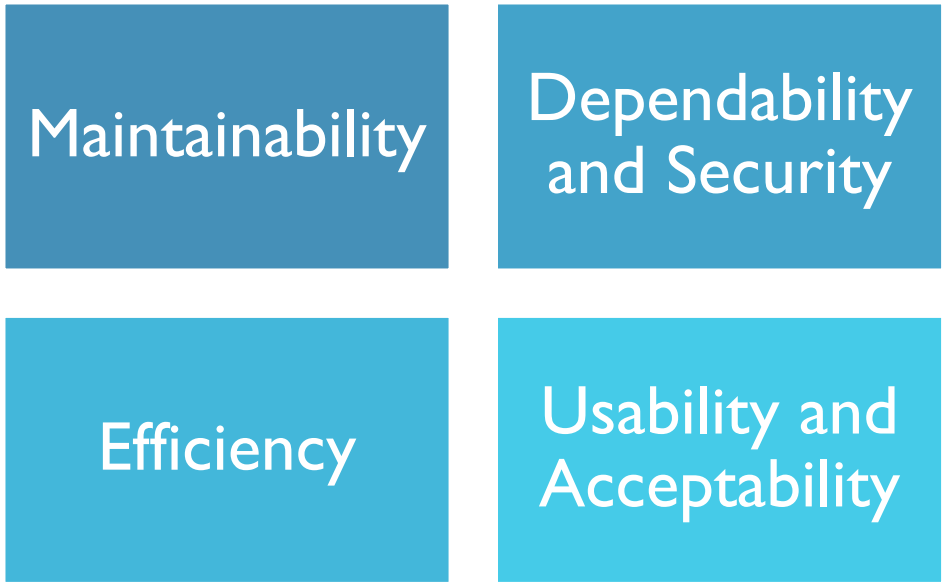
Ambiguity

SOFTWARE IS CHANGEABLE

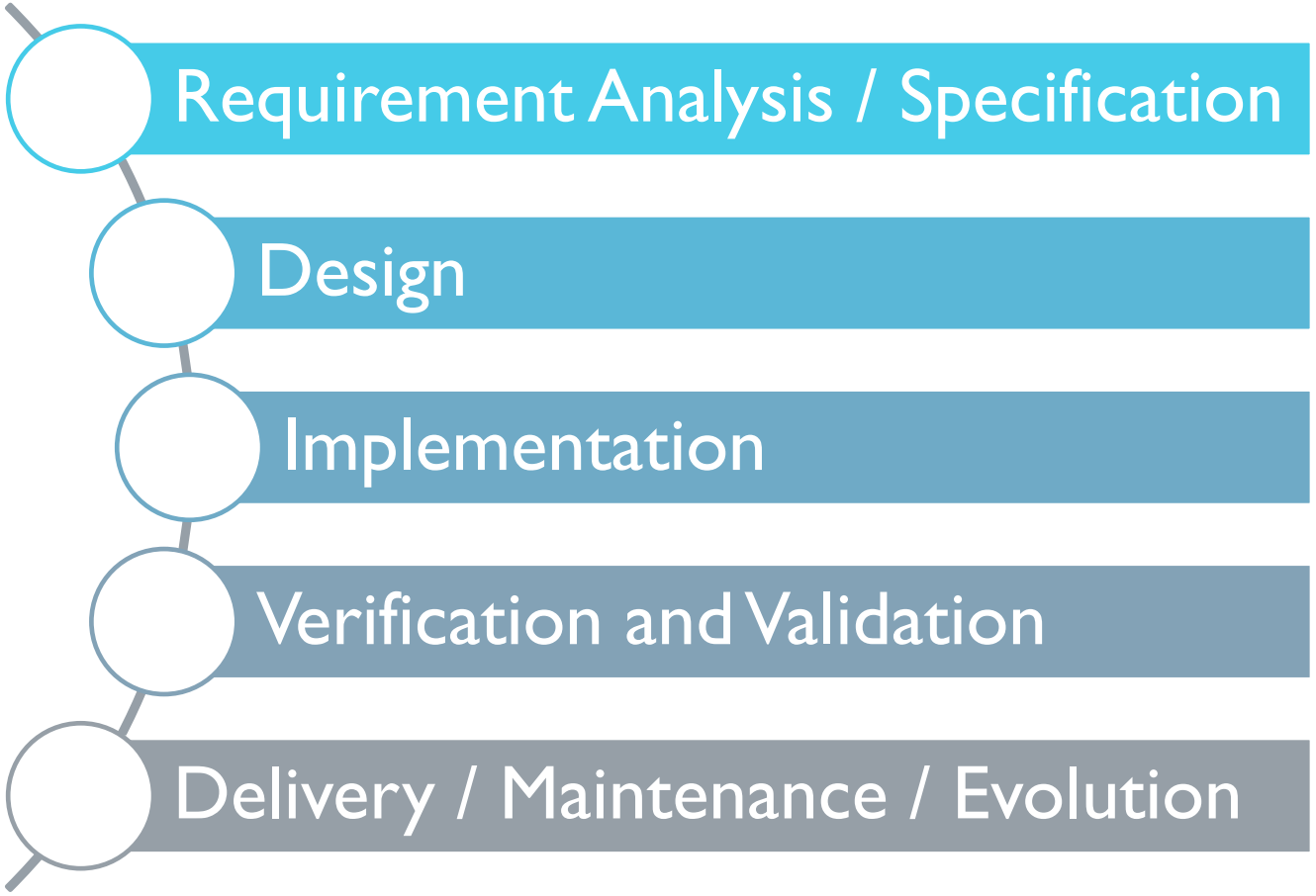
- Software must evolve to remain useful
  - Discovered errors must be fixed
  - Software is cheap and easy to change as opposed to hardware components
  - The time between technological changes is often shorter than the duration of the project
  - Requirements change!!!

