# CT5805701

營建資訊系統之軟體工程
Software Engineering in Construction Information Systems
BlackBoard: elearning.ntust.edu.tw

Time: 1:30PM – 4:20PM, Wednesday
Classroom: IB-508

Lecturer: Yo-Ming Hsieh (謝佑明)
ymhsieh@mail.ntust.edu.tw



# Discussions

- What is software ?
  - Coded instructions (programs) that make a computer do useful work.
  - The programs and instructions which direct a computer.
  - Programs written by programmers using computer languages.



# What is Software Engineering?

- Google: *define: software engineering*
  1. The system of applying of an engineering discipline to the design, implementation and maintenance of software systems.
  2. A collection of theories, techniques, and tools which enable fallible humans to design, construct and maintain large software products in a reliable and cost effective manner.
- Google: *define: system*

A system is an assemblage of inter-related elements comprising a unified whole. From the Latin and Greek, the term "system" meant to combine, to set up, to place together. A sub-system is a system which is part of another system. A system typically consists of components (or elements) which are connected together in order to facilitate the flow of information, matter or energy. ...



# Outline of this course

Part I. Development Methods
- Software Processes
- Time and resource estimation

Part II. Information systems
- Software modeling
- Database technology
- Virtualization technology
- Information system architecture
  - 2-tier applications
  - 3-tier web applications
- Web services
- Internet security



# Grading

- Assignments: 50 pts
  - The assignments will go through a complete cycle of software development.
  - The assignments will build ***a platform*** *for* ***exchanging second-hand textbooks***
    - 2-tier architecture (application + database)
    - 3-tier architecture (web client + web server + database)
    - Coding is minimal.
  - Assignments late within a week get a **10%** reduction in its grade.
  - Assignments late for more than a week will NOT be accepted.
- Two exams: 40 pts
- Class participation: 10 pts



# Lecture 1

Review of
Programming Fundamentals

## Today's Topic

- The process of small-scale software development
- Large-scale software projects
- Rapid software development strategy
  - Four general strategies
  - Four dimensions of development

7

## The process of small-scale software development

1. Requirement specification
2. Analysis
3. Design
4. Implementation
5. Testing and verification

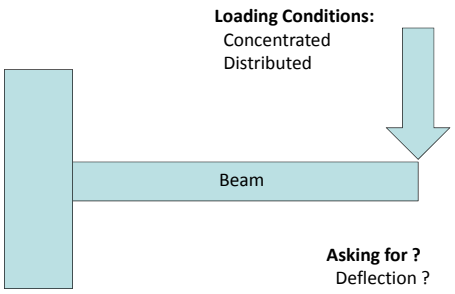
It should be noted that the process of software development is almost the same as problem-solving.

8

## The process of software development (1) Requirement Specification

- Understand EXACTLY what the problem is
- Understand what is needed to solve it
- What the solution should provide
- Constraints and special conditions
- How precisely we can define a problem depends on our degree of familiarity with the problem domain.
- If we are not familiar with the problem, we should either quickly acquire an education in it or contact people who are knowledgeable about it.

9




10

## What questions should you ask ?

Buy a meal at McDonald

Build a computer

Write a software

11

## The process of software development (2) Analysis

- Identify the following:
  - Input
  - Output
  - Special constraints or conditions
    - The process from input to output must be less than 1ms.
  - Formulas and equations to be used

12

**Output ?**

Deflection at end point

**Formula ?**

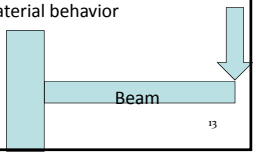
$$w(x) = -\frac{Px^2(3L-x)}{6EI} \qquad I = \frac{1}{12}bh^3$$

**Input ?**

$x, P, L, E, I, b, h$

**Special Constraints and Conditions?**

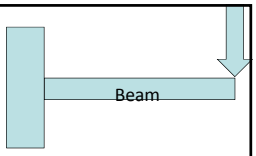
(The equation applies for elastic material behavior and homogeneous property) !!




13

**Output ?**

Beam section size b & h




**Formula ?**

$$w(x) = -\frac{Px^2(3L-x)}{6EI} \quad I = \frac{1}{12}bh^3$$

**Input ?**

$x, P, L, E$

**Special Constraints and Conditions?**

(The equation applies for elastic material behavior and homogeneous property) !!

