CSCI 2132 – Software Development

Software Development Life Cycle, Testing and Debugging

February 16, 2017

Dalhousie University

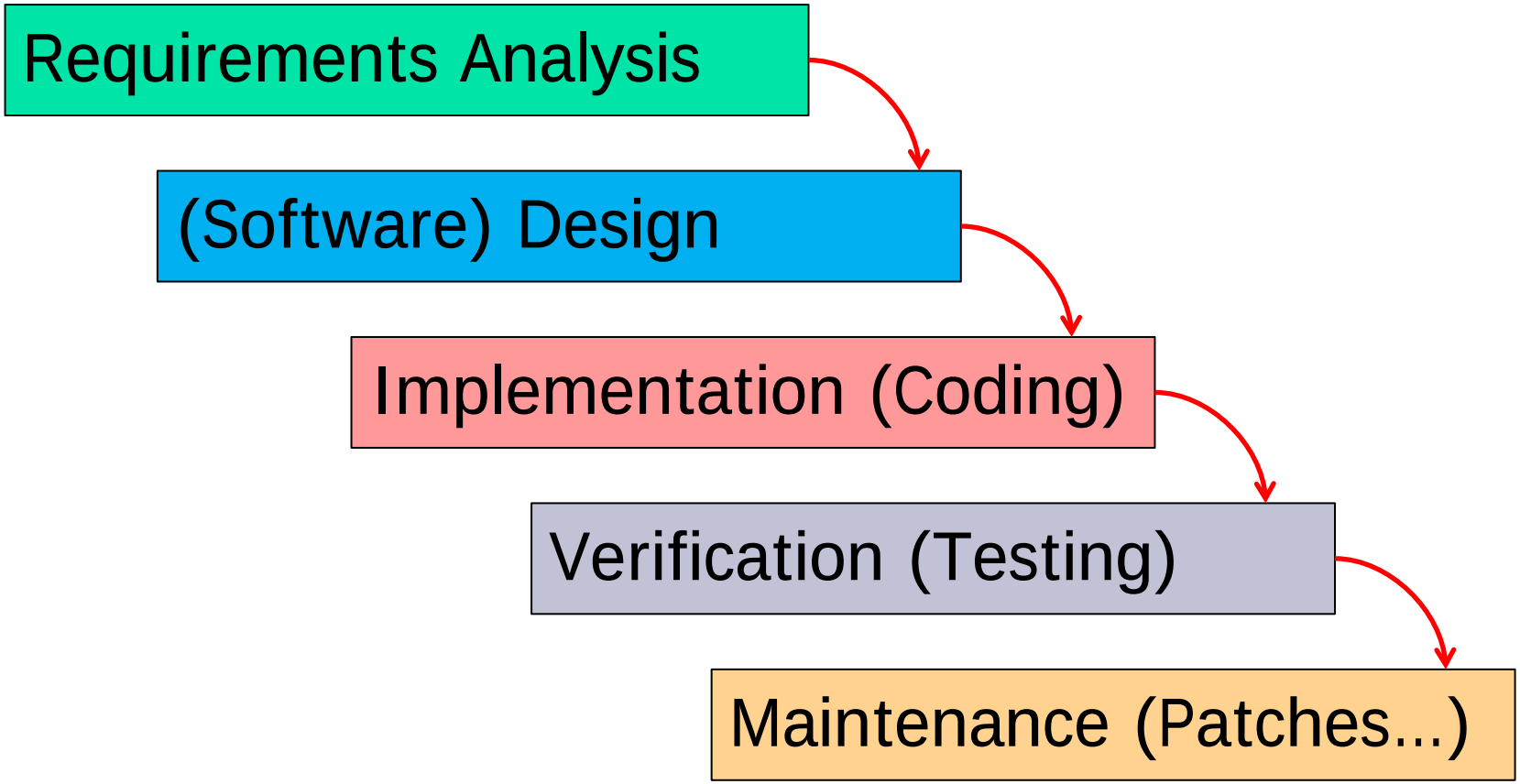
Meng He

Software Development Life Cycle (SDLC)

- SDLC is a general term that describes structure imposed on the development of a software product
- Purpose
  - To reduce the risk of missing the deadline
  - To ensure product quality
  - …
- Many models have been proposed to describe SDLC

The Waterfall Model

- A sequential design process



Waterfall: Advantages and Disadvantages

- Advantages
  - Natural and easy to understand: we use this for assignments
  - Widely used
  - Reinforces notion of "design before coding"
  - Clear milestones
- Disadvantages
  - Often not practical
    - Clients may change requirements
    - Designers may not be aware of implementation difficulties

The Rapid Prototyping Model

1. Gathering preliminary requirements
2. Fast prototyping
3. User evaluation of the prototype
4. Repeat the above steps if necessary
5. Discard the prototype and development the software using a formal process

Rapid Prototyping: Advantages and Disadvantages

- Advantages
  - Ensure that software products meets clients' requirements
  - Reduce time/cost if clients may request changes during the process
- Disadvantages
  - Adequate and appropriate user involvement may not always be possible
  - Cost of prototype development

More about Models

- There are many other models
- To be studied by the Software Engineering course
- Choose an appropriate model depending on the particular software to be developed

Software Testing

- Motivation
  - Robust software
  - Maintain reputation
  - Lower cost: fixing a bug before release is always cheaper than after release
  - …
- There are job positions on testing
  - Software engineer in test

What do We Test

- Whether a program works
- That is, whether it meets the specification
- Specification contains:
  - A description of input
  - A description of output
  - A set of conditions
  - Specifying what the output should be given input and conditions

How do We Test

- Mindset
  - How to make the program fail?
- Typical test cases
  - Regular cases
  - Boundary cases
  - Error cases

Types of Testing

- White box testing
  - Use internal knowledge of implementation to guide the selection of test cases
  - To achieve maximum code coverage
- Black box testing
  - Use specification to guide the selection of test cases
  - To achieve maximum coverage of cases given in the specification

Debugging

- Debugging: a methodical process of finding and reducing bugs, or defects, in a computer program
- The key step: Identifying where things go wrong
  - Track program state
    - Current location in the program
    - Current values of variables
    - Numbers of iterations through a loop
  - Find when expected program state does not match actual program state

Printf Debugging

- Idea: Use printf statement to print
  - Values of variables
  - Program location

- Example
  - printf(“Entering the second loop\n”);

Strategies of printf Debugging

- The linear approach
  - Start at the beginning of the program adding printfs
  - Until you reach the bug (state where your printout differs from what you expect)
- Binary search
  - Select half-way point
  - Determine if the bug has occurred
  - If yes, look in the first half
  - If no, look in the second half

Disadvantages of printf Debugging

- Time consuming for large programs
  - Modify program
  - Recompile
  - Rerun

Solution: gdb

- A symbolic, or source-level, debugger

- A program that allows programmer to
  - Access another program's state as it is running
  - Map the state to source code (variable names, line numbers, etc – why we compile with –g option)
  - View variable values
  - Set breakpoints

Breakpoints

- Internal pausing places in a program

- Breakpoints allow programmers to
  - Print values of variables
  - Step through code
  - Resume running the program till the next breakpoint

Commands

- See lab 5
- Notes about some commands
  - break line_number
  - break function_name
  - next: executes the next statement (function call = 1 statement)
  - step: executes the next statement, stepping into functions

Basic Operations

- Set breakpoints
- Examine variables at breakpoints or trace through code
- Until the bug is found
- Strategy: linear or binary search
- Advantage: No recompiling!