Introduction to 15-410/605

Dave Eckhardt
de0u@andrew.cmu.edu

Dave O'Hallaron
droh@cs.cmu.edu

Course Numbers

- Undergraduate ⇒ 15-410
- ECE M.S. students ⇒ probably want 15-605
- SCS M.S. students, INI M.S. students ⇒ 15-605
- Ph.D. students ⇒ might want 15-799A
  - Probably not this semester, but could be S'17?
  - Discussed with your advisor? See Prof Eckhardt?
- Other – consult your advisor
  - Your advisor *must* contact Prof Prof Eckhardt

Wait List

- Registrar's wait-list order is *irrelevant*
  - He has his ordering, we have ours
  - We admit based on readiness (mixed with need)
  - Usually *our estimate centers on your advisor*
- There may not be room for everybody
  - Some students will need to try again next semester
- If you're not on the wait list yet, you are *invisible*
  - Invisible students *definitely* won't get into the course!
  - If you are invisible, send mail *before noon today*

Wait List

- Background material (15-213) *is not optional*
  - M.S. students: take 213 and get an A (B *may* be ok)
  - Ph.D. students: have your advisor contact Prof Eckhardt
- Rare exceptions exist
  - Took a course with the 213 textbook – see Prof Eckhardt
  - Multiple years of specific industry experience – consult your advisor (today)
- Otherwise, please switch to 213 wait list instead
- 213 may not be enough (depending on background)

Wait List

- ECE
  - Seniors: likely; encourage your advisor to contact Prof Eckhardt
  - Juniors: plausible; encourage your advisor to help Prof Eckhardt prioritize
  - M.S.: if you got “talk to your advisor” mail, must; otherwise: plausible
- INI
  - 50/50? I am awaiting input from INI
- Others? *I must hear from your advisor!*
- *You must read your e-mail!!!*

Logistical Query #1

- Who has a class that conflicts with the 410 lecture?
  - Contact Prof Eckardt after class (potential for big trouble)

Logistical Query #2

- Who had trouble with 213?
  - Contact Prof Eckhardt after class (potential for big trouble)
  - ***If you didn't get a B or an A, see him***
  - ***If the malloc() lab didn't go well, see him***

Self-Assessment

- Self-assessment exercise on course web site?
  - Not mandatory if you did well in 15-213
  - A very good sanity-check, though!

Textbook (traditional)

- Option 1
  - Operating System Concepts, 8th edition
    - Silberschatz, Galvin, & Gagne
- Multiple “cheap” options exist!
  - eBay/Amazon/Alibris/...
  - If you try an e-book edition instead of paper, please tell us if you like it
  - Used copies of 7th edition work pretty well
    - Web site lists reading assignments for 6th through 8th editions

Textbook (experimental)

- Option 2
  - Operating Systems: Principles & Practice
    - Anderson & Dahlin
- Main differences
  - More focus on typical modern kernels and hardware
  - Less focus on historical systems
  - Stronger coverage of file systems and storage
  - Weaker coverage of security
- Available online

Textbook (which one?)

- We think you can use either one
  - Heavily-tested material is typically covered in lecture and projects
- We are interested in your opinion!
  - Which one, physical book vs. e-book, e-book purchase vs. rental...
  - We will ask for your thoughts at the end of the semester

Outline

- People
- Administrative information
  - Academic conduct
- Class goals
- Reading material

Dave Eckhardt

- Associate Teaching Professor, CS
  - Ph.D., Computer Science, CMU, 2002
    - “An Internet-style Approach to Managing Wireless Link Errors”
  - http://www.cs.cmu.edu/~davide
- Building Unix kernels since ~1985
  - PDP-11, Version 7 Unix
  - “Not really a BSD bigot”

Dave O'Hallaron

- Professor, CS and ECE
  - Ph.D., CS, Univ of Virginia, 1986
    - "Models for Concurrent Programming"
  - http://www.cs.cmu.edu/~droh
- Former Director, Intel Labs Pittsburgh
- co-author: Bryant and O'Hallaron, "Computer Systems: A Programmer's Perspective: 3rd Edition"
- co-creator (with R. Bryant) of 15-213
- Research: High Performance Computing

TA's

- Mixture of “repeat offenders” and “this year's model”
- As a team
  - Strong background
  - Here to help!

