EECS 2311

Software Development Project

Click to edit Master text styles
Second level
Third level
Fifth level

January 11, 2017

YORK UNIVERSITÉ UNIVERSITY

Software Engineering

- Software Engineering is the science and art of building significant software systems that are:
  1. on time
  2. on budget
  3. with acceptable performance
  4. with correct operation

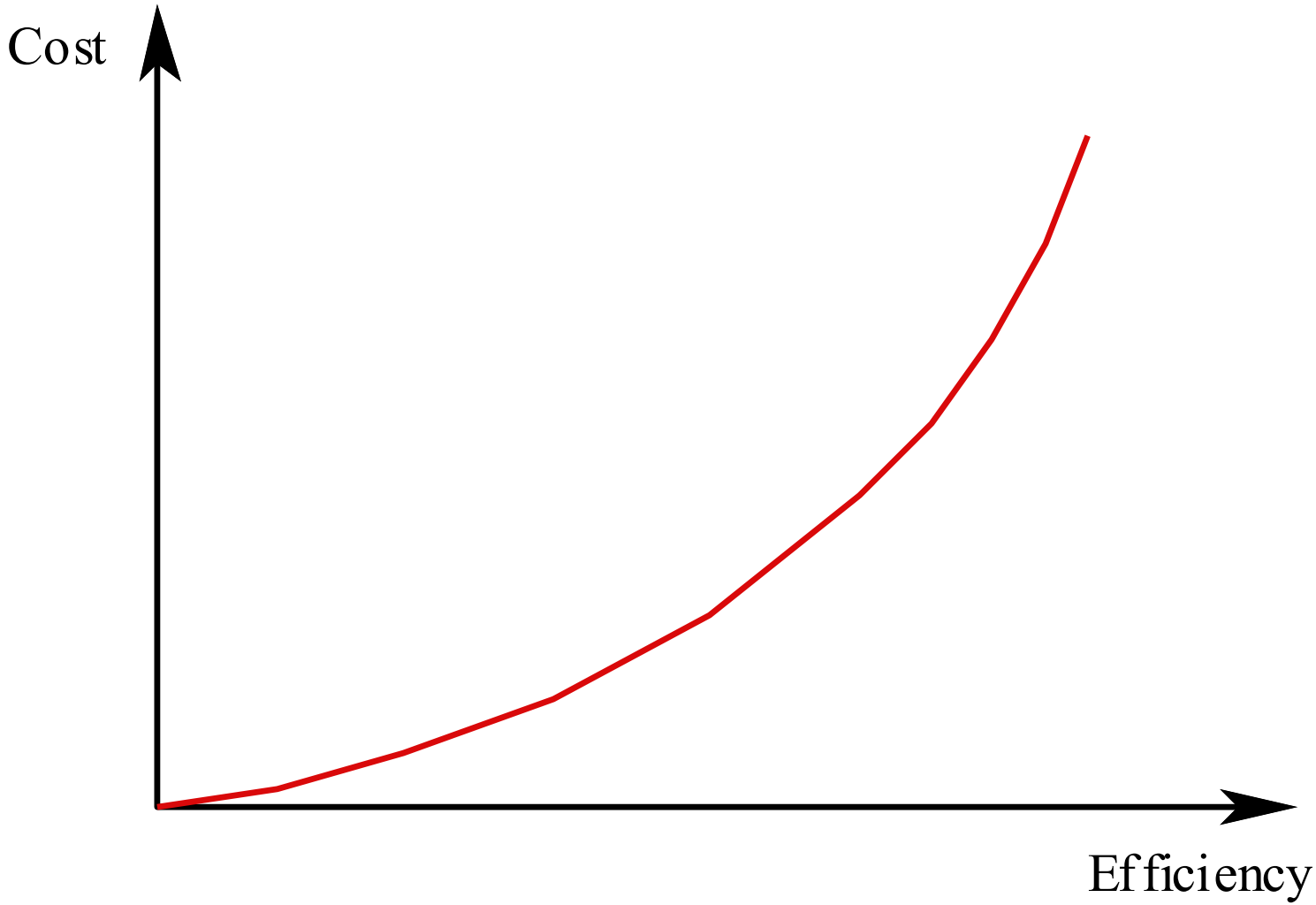
Software Product Attributes

- Maintainability
- Dependability
- Efficiency
- Usability

Importance of Product Characteristics

- The relative importance of these characteristics depends on the product and the environment in which it is to be used.
- In some cases, some attributes may dominate
  - In safety-critical real-time systems, key attributes may be dependability and efficiency.
- Costs tend to rise exponentially if very high levels of any one attribute are required.

