

CS121 - Software Development

- Administrative Introduction
 - course goals and focus
 - course structure and grading
 - academic honesty
- Introduction to Software Development
 - why is Software so difficult?
 - elements development methodology

1/16/2007

Introduction to S/W Development

1

Instructor: Mark Kampe

- not regular faculty, not an academic
- I am a retired engineer
 - spent 30 years building operating systems
 - done every job in an engineering organization
 - worked for small and large companies
 - active in architecture, process, and training
- I teach for fun
 - and to ensure you learn some engineering

1/16/2007

Introduction to S/W Development

2

Contact Information

- Office: Edmunds 2-221
extension 70969
- Office Hours: Tu/Th 12:00-13:00
Tu 16:00-17:00
and by appointment
- E-mail: markk@cs.pomona.edu

1/16/2007

Introduction to S/W Development

3

Why this course is important

- All of you will develop complex software
 - whether or not you work as programmers
- Software Construction is difficult
 - even simple programs can be hard to build
 - even carefully written programs behave badly
- This is the study of
 - problems inherent in software construction
 - tools and processes for addressing them
 - ways to improve your chances of success

1/16/2007

Introduction to S/W Development

4

Focus of this course

- medium-sized projects
 - multi-staff-month to multi-staff-century
- broad coverage
 - overview the full range of development activities
- comparative approach
 - not teaching one particular development discipline
 - an exploration of issues and approaches
- practical focus
 - what you'll use, rather than what's been written
- driven by a set of "key learning objectives"

1/16/2007

Introduction to S/W Development

5

Key Concepts and Issues

- I am here to help you master some concepts
 - They are fundamental to software construction
 - I believe they will be of great value to you
- The lectures are built around them
 - I will point them out, and give you a list of them
- The tests are built around them
 - can you describe or discuss them?
 - can you apply them to problem situations?
 - can you use them to gain new insights?

1/16/2007

Introduction to S/W Development

6

Representations and Techniques

- I am here to introduce basic methodology
 - languages you will understand and speak
 - skills you will develop and apply
 - processes you will understand and follow
- The Labs are designed around these
 - you will use all of these skills & techniques
- The Final Exam will test you on them
 - explain how to apply them to new problems

1/16/2007

Introduction to S/W Development

7

Texts and Reading

- There will be daily reading assignments
- There is one basic text:
McConnell, *Code Complete, 2nd ed*
 - GOOD: clear, focused, and practical advice
 - WEAK: little discussion of issues and principles
- This will be supplemented by many papers
 - GOOD: most of them short and highly focused
 - WEAK: they are of varying depth & quality

1/16/2007

Introduction to S/W Development

8

Interactive Lectures

- Lecture periods will not be used to
 - review subjects in the assigned reading
 - you are expected to have done the reading
- Lecture periods will be used to
 - clarify the reading and work examples
 - synthesize conclusions from divergent opinions
 - interactively explore implications & applications
- all lecture slides will be posted on-line
 - to simplify your note-taking and study
- Participation is critical and will be graded

1/16/2007

Introduction to S/W Development

9

Quizzes

- When: first five minutes of lecture period
- Scope: that day's assigned reading
- Format: closed book
 - 4-6 short and simple questions
- Goals:
 - check your familiarity with key concepts
 - force you to do the assigned reading
 - enable you to get more out of the lectures

1/16/2007

Introduction to S/W Development

10

Regular Exams

- When: eighth week, first 1/2 of final period
- Scope: last 8 weeks of reading & lectures
- Format: closed book
 - 10-15 short essay questions
 - very straight forward (if you're keeping up)
- Goals
 - test understanding of key concepts
 - test ability to apply principles to real problems

1/16/2007

Introduction to S/W Development

11

Planned Final Exam

- When: second 1/2 of final period
- Scope: all reading, lectures & labs
- Format: open book
 - 6-10 novel/difficult/multi-part essay questions
 - you get to choose a subset to answer
- Goals
 - test mastery of key concepts
 - ability to apply them to new problems
- expect an alternative after spring break