The maximum deflection be less than 1mm.

The material has to be iron

14

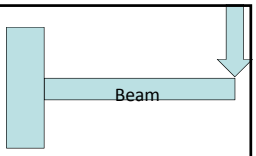
## The process of software development (3) Design

- Algorithm: the method of solution
  - An algorithm is a sequence of a finite number of steps arranged in a specific logical order, which, when executed, produce the solution for a problem.
  - An algorithm must satisfy some requirements:
    - Unambiguousness
    - Generality
    - Correctness
    - Finiteness
- Presentation of algorithms: "pseudo coding" or flowcharting

15

**Output ?**

Beam section size b & h




**Formula ?**

$$w(x) = -\frac{Px^2(3L-x)}{6EI} \quad I = \frac{1}{12}bh^3$$

**Input ?**

$P, L, E \qquad P = 100, L = 20$

**Special Constraints and Conditions?**

The maximum deflection be less than 3mm.

The material has to be iron

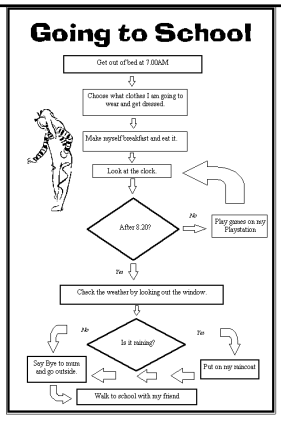
```
For E in (all possible E values for iron)
  Computed required I using ...
  For b in all possible width
    Compute h using ...
    Output b, h, and E
  End For
End For
```

16

## Flowchart

- Three basic symbols
  - Square: activities or tasks
  - Diamond: decision points
  - Arrows: flow of control
- Basic principles
  - Tasks have only one exit.
  - Tasks may have one or more entries.
  - Decisions have two or more exits. Each exit arrow should label the criterion.


http://www.edrawsoft.com/images/examples/Process-Flowchart.png




http://english.unitecnology.ac.nz/resources/units/pathways/flow_chart.gif

18

## Pseudo Code Example (1/2)

```
Main Procedure Monopoly_Game
   Hand out each player's initial money.
   Decide which player goes first.
   Repeat
      Call Procedure Monopoly_Move for next player.
      Decide if this player must drop out.
   Until all players except one have dropped out.
   Declare the surviving player to be the winner.
```

19

http://www.wiley.com/college/busin/icmis/oakman/outline/chap05/slides/pseudo.htm

## Pseudo Code Example (2/2)

```
Procedure Monopoly_Move
   Begin one's move.
   Throw the dice.
   Move the number of spaces on the board shown on the dice.
   If the token landed on "Go to Jail,"
      then go there immediately.
   Else if the token landed on "Chance" or "Community Chest,"
      then draw a card and follow its instructions.
   Else
      follow the usual rules for the square (buying property,
      paying rent, collecting $200 for passing "Go", etc.).
   End one's move.
```

20

http://www.wiley.com/college/busin/icmis/oakman/outline/chap05/slides/pseudo.htm

## The process of software development (4) Implementation

- Translate each step of the algorithm into a statement in that particular language and end up with a computer program.
- Programming errors
  - design errors
  - syntax errors
  - run-time errors

21

## The process of software development (5) Testing and Verification

- Testing
  - the process of executing a program to demonstrate its correctness
- Verification
  - the process of ensuring that a program meets user requirements

22

## Summary

The process of small-scale software development

**Large-scale software projects**

Rapid software development strategy (for large-scale software projects)

Four general strategies

Four dimensions of development

23

## Large-scale software projects

- For large scale software projects
  - Windows NT 4.0
    - http://www.cnn.com/2004/TECH/biztech/02/13/microsoft.source/
    - 28 millions lines of code, 95103 files
  - Windows 2000
    - http://www.dwheeler.com/sloc/
    - In 2001: 35 million lines of code
    - Windows XP: 40 million lines of code
    - Windows Vista: 50 million lines of code
  - Redhat Linux 7.1
    - http://www.dwheeler.com/sloc/
    - >30 million lines of code