Efficiency Costs



The Software Process

- Structured set of activities required to develop a software system
  - Specification
  - Design
  - Validation
  - Evolution
- Activities vary depending on the organization and the type of system being developed.

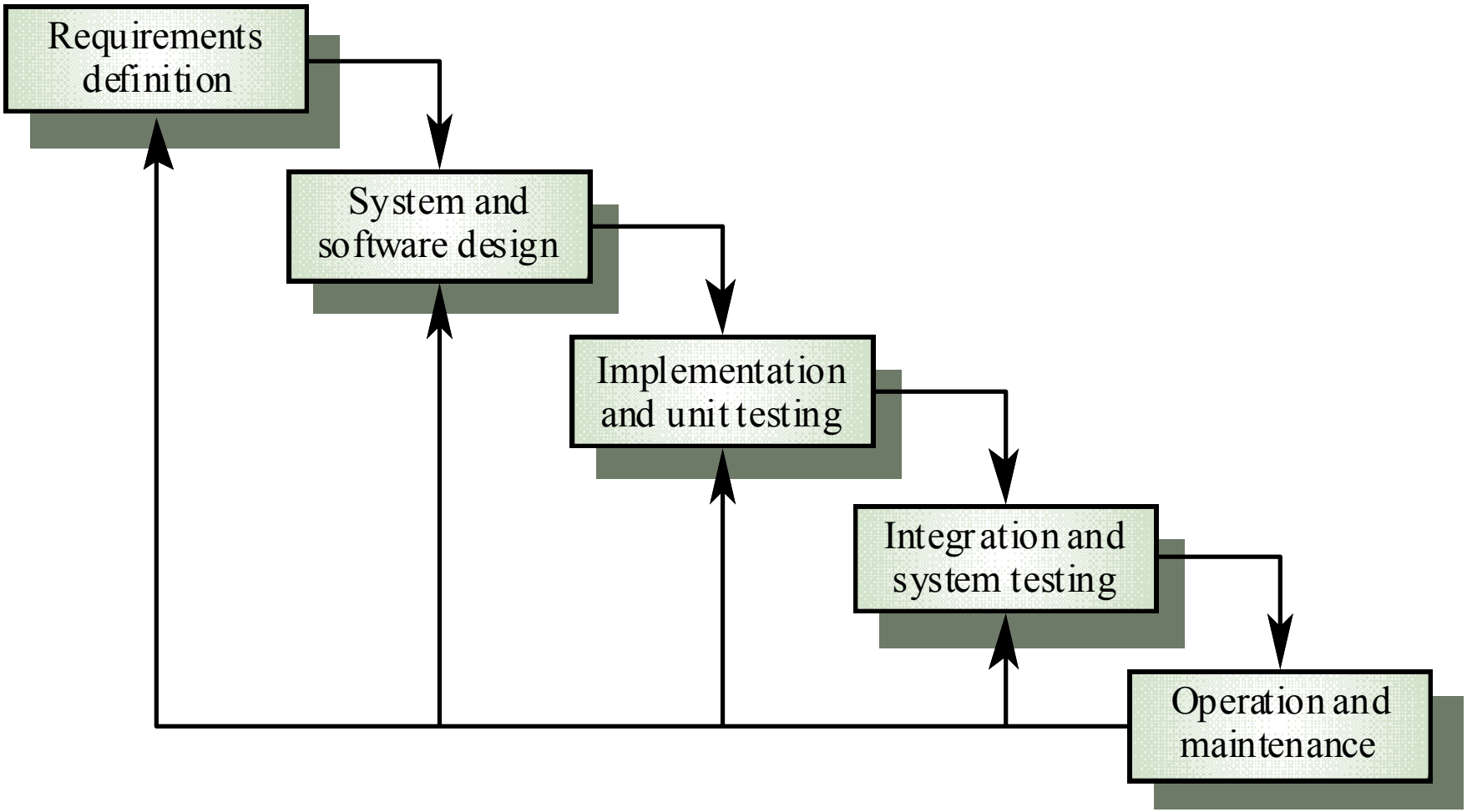
Engineering Process Model

- **Specify**: Set out the requirements and constraints on the system.
- **Design**: Produce a model of the system.
- **Manufacture**: Build the system.
- **Test**: Check the system meets the required specifications.
- **Install**: Deliver the system to the customer and ensure it is operational.
- **Maintain**: Repair faults in the system as they are discovered.