1/16/2007

Introduction to S/W Development

12

Projects

- five relatively small (2-3 week) projects
 - 1st to be done individually, the rest in teams
- Scope:
 - build automation and version control,
 - requirements, design, test plans, reviews
- Goals:
 - develop practical software development skills
 - develop teamwork skills
 - these are not programming projects

1/16/2007

Introduction to S/W Development

13

Course Web Site

- <http://cs.pomona.edu/classes/cs121>
 - reading, lecture, quiz and exam schedule
 - supplementary reading materials
 - project assignments, information, materials
 - copies of all lecture slides
 - lists of key points and definitions
 - solutions to exam/homework problems
- Sakai (one combined section)
 - grades, announcements, discussions

1/16/2007

Introduction to S/W Development

14

Course Grading

- Final course grade is a function of:
 - concepts 45% (measured on 3 exams)
 - skills 45% (exercised in 5 projects)
 - preparation 10% (24 daily quizzes)
 - with bonus points for discussion participation (quality, not quantity)
- I look at curve to assign final grades
 - but expect it to be close to 90/80/70/60

1/16/2007

Introduction to S/W Development

15

Late Assignments & Make-Ups

- Quizzes there are no make-ups
- Homework don't earn points
- Exams only for medical emergencies
 - you will need a letter from an attending MD
 - different tests, given after end of the semester
- Projects each person has 3 slip-days (team has sum of individuals')
after that it is 10% off per day

1/16/2007

Introduction to S/W Development

16

Grade Changes

- I will always fix any grading error
 - if I mis-record a score, show me the original
- I will always explain how I score problems
 - how many points for which parts of the answer
- I will always **consider** re-scoring a problem
 - if I miss or mis-understand your answer
 - but only if your answer is better than I thought
- I never change grades due to “hardship”
 - but I do give lectures on personal responsibility

1/16/2007

Introduction to S/W Development

17

Academic Honesty

- Academic Dishonesty cheats us all
 - devalues grades & degrees,
 - threatens our accreditation
- My Policy
 - zero tolerance, no second chances
 - no warnings, I report everything
- Don't test me

1/16/2007

Introduction to S/W Development

18

The Rules - Quizzes/Exams

- do not copy anyone else's answers
 - if you study with a friend, sit apart
- no talking or exchanging notes during test
 - if you have a question, raise your hand
- no use of notes or text on closed book tests
 - if you need a dictionary, tell me before the test
- no removal of test materials from classroom
 - all tests must be turned in to the instructor

1/16/2007

Introduction to S/W Development

19

The Rules - Projects

- You must prepare your own work products
 - in projects 2-5 you can work with team-mates
 - don't get solutions from other teams
 - don't submit solutions from web or elsewhere
 - if you didn't write it yourself, cite the source
- Protect yourself
 - don't share your solutions with others
- Some projects involve meetings w/others
 - requirements gathering and reviews are OK

1/16/2007

Introduction to S/W Development

20

What's new (for me) this time?

- the eclectic supplementary reading
 - much more focused (less waste material)
 - a range not found in any existing text book
- the more interactive lecture sessions
 - I need to develop some new techniques
 - I need to learn how to grade participation
- integrating projects into the class sessions
 - at UCLA these have been more distinct
- I anticipate many "learning experiences"

1/16/2007

Introduction to S/W Development

21

Introduction to S/W Development

- Why is software so difficult?
 - hardware v.s. software
 - what makes software so complex
 - the "Legacy" software problem
- The Software Engineering problem
 - causes and costs of bad software
 - problems and myths in s/w development
- Elements of a solution

1/16/2007

Introduction to S/W Development

22

Software v.s. Hardware

- How Software is easier
 - software costs less to build and ship
 - software can be replicated perfectly
 - software does not experience "wear"
- How Software is harder
 - software products are much more complex
 - software is very difficult to inspect or test
 - few engineers/technicians truly understand it
 - the industrial revolution was 200 years ago

1/16/2007

Introduction to S/W Development

23

Why Software is so Complex

- intrinsic complexity
 - size & complexity of functional requirements
 - number of "moving parts", modes of interaction
 - number of other systems with which it interacts
 - number of relevant environmental factors
- continuously changing requirements
 - ever-increasing user and data loads
 - ever-evolving interface requirements
 - functional requirements change at web speed

