CATECS Announcements

- In-Class Students
  - CATECS has a busy studio schedule
    - Be sure to exit promptly so next class can begin on time
  - Food and Drink are not technically allowed
    - Drinks are tolerated
      - as long as you keep the studio clean!

Live-Site Students

- Place speakerphone away from the TV
  - Make sure it's pointed away from the TV
- If you have connection problems
  - Hang up, wait 15 seconds, then call again
- If your speakerphone has a mute button
  - Use it when not talking!

Class Participation

- I expect you to participate!
  - Questions
    - “Stupid questions” -- No such thing
  - Discussion
    - “Silent Tomb” -- Not allowed
- CATECS students
  - Live-site students (same as above)
  - Tape students (via e-mail)
The Instructor

• Ken Anderson
  – Office Hours: ECOT 523
    • By Appointment only...
    • Send me e-mail to set an appointment
  – E-mail
    • <kena@cs.colorado.edu>
  – Phone
    • +1.303.492.6003

The Instructor, continued

• Ken Anderson
  – Mailing Address
    Dr. Kenneth M. Anderson
    University of Colorado, Boulder
    Department of Computer Science
    ECOT 717, Campus Box 430
    Boulder, CO 80309-0430
  – Department FAX
    • +1.303.492.2844

The Instructor, Background

• Assistant Professor
  – Fourth semester, taught 5828 last Spring
  – Ph.D. from University of California, Irvine
  – Research Topics
    • Open Hypermedia
    • Software Engineering
  – Software Experience
    • Three Systems ranging from 30K-60K LOC

Teaching Philosophy

• “sage-on-stage” vs. “guide-at-your-side”
• lecture vs. participation
• Answering questions
  – Sometimes the answer will be “I don’t know!”
• I welcome comments and questions from students!
Useful URLs

- CATECS
  - <http://www.colorado.edu/ContinuingEducation/CATECS/>
- Computer Science Department
  - <http://www.cs.colorado.edu/>
- Instructor’s Homepage
  - <http://www.cs.colorado.edu/~kena/>
- Class Homepage
  - <http://www.cs.colorado.edu/~kena/classes/5828/s00/>

About the Class Website

- You have one continuous homework assignment this semester:
  - Check the class website EVERY day
    - Preferably more than once each day
- Website will be your source for
  - Class schedule
  - Homework assignments
  - Pointers to class-related information

Prerequisites

- Background in Basic SE Concepts
  - Software Systems
  - Software Lifecycles
    - Requirements
    - Design
    - Implementation
    - Maintenance
  - Software Tools (e.g. make, rcs, etc.)

Currently-Planned Course Topics

- Basic Principles of Software Engineering
  - Essentially a review
- Formal Software Specification Techniques
- Fred Brooks
  - Mythical Man-Month
  - No Silver Bullet
  - 20th-year Reflections
- “Hot” Topics
Course Evaluation

- Fred Brooks Paper 30%
- Semester Project 70%
- --------------------------- 100%
- Total

Homeworks “fine-tuning” of grade

No Exams

General Notes on Assignments

- Electronic Submission OK
  - Text or Postscript/PDF formats only
  - You will probably want to use paper for homework assignments, however
    - CATECS requires the following information on the first page of all assignments
      - student name, course number, company name, assignment name or number

Homework Assignments

- Format
  - Examine the SE literature in more depth
  - Practice the techniques covered in class
- Typically one-week in length
  - (CATECS students will be one week behind)
  - Some assignments may be allocated more time based on difficulty

Semester Project

- Explore a topic of the class in-depth
  - Examples
    - Investigate a specification language not covered in class
    - Specify a program’s behavior with Petri-Nets
    - Build an analysis tool
    - Analyze your company’s software lifecycle
  - Work will thus vary across projects
    - Éffort should be equivalent to a 25 page paper
- Project proposals are due February 3rd
  - I will send out examples of previous projects
Example Project Description

• <Show Example Project Description>

Fred Brooks Paper

• 10 page paper
• Identify a theme
  – Critically evaluate it
  – Show how Brooks develops the idea and supports it
  – (If possible) relate it to your present-day work experience
• Submit paper ideas via e-mail for approval

Course Textbooks

• Fundamentals of Software Engineering
  – by Ghezzi, Jazayeri, and Mandrioli
  – © 1991
• The Mythical Man-Month
  • 20th Anniversary Edition
  – by Fred Brooks
  – © 1975, 1995

Historical Background: 30 years

• First Software Engineering Conference
  – NATO-sponsored conference in 1968
• “Software Crisis”
  – Systems were designed by identifying the hardware first
    • Software was allocated about 1-2% of the budget
  – However, software was causing all the problems (!) and thus needed more attention
Progression of SE

- An evolution of the programming activity
  - Early stages of computing
    - User/Developer were the same person
    - Problems were well-understood
      - First programs calculated metrics about artillery shells for the Navy!
  - High level languages began to appear in the 1950s
    - Along with the profession of “programmer”

SE Progression, continued

- 1960’s
  - Large Software Systems for Commercial Ventures
    - Teams of Programmers
    - Separate end-users
    - Complex Problems
  - “Software Crisis” coined as problems became apparent

The problem?

- Software is typically
  - late
  - over budget
  - faulty
  - costly to maintain
  - difficult to evolve
  - etc.

Consider the following:

- Loss of NASA’s Mars Climate Observer
  - due to conversion error of English and Metric units!
- Leap-year bug
  - A supermarket was fined $1000 for having meat around 1 day too long on Feb. 29, 1988
- Denver International Airport
  - Luggage system: 16 months late, 3.2 billion dollars over budget!
- <other examples>
SE Progression, continued

• 1968
  – Software Engineering formed
  – Many “solutions” put forward
    • New approaches to Project Management
    • New Team Organizations
    • Better Languages and Tools
    • Organizational Standards

• And here we are 30 years later! :-)

Intuitive notions of SE

• Programming?
• Chemist vs. Chemical Engineer
  – How does this analogy apply to SE?

Software Engineering

• Software
  – Computer programs and their related artifacts
    • e.g. requirements documents, design documents, test cases, specifications, protocol documents, UI guidelines, usability tests, ...

• Engineering
  – The application of scientific principles in the context of practical constraints

What is Engineering?

• Engineering is
  – a sequence of well-defined, precisely-stated, sound steps, which follow a method or apply a technique based on some combination of
    • theoretical results derived from a formal model
    • empirical adjustments for unmodeled phenomenon
    • rules of thumb based on experience

• This definition is independent of purpose...
  – i.e. engineering can be applied to many disciplines
Software Engineering (Daniel M. Berry)

- Software engineering is that form of engineering that applies:
  - a systematic, disciplined, quantifiable approach,
  - the principles of computer science, design, engineering, management, mathematics, psychology, sociology, and other disciplines,
- to creating, developing, operating, and maintaining cost-effective, reliably correct, high-quality solutions to software problems.

Software Engineering

- the study of software process, requirements and design notations, implementation strategies, and testing techniques
- the production of quality software, delivered on-time, within budget, and satisfying its users’ needs
- halfway between a discipline and an art form(!)