CS 216

Lecture 8

February 28th, 2014

Administrivia

Midterm is next Friday

Review session next Monday

You are responsible for:
Lectures, homework,
practica, programming
assignments

40 questions, multiple choice.

Nursing computer lab.

Singleton

Class with one instance.

Accessed by a static method, and has a private constructor.

Design Pattern

Reusable solution to a common problem.

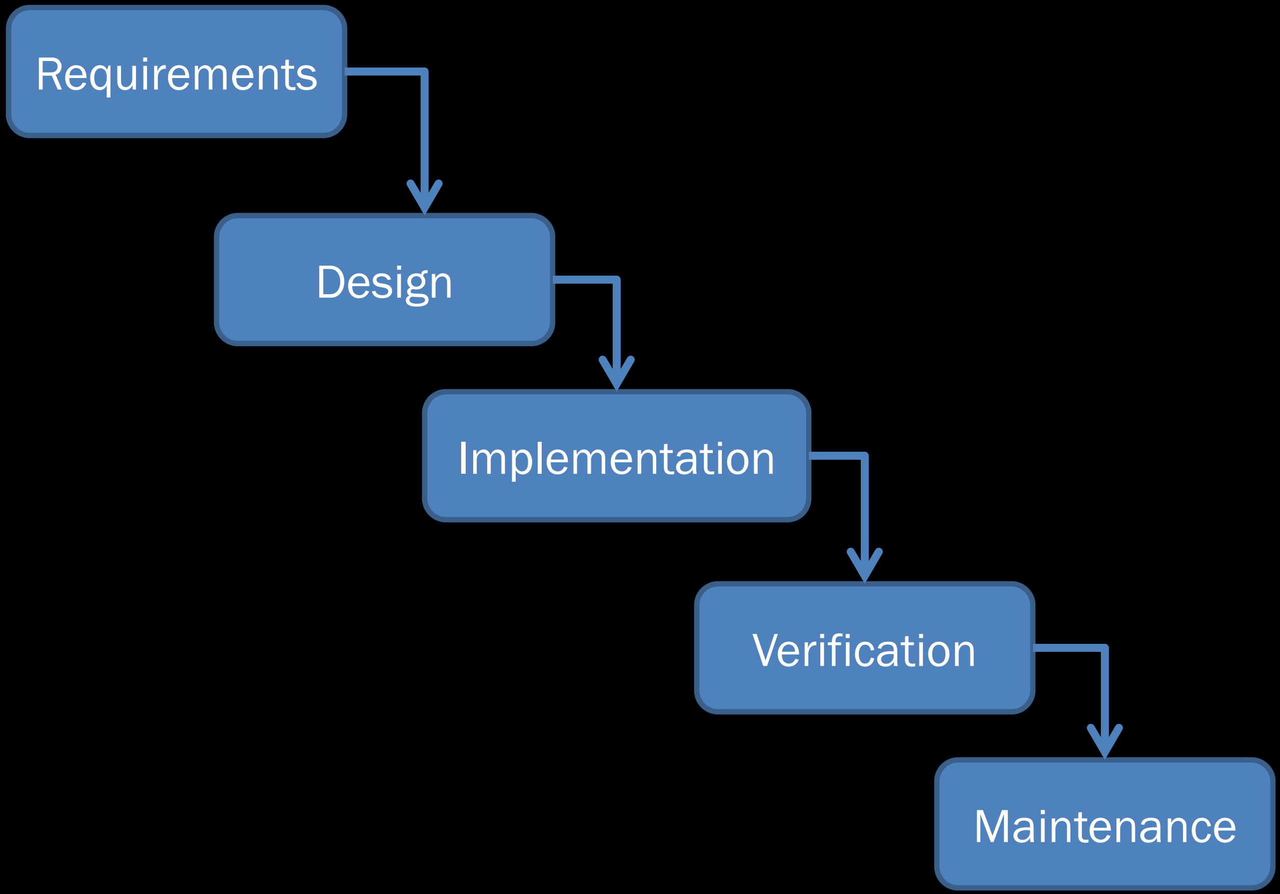
Iterator

Class Factory

The software engineering process

Development Models

The Waterfall Model



Requirements
Design
Implementation
Verification
Maintenance

This is the fallback for big projects (e.g., Office)

Sounds advantageous from the business side

Everything’s planned. You know what the software does from the start.

It is not without problems, of course.

It requires a lot of knowledge up front – at the requirements stage.

And these requirements have to be very good!

It’s hard to predict how much time something is going to take.

Things can change fast during development.

No battle plan ever survives contact with the enemy

- Helmuth von Moltke

So what’s the alternative?

Agile Development

Does away with singular distinct phases

Instead, you have “iterations” or “sprints”

Each sprint is supposed to be two weeks to a month and contains all phases.

Most agile processes place a heavy emphasis on personal communication.

What does this do for you?

Fast response to a changing environment.

Problems?

Often less efficient if the task is well understood from the start.

Has trouble
“ending”

The reality

Most really big projects start with a waterfall-like model.

The business case often wins out.

Once you hit a maintenance phase, things start looking more agile.

Maintenance

This is what happens after software is “done”.

General estimate is that about 80% of effort is spent on maintenance.

This isn’t a bad thing. Not by a long shot.