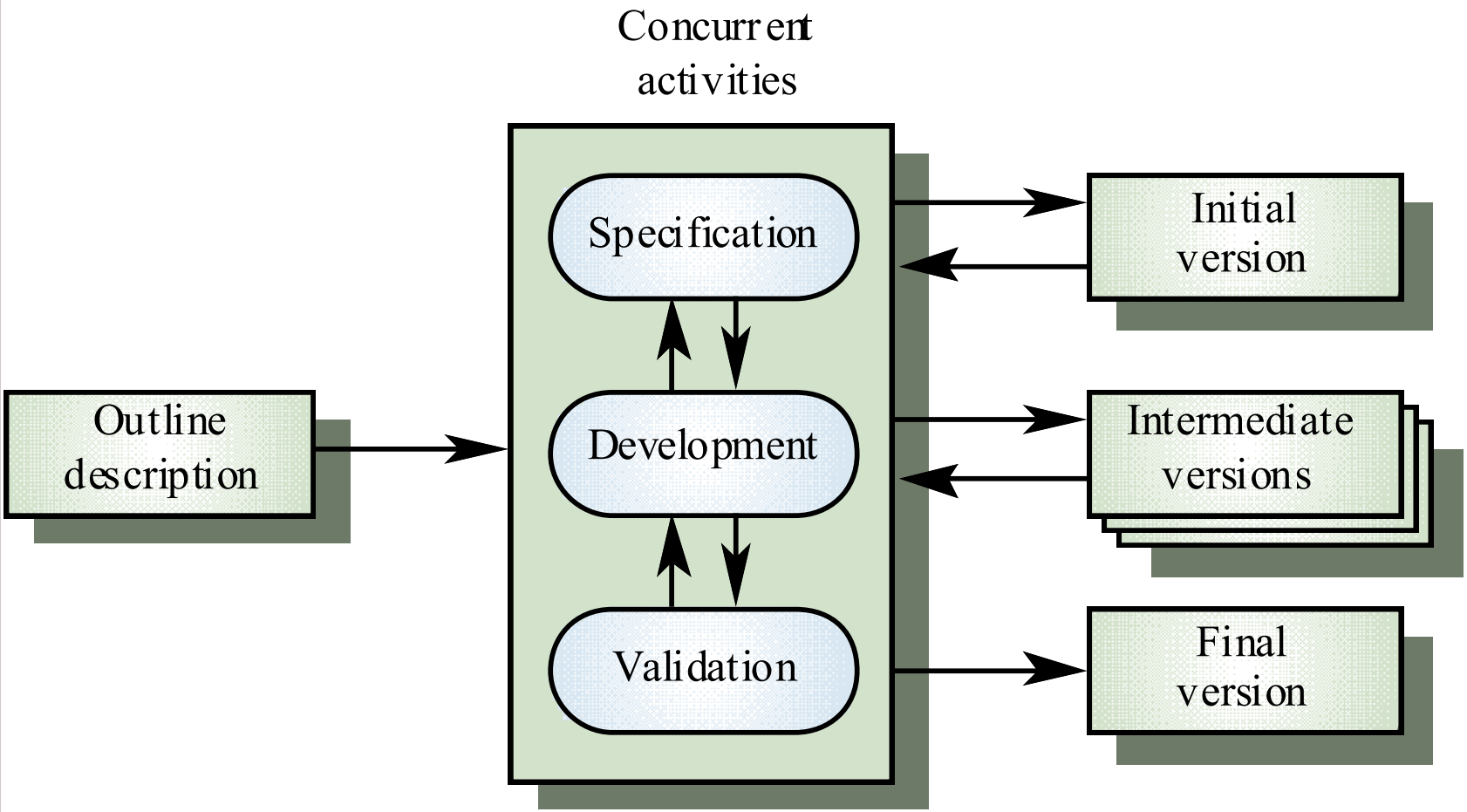
Software Engineering is Different

- Normally, specifications are incomplete.
- Very blurred distinction between specification, design and manufacture.
- No physical realization of the system for testing.
- Software does not wear out - maintenance does not mean component replacement.

Waterfall Process Model



Evolutionary Process Model



Agile methods

- Dissatisfaction with the overheads involved in design methods led to the creation of agile methods. These methods:
  - Focus on the code rather than the design;
  - Are based on an iterative approach to software development;
  - Are intended to deliver working software quickly and evolve this quickly to meet changing requirements.
- Agile methods are probably best suited to small/ medium-sized business systems or PC products.

Principles of agile methods

- Customer involvement
- Small releases
- Embrace change
- Test-first design
- Refactoring
- Continuous integration

Our project

- Develop software for a hardware device used to help kids read Braille
- The hardware device will have a number of Braille cells, as well as a number of physical buttons
- The system presents Braille characters/words to the user who then responds by pressing buttons
- The hardware will be under development this term as well, so our first task will be to develop a...

Simulator

- A piece of software that simulates the behaviour of the hardware device
- Has a user interface similar to that of the device
- Is fully tested to behave as the hardware device
- Presents an API that the rest of the system will use to communicate with it
- The simulator along with its test cases is due on **Feb 7**

Player

- The interactive scenarios that the system will go through are played by a Java app that runs on a Raspberry Pi inside the hardware device. We will call this app the Player
- The Player reads prepared scenarios from a file and executes them accordingly
- Determining the format of that file is part of the project
- The Player along with testing and documentation is due on **Mar 7**

Authoring app

- The interactive scenarios that the system will go through will be created in a desktop app called the Authoring app