Yinz - Reading

- Read a Ph.D. thesis?
- Academic journal article?
- Attended an academic conference?
- Read a non-class CS book last semester?

Information Sources

Web site http://www.cs.cmu.edu/~410

- You are ***utterly required*** to read the syllabus

Q: Can I used a linked list for...?

Q: I have a final exam conflict...

Q: The license server is down...

Q: AFS says “no such device”...

- A: staff-410@cs.cmu.edu

Information Sources

Q: I am experiencing [delicate situation X] ...

A: e-mail to faculty

Note: most likely no Piazza this semester

- Experiment was run in a previous semester
- Results equivocal

Course Goals

- Operating Systems
  - What they are
  - Design decisions
  - Actual construction
- Team programming
  - Design, documentation
  - Source control
  - People skills

Course Plan

- Lectures
  - ***Many*** topics will be covered by text
  - But skipping many lectures *will* challenge your grade
    - The map is not the terrain, the slides are not the lecture
    - You will miss Q&A
  - We expect you to attend lectures
    - Details: see syllabus

Course Plan

- Projects
  - "Stack crawler" - readiness check ***[1-person project]***
  - Bare-machine video game ***[1-person project]***
  - Thread library
  - OS kernel
  - Kernel extension
- Project environment
  - Wind River Simics™ PC simulator
  - Your projects can also run on real PC hardware

Course Plan

- Homework assignments
  - ~2, to deepen understanding of selected topics
- Reading assignment
  - Pick something fun, write a ***brief*** report
- Mid-term, Final exam
  - Closed-book

Team programming

- Why?
  - Allows attacking larger problems
  - Teaches ***job skills*** you will need
    - Setting milestones
    - Setting up a productive work flow
    - Involving “management” before it's too late
- Team programming != “software engineering”
  - No requirement analysis
  - No release staging, design for growth, ...
  - Not a complete “life cycle”

Health Problems

- ***Somebody*** will probably get mono or pneumonia
  - If not, only because of something more creative
- Work-blocking health problem?
  - Go ***early*** to University Health (etc.)
  - ***Avoid*** "For the past two weeks I dragged myself to class but couldn't focus on programming"
  - Try to get paper documentation of work restrictions
  - Your program administrator will inform instructors
    - CS: cathyf@cs ; ECE: jmpeters@ece / woodhead@andrew ; ...

Partner Problems

- ***Somebody*** will have serious partner trouble
  - You need to "involve management" early
    - Sometimes (50%) we can fix the problem
    - If the problem can't be fixed, we can reduce the fallout
      - ...only if we know while the trouble is happening
  - *Don't* "buffer up" partner trouble until the last week of classes
    - At that point, we basically can't help
  - Details: see syllabus

Academic honesty

- See syllabus!
  - Reading the syllabus on this topic is not optional
- Learning is good
  - ...practices which avoid learning are ***double-plus ungood***
- Plagiarism is bad
  - ...credit ***must*** be given where due
- "Outside code" is *not* a simple yes/no issue
  - You *must not read any outside code* without carefully consulting the syllabus

Academic conduct

- Being a partner
  - Responsible
    - I am writing three grad school applications next week
  - Irresponsible
    - [vanish for 1 week, drop class]

Closing

- comp.risks newsgroup
  - Developers should read this
  - Managers should read this
  - Journalists should read this
- OSC textbook
  - Chapters 1, 2; Chapter 13.1, 13.2, 13.3.3
- OS:P+P textbook
  - Chapters 1, 2; Sections 3.0, 3.5; Section 11.3
- ***Start choosing a partner for P2/P3***

Grading philosophy

- C – all parts of problem addressed
- B – solution is complete, stable, robust
- A – excellent
  - Somebody might want to re-use some of your code
- Numbers
  - A = 90-100%, B = 80-90%, ... (roughly)
- "Curving"? Maybe, not necessarily
  - Lots of A's would be ***fine with us***
  - ***But this requires clean, communicative code!***

Todd Mowry

- Professor, CS
  - Ph.D., EE, Stanford, 1994
    - "Tolerating Latency Through Software-Controlled Data Prefetching"
  - http://www.cs.cmu.edu/~tcm
- Former Director, Intel Labs Pittsburgh
- Research: Log-Based Architectures