24

## A good software project

- Correct
  - Meets user requirements
- On schedule
- Cost efficient

25

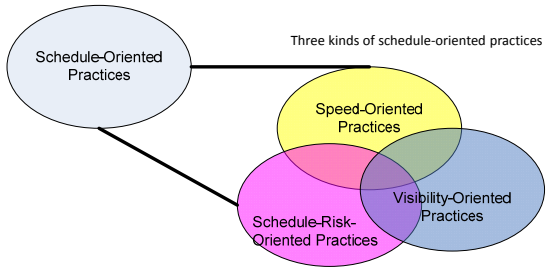
## General Strategy for Rapid Development (1/2)

1. Classic-mistake avoidance
2. Development fundamentals
3. Risk management
4. Schedule-oriented practices


26

Source: RD

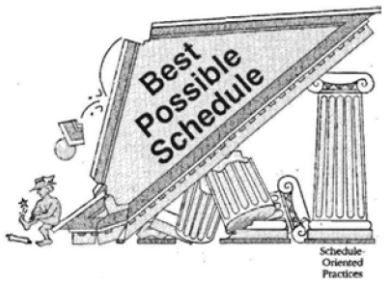
## Schedule Oriented Practices




27

Source: RD

## General Strategy for Rapid Development (2/2)

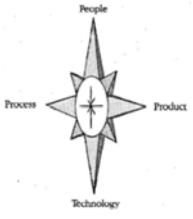



28

Source: RD

## Four Dimensions of Development

People
Process
Product
Technology




29

## People
### Peopleware

- "Peopleware"
- Peopleware may have more impact on software productivity and quality
  - Productivity with similar levels of experience varies by a factor of at least 10 to 1.
  - The performance of entire teams varies on the order of 3, 4, or 5 to 1.
  - "technology is not the answer; the most effective practices are those that leverage the human potential of their developers" ~ Basili et al. 1995
- Look into: staff selection, motivation, teamwork, and training

30

## People
### Staff Selection (1/2)

- Top talent – use better and fewer people
- Job matching – fit the task to the skills and motivation of the people available
- Career progression – help people to self-actualize than forcing them to work where they have the most experience or where they are most needed

31

## People
### Staff Selection (2/2)

- Team balance – select people who will complement and harmonize with each other
- Misfit elimination – eliminate and replace problem team members as quickly as possible
- Other factors – design ability, programming ability, programming-language experience, machine and environment experience, and application-area experience

32

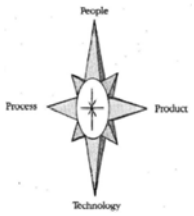
## People
### Team Organization & Motivation

- Tailor teams to match
  - Project size
  - Schedule goals
  - Product attributes
- Motivation
  - No factor other than motivation will cause a person to forsake evenings and weekends without being asked to do so.

33

## Four Dimensions of Development

People
Process
Product
Technology




34

## Process (1/3)

- Management and Technical Methodologies
- Process represents an area of high leverage in improving development speed
- Hughes aircraft, Lockheed, Motorola, NASA, Raytheon, and Xerox
  - explicitly focused on improving their development process
  - Cut time-to-market by about one-half
  - Reduced cost and defects by factors of 3 to 10

35

## Process (2/3)

- Rework avoidance
  - Avoid requirement change at late stages
- Quality Assurance (Q/A)
  - Assure the product has an acceptable level of quality
  - Detect errors at the stage when they are least time-consuming (and least costly) to correct. → catch errors as close as possible to the time that they are introduced
- Development fundamentals
  - Analysis → design → construction → integration → testing will not product lightning-fast schedules, but they prevent disasters
  - Half of the challenge of rapid development is avoiding disaster.

36

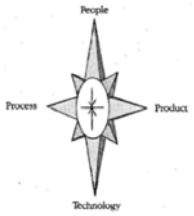
## Process (3/3)

- Risk management
- Resource Targeting
  - Get the most bang for your buck
- Lifecycle planning
  - Several lifecycle models to be introduced later
- Customer orientation
  - Develop software to its spec. is half job done
  - The other half is help the customer figure out what the product should be (Thus, requirement specification is very important).

37

## Four Dimensions of Development

People
Process
Product
Technology




38

## Product

- The most tangible dimension
  - product size
    - 80/20 rule
    - Additional features require additional specification, design, construction, testing, and integration
    - 1/2 produce size → 2/3 effort saving
  - product characteristics
    - Performance, memory footprint, robustness, reliability
    - Don't insist on too many priorities at once!