- The Authoring app will provide facilities to create the flow of the scenario, record audio (or generate audio via text-to-speech), and save the scenario in the story format
- The Authoring app along with testing and documentation is due on **Apr 5**

To get started

- Read up on Braille online to become familiar with it
- Review Swing and event-driven programming
- Setup your team's github repository and make sure every team member can push / pull

Intentionally vague requirements

- In a real software development project, requirements are vague and ever-changing
- The exact requirements will be refined iteratively by meeting with the “customer” on a weekly basis

Teams

- Teams are assigned randomly by the “manager”
- As enrollment in the course changes in the first few weeks, the “manager” will rearrange the teams
- Same as a real software project!

Team 1

- **Banh, Kevin**
- **Israr, Zaeem**
- **Khademi, Amir-Hossein**

Team 2

- **Khatri, Dilshad**
- **Noel, Drew**
- **Tung, Jonathan**

Team 3

- **Bhardwaj, Siddharth**
- **Dao, Eric**
- **Lee, Dong**

Team 4

- **Li, Derek**
- **Mohamed, Yassin**
- **Solovey, Artem**

Team 5

- **Ejaz, Rabia**
- **Torres Fleites, Danilo**
- **Usman, Syed**

Team 6

- **Harrymanoharan, Jessanth**
- **Manjra, Ibrahim**
- **Nnorom, Elijah**

Workload

- This course requires 8-10 hours per week per student
- Have to start working immediately
- In the second part of each lecture, each team will present their progress to the instructor and receive feedback
  - “Customer” on site!

Evaluation

- 5% - Lab tasks completed
- 20% - Simulator (due Feb 7)
- 35% - Player (due Mar 7)
- 40% - Authoring app (due Apr 5)
- Each team submission will receive a grade based on its merit. Individual grades may be less if full participation has not been demonstrated.

Design competition

- Independent of this course, a design competition to produce software for a Braille learning hardware device has been established by Classy Cyborgs
- Teams are free to choose whether they want to enter their submission to the design competition. Entering the competition or not will not affect your mark
- There are monetary prizes for the winner of the competition