Todd Mowry

- Professor, CS
  - Ph.D., EE, Stanford, 1994
    - "Tolerating Latency Through Software-Controlled Data Prefetching"
  - http://www.cs.cmu.edu/~tcm
- Former Director, Intel Labs Pittsburgh
- Research: Log-Based Architectures

Carl Sandburg NHS, nps.gov

Todd Mowry

- Professor, CS
  - Ph.D., EE, Stanford, 1994
    - "Tolerating Latency Through Software-Controlled Data Prefetching"
  - http://www.cs.cmu.edu/~tcm
- Former Director, Intel Labs Pittsburgh
- Research: Log-Based Architectures

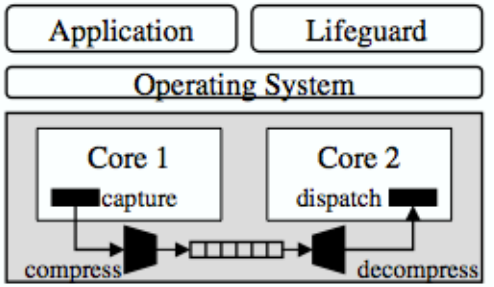

Figure 1: Dual-core LBA system

Garth Gibson

- Professor of CS and ECE
  - Ph.D., Computer Science, Berkeley, 1991
    - "Redundant Disk Arrays: Reliable, Parallel Secondary Storage"
  - http://www.cs.cmu.edu/~garth
- Research: Big Data Systems
  - Founder, CMU Parallel Data Lab (pdl.cmu.edu)
  - Founder, Panasas, Inc. (scalable object storage; panasas.com)
  - Manages 1.5 TF Hadoop/MapReduce cluster for CMU eScience (www2.pdl.cmu.edu)
  - Manages Linux pNFS prototype developers (wiki.linux-nfs.org)

Garth Gibson

- Professor of CS and ECE
  - Ph.D., Computer Science, Berkeley, 1991
    - "Redundant Disk Arrays: Reliable, Parallel Secondary Storage"
  - http://www.cs.cmu.edu/~garth
- Research: Big Data Systems
  - Founder, CMU Parallel Data Lab (pdl.cmu.edu)
  - Founder, Panasas, Inc. (scalable object storage; panasas.com)
  - Manages 1.5 TF Hadoop/MapReduce cluster for CMU eScience (www2.pdl.cmu.edu)
  - Manages Linux pNFS prototype developers (wiki.linux-nfs.org)

Roger Dannenberg

- Professor of CS, Art & Music
  - Ph.D., Computer Science, CMU, 1982
    - "Resource Sharing in a Network of Personal Computers"
  - www.cs.cmu.edu/~rbd
- Research: Computer Music (Systems)
  - Co-designer of Audacity audio editor
  - Patents behind SmartMusic & Rock Prodigy
  - Co-director of Music and Technology B.S. & M.S.
- Performed live on MTV

Yinz - Background

- Junior/senior/other?
- CS/ECE/INI/other?
- Group programming before?
- Done a branch merge before?

Yinz – Career plans

- Industry
- Graduate school
- Law/med/business school?
- Mountain top?

Team programming – Styles

- Waterfall model
- Spiral model
- “Extreme Programming”
- “Pair Programming”
  - Williams & Kessler, Pair Programming
- What you choose is up to you
  - This is an opportunity to read about models

Team programming - Design

- Decomposition into modules
  - (Yes, we expect modularity even in C!)
- Design for ***team implementation***
  - May need to adjust design to work in parallel

Team programming - Documentation

- For the non-compiler consumers of source code
- Doxygen documentation extraction system
  - Embed documentation in comments
  - Generate HTML index
  - Generate LaTeX
  - ...
- We intend to ***read your documentation***
- We intend to ***read your code***

Team programming - Source control

- Other buzzwords
  - Revision control, configuration management
- Goals
  - Re-create past builds
  - Compare stable states
  - Control inter-developer interference
  - [Manage multiple shipped product versions]

Team programming - Source control

- Even for “small” projects?
  - “It worked 3 hours ago, now it dies on start-up”
  - “I thought I fixed that already!”

- Most students who really try it keep using it

Team programming - people skills

- Working with other people is ***hard***
  - People think differently
  - People plan differently
- Pre-planning
  - Agree on work style, arrangements
    - Setting milestones
    - Pre-scheduled common time slots
- Handling problems
  - Involving “management” before it's too late
    - See syllabus