1/16/2007

Introduction to S/W Development

24

The Legacy Software Problem

- H/W obsolescence is not a major problem
 - most hardware “wears out” long before that point
 - next gen h/w will be spec’ed for next gen needs
- Software does not “wear out” (sic)
 - it will have the opportunity to become obsolete
- Enterprises don’t replace mission critical s/w
 - major investments in acquisition and training
 - changes are almost inevitably highly disruptive
- S/W is forced to survive/evolve much longer
 - this greatly complicates design and maintenance

1/16/2007

Introduction to S/W Development

25

Why We Ship Bad Software

- Time is more important than quality
 - our customers need the functionality
 - we need to book the income this quarter
 - we already have a bad “on-time” reputation
- We don’t actually know how bad it is
 - our testing methodology inadequate
 - we don’t know how it will be used anyway
- Developers are usually goaled on delivery
 - bugs are someone else’s problem

1/16/2007

Introduction to S/W Development

26

The Costs of Bad Software

- Costs to consumers
 - increased costs of ownership
 - lost and reduced productivity/business
 - direct costs of management and support
 - costs of software failures
 - loss of service, revenue, and customers
 - death & destruction directly caused by bugs
- Costs to producers
 - cost of late, over-budget, inadequate projects
 - cost of support greatly exceeds development

1/16/2007

Introduction to S/W Development

27

Problems To Be Solved

- the time and cost for s/w development
 - initial development and ongoing enhancement
- the quality of the delivered products
 - problem rates and severities
 - costs of support, fixes, and patches
- we cannot predict these
- we cannot measure these
- we cannot effectively manage these

1/16/2007

Introduction to S/W Development

28

Common S/W Myths/Delusions

- We’ve already got adequate standards.
- Process takes more time than it saves.
- You’ll know when it’ll be done when I finish.
- If it’s late, we can add more people.
- It works for me, ship that sucker.
- Once it’s done, we can see how good it is.
- If it’s got bugs, we’ll find and fix them.

1/16/2007

Introduction to S/W Development

29

A Formula for Failure

- today’s dominant s/w development paradigm:

```
do {  
    have an idea;  
    build it;  
    ship it;  
    see how it works;  
} until (we give up);
```

- we aren’t sure what we should build
- we don’t know how good it is going to be
 - and we still don’t know this after we’ve built it

1/16/2007

Introduction to S/W Development

30

Software Engineering

The establishment and use of sound engineering principles, in order to reliably and economically obtain software that satisfies the user's requirements, is reliable, and works efficiently in real deployment environments.

1/16/2007

Introduction to S/W Development

31

Basic Methodology

- understand the problem
 - project definition, requirements development
- plan the solution
 - architectural design, modeling, prototyping
 - project planning (risks, resources, schedules)
 - User Interface & component design
- execute the plan
 - reviews, implementation, testing, integration
 - validation, deployment, support

1/16/2007

Introduction to S/W Development

32

Back up slides

Homework

- **When:** may be given a few times
- **Scope:** a few troublesome areas
- **Format:** a few focused problems
- **Goals:**
 - reinforce key concepts, issues & skills
 - test ability to apply key concepts
 - work areas where students often have trouble
- **Grading:** not included in course grade
 - I will correct and return for your feedback

1/16/2007

Introduction to S/W Development

34

The Rules – Homework

- It is OK to study with friends
 - discussing problems helps us understand them
- You must do your homework by yourself
 - do not submit another student's work
 - do not submit solutions you find on the web
 - if you do research **cite your sources**
- I decide when assignments are too similar
 - and forward them immediately to the Dean

1/16/2007

Introduction to S/W Development

35

Grading - Partial & Extra Credit

- **Partial Credit**
 - available on all quizzes, exams and projects
 - points are not merely for the final answer
 - points for a clear understanding of the problem
 - points for a reasonable approach to the problem
 - points for correct elements in a flawed solution
- **Extra Credit**
 - extra credit exam problems
 - for any answer that shows significant insight

1/16/2007

Introduction to S/W Development

36