Maintenance just happens to be when the project is generating revenue.

(Or actually being *used*, for research or FOSS projects)

Test-Driven Development

You’re used to requirements.

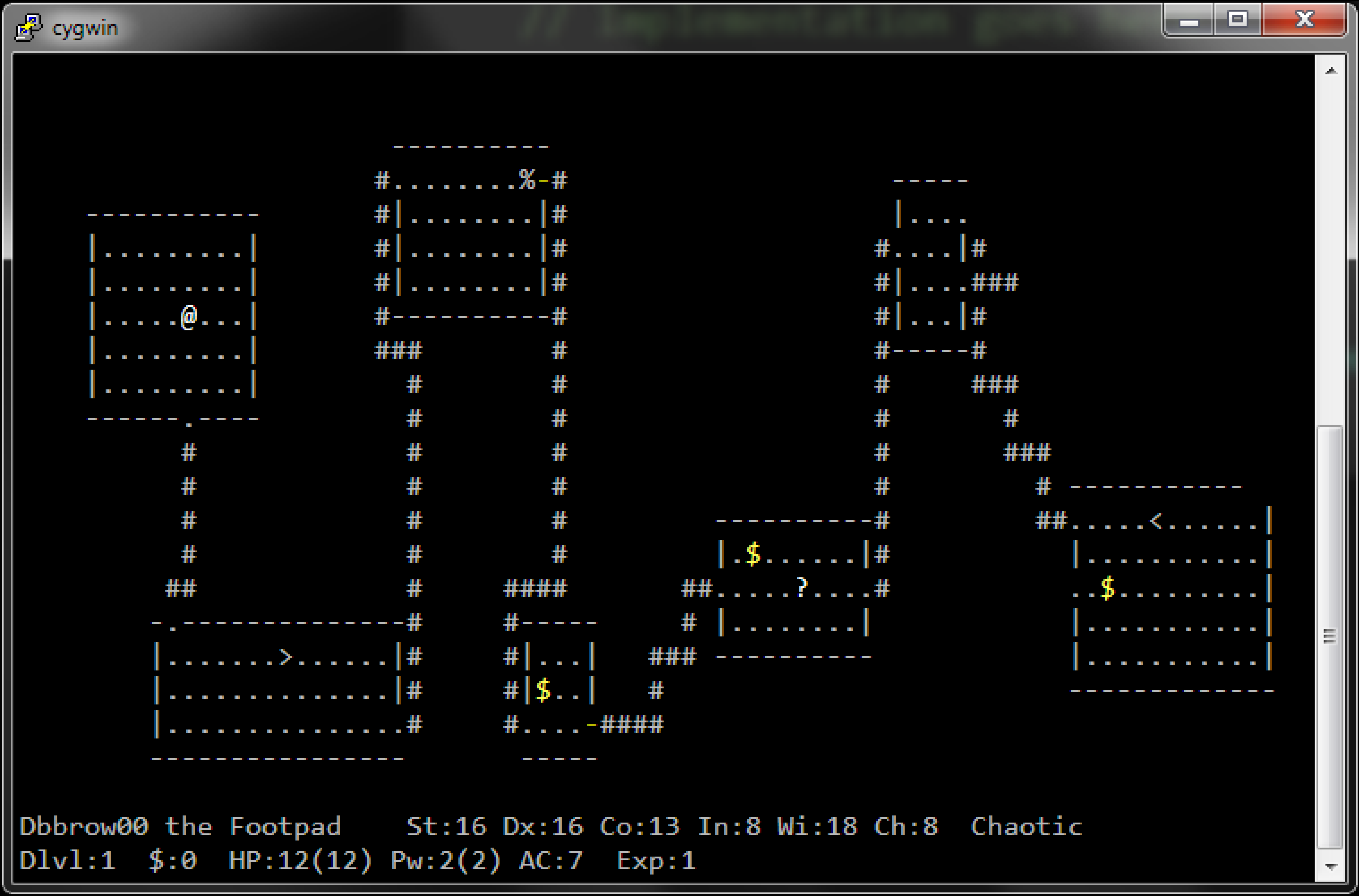
Another method of specification is to write the test cases first.

Why?

Deterministic representation of requirements.

Coding to a test tends to produce less waste than coding to an English requirement.

PA2



```
                       ----------
                      #........%-#                    -----
    -----------       #|........|#                    |....
    |.........|       #|........|#                   #....|#
    |.........|       #|........|#                   #|....###
    |.....@...|       #----------#                   #|...|#
    |.........|       ###        #                   #-----#
    |.........|         #        #                   #     ###
    ------.----         #        #                   #       #
          #             #        #                   #       ###
          #             #        #                   #         # -----------
          #             #        #         ----------#         ##.....<......|
          #             #        #         |.$......|#           |...........|
         ##             #     ####       ##.....?....#           ..$.........|
        -.--------------#     #-----     # |........|            |...........|
        |.......>......|#     #|...|   ### ----------            |...........|
        |..............|#     #|$..|   #                         -------------
        |...............#     #....-####
        ----------------       -----

Dbbrow00 the Footpad    St:16 Dx:16 Co:13 In:8 Wi:18 Ch:8  Chaotic
Dlvl:1  $:0  HP:12(12) Pw:2(2) AC:7  Exp:1
```