SOFTWARE METHODOLOGY

- Methods for developing **quality** software
- Good practices for software development



ESSENTIAL ACTIVITIES FOR SOFTWARE DEVELOPMENT



SOFTWARE PROCESS MODELS

- A Software Process Model is an abstract representation of a software production process
- Generic process models (paradigms)
  - Waterfall model

  - Incremental or evolutionary
    - Spiral Model
    - Agile

GOOD PRACTICES FOR SOFTWARE DEVELOPMENT

- Clean code
  - Clean code can be read and enhanced by a developer other than its original author
  - It has unit and acceptance tests.
  - It has meaningful names.
  - It provides one way rather than many ways for doing one thing.
  - It has minimal dependencies, which are explicitly defined, and provides a clear and minimal API.
  - The logic should be straightforward to make it hard for bugs to hide, the dependencies minimal to ease maintenance

GOOD PRACTICES FOR SOFTWARE DEVELOPMENT

- Agile - https://www.agilealliance.org/agile101/subway-map-to-agile-practices/
  - Adapt software change better
  - frequent delivery (continuous integration and delivery)
  - Involve customers in the development process
  - Popular Agile methods – pair programming (eXtreme Programming), Scrum

PAIR PROGRAMMING

- Key practice of eXtreme Programming
- Consists of two programmers sharing a single workstation (one screen, keyboard and mouse among the pair). The programmer at the keyboard is usually called the "driver", the other, also actively involved in the programming task but focusing more on overall direction is the "navigator"; it is expected that the programmers swap roles every few minutes or so.

PAIR PROGRAMMING – EXPECTED BENEFITS

- Increased code quality: "programming out loud" leads to clearer articulation of the complexities and hidden details in coding tasks, reducing the risk of error or going down blind alleys
- Better diffusion of knowledge among the team; in particular, when a developer unfamiliar with a component is pairing with one who knows it much better
- Better transfer of skills, as junior developers pick up micro-techniques or broader skills from more experienced team members
- Large reduction in coordination efforts, since there are N/2 pairs to coordinate instead of N individual developers
- Improved resiliency of a pair to interruptions, compared to an individual developer: when one member of the pair must attend to an external prompt, the other can remains focused on the task and can assist in regaining focus afterwards

PAIR PROGRAMMING – COMMON PITFALLS

- Both programmers must be actively engaging with the task throughout a paired session, otherwise no benefit can be expected
- A simplistic but often raised objection is that pairing "doubles costs"; this is the worst-case outcome of poorly applied pairing
- At least the driver, and possibly both programmers, are expected to keep up a running commentary; pair programming is also "programming out loud" – if the driver is silent, the navigator should intervene
- Pair programming cannot be fruitfully forced upon people, especially if relationship issues, including the most mundane (such as personal hygiene), are getting in the way; solve these first!

PROGRAMMING GROUND RULES

- Enforce the coding standard for this class
  - Comments
  - Meaningful names
  - Formatting
  - No MAGIC numbers!
  - Modularity
    - Do only ONE THING in a function/method
    - A long method is an indication that the method is doing too much!

OBJECT-ORIENTED SOFTWARE DEVELOPMENT

- Object-oriented analysis, design and programming are related but distinct.
- OOA is concerned with developing an object model of the **application domain. (What)**
- OOD is concerned with developing an object-oriented system model to implement requirements (**solutions domain**) (**How**)
- OOP is concerned with **realizing** an OOD using an OO programming language such as Java or C++.

CHARACTERISTICS OF OO DESIGN

- Objects are abstractions of real-world entities such as people, things and events
- An object encapsulates data and processes
- System functionality is expressed in terms of object services (methods)
- Shared data areas are eliminated.
  - Objects communicate by message passing (arguments and parameters)
- Objects may be distributed and may execute sequentially or in parallel.

ADVANTAGES OF OO DESIGN

- Reusability by inheritance
  - A superclass is a generalization of its subclasses
  - A subclass inherits the attributes and methods of its parent class
- Extendibility by creating new subclasses with new behaviors
- Data hidden by encapsulation
- For some systems, there may be an obvious mapping from real world entities to system objects
- Maintainability - objects may be understood as stand-alone entities.

THIS COURSE

- Specification or requirements will be given by the instructor
- A little bit OO Design with Unified Modeling Language (UML)
  - Class diagram (visualize the software structure)
- Implementation
  - OOP using Java
  - IDE – Eclipse/Android Studio OR IntelliJ
- Testing
  - Test design (test specifications and test cases)
  - Testbed main (implement the test cases)
  - JUnit test (a framework for writing test code)
- Pair Programming

THANK YOU