39

## Technology

- Platform, operation system, software components, development tools, …
- From assembly → high-level languages was one of the most influential changes in software-development history
- Integrated Development Environment, IDE
  - E.g. Microsoft Visual Studio, Eclipse, Borland C++ builder, …
  - Help manages the complexity of software projects when there are many classes and methods.
- Visual Programming
  - programming can be done by drag-and-drop without writing a single line of code

40

## Summary for 4 dimensions

People

Process

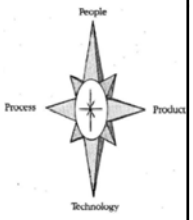
Product

Technology

41

## Which dimension matters the most

- Different projects have different needs
  - Accept the limitations on the dimensions you cannot change
  - Emphasize the other dimensions to get the rest of the schedule benefit you need
- Examples
  - Fuel-injection system for a car
  - In-house business program
  - A feature driven shrink-wrap market




42

## Characteristics of standard approaches to schedule-oriented development

| | Effect of Development Approach On | | |
|---|---|---|---|
| Development Approach | … Schedule | … Cost | … Product |
| Average practice | Average | Average | Average |
| Efficient development (balancing cost, schedule, and functionality | Better than average | Better than average | Better than average |
| Efficient development (tilted toward best schedule) | Much better than average | Somewhat better than average | Somewhat better than average |
| All-out rapid development | Fastest possible | Worse than average | Worse than average |

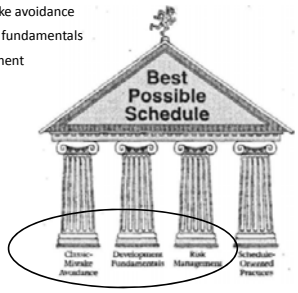
All-out rapid development: work as fast as you can, as hard as you can, and the only thing left to do at that point is to pay more, reduce the feature set, or reduce the product's polish.
compress schedule by 25% requires an 75% increase of team size
→ 33% cost increase

43

Source: RD

## Efficient Development

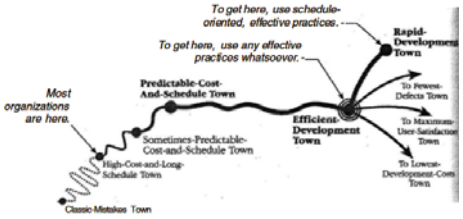
1. Classic-mistake avoidance
2. Development fundamentals
3. Risk management




44

Source: RD

## The Road to Rapid Development




45

Source: RD

## An Alternative Approach (1/2)

- How
  - Hiring the best people
  - Total commitment to the project
  - Full autonomy
  - Motivating them to an extreme degree → work 60, 80, or 100 hours a week until the completion of the project

46

## An Alternative Approach (2/2)

- Usually code-like-hell approach
- Hit-or-miss
- Causes long-term motivation problems
- Unrepeatable
- Hard on non-software organizations
- Wastes human resources extravagantly

47

## Summary

- The process of small-scale software development
- Large-scale software projects
- Rapid software development strategy
  - Four general strategies
  - Four dimensions of development

48

## Assignment #0

- Reading: Class Mistake Case Study
  - http://www.stevemcconnell.com/rdmistak.htm
- Identify class mistakes enumerated at:
  - http://www.stevemcconnell.com/rdenum.htm

49

## People

1. Undermined motivation
2. Weak personnel
3. Uncontrolled problem employees
4. Heroic
5. Adding people to a late project
6. Noisy, crowded offices
7. Friction between developers and customers
8. Unrealistic expectations
9. Lack of effective project sponsorship
10. Lack of stakeholder buy-in
11. Lack of user input
12. Politics placed over substance
13. Wishful thinking

50

## Process

14. Overly optimistic schedules
15. Insufficient risk management
16. Contractor failure
17. Insufficient planning
18. Abandonment of planning under pressure
19. Wasted time during the fuzzy front end
20. Shortchanged upstream activities
21. Inadequate design
22. Shortchanged quality assurance
23. Insufficient management controls
24. Premature or overly frequent convergence
25. Omitting necessary tasks from estimates
26. Planning to catch up later
27. Code-like-hell programming

51

## Product

28. Requirement gold-plating
29. Feature creep
30. Developer gold-plating
31. Push-me, pull-me negotiation
32. Research-oriented development

52

## Technology

33. Silver-bullet syndrome
34. Overestimated savings from new tools or methods
35. Switching tools in the middle of a project
36. Lack of automated source-code control

53

54