# CSE 60641: Operating Systems

**http://cse.nd.edu/courses/cse60641/www/**

- Instructor: **Surendar** Chandra (surendar@nd.edu)
  - Office Hours: Thu: 2:00-3:00 Fri: 3:00-4:00 (other times, by email appt)
  - Email is the best way to reach me, I am also logged into AIM: surendar
- TA: None
- Listserv: cse60641-01-fa08@listserv.nd.edu
  - Send questions to list so that everyone can follow along

# Prerequisites

- We read one paper a day (with some exceptions)
- CSE 30341 (undergraduate Operating Systems)
  - http://www.cse.nd.edu/~surendar/teach/spr08/cse30341/
  - Each lecture has audio/video/slides
  - Go over materials from ugrad OS. Take the exams to be sure that you are prepared
- If you are not prepared, take ugrad OS in Spr 2009 and then take grad OS if Fall 2009.
  - You need good grades for a Quals pass

# Introductions

- Tell us your name, academic background, research interests
- What do you hope to achieve in this class
  - core course, so something beyond “I had no choice”
- Machines that you own, OS you have worked with
- Prior OS experience

# Outline for today

- Course policies:
  - Course organization and expectation
  - Grading policy, late policy, reevaluation policy
  - Academic honesty

# Course Organization

- Read and discuss "seminal" papers
  - Not designed to introduce latest OS research
    - Good venues: SOSP, OSDI, ASPLOS, TOCS
- You will read the assigned paper before class.
  - Email me a brief summary of the main points of a particular paper, points that you liked and disliked by 5:00 pm on the day prior to the actual lecture. You are strongly encouraged to discuss the ideas with other colleagues
  - Questions to take home assignments and exams will test your understanding of the papers. Once the relevant lecture was over, you are not allowed the discuss the paper with anyone else (till the due date for the relevant assignment). All assignments and exams are individual effort and will test your understanding of the material.

# Course Project(s)

- Course project designed to give you hands-on experience. This is not a course to test your programming skills. To the extent possible, I am language/operating systems/project details agnostic. I do not require code submissions. I want you to summarize what you did and what you found
  - Identify a problem and motivate others why everyone should care about this particular problem
  - Develop your solution strategy and argue why it would be practical to implement
  - Perform experiments to verify your hypothesis
  - Argue how your experiments prove your point
  - Novelty is not important, it is acceptable to repeat the steps from some published paper

# Projects (2) – groups of size two

- First one – measure something, second one – solve some problem identified by the measurement
  - Talk to me if you would rather do something else
- Project report should be electronically turned in with a succinct and report (6 readable pages - you can use ACM, USENIX or IEEE style) on your implementation strategy and what you learned
- You do not need to submit the code for the project unless there is some dispute about the validity of the results

# Resources

- Cushing 208 lab
  - 6 desktops with 2 GB main memory, dual core
    - expsys-desktop1 through expsys-desktop6
  - Miscellaneous: Cerfcube, iPAQ, laptops etc.
  - HP Grant:
    - 1 - 2x1.5 GHz Itanium2 with 8 GB of memory ($40K)
    - 1 - 4x1.5 GHz Itanium2 with 8 GB of memory ($80K)
  - All these machines are exclusively for this class
  - Try out whatever you want to one these machines, it is okay to break the software on these machines (but not the hardware, don't drop the machines!!)

# Systems philosophy

1. Results from your own implementation

2. Results from your own simulation

3. Implementation results from a paper

4. Simulation results from a paper

5. Everything else (hearsay, rumors, “I think so”, “I think that it is how it should work” etc)

# Course grades

- Exit interview (10%)
  - Test how well you tie in what we learnt in lectures with projects
- Mid term (10%), Final exam (20%)
  - Open book, open notes, individual, take home
- Assignments: 30% (5x6% ea.)
  - Open book, open notes, individual, take home
- Group project: 20% (2x15% ea.)
  - Project1 (15%): measurement based
  - Project2 (15%): solution based

# Reevaluation policy

- Arithmetic errors, missed grading will be reevaluated.
- I encourage you to discuss concerns with your solution with me
- I discourage re-evaluation of partial credits:
  - Football penalty policy:

    If you think you deserve a better partial grade, write down the reason why you think that you deserve a better grade and how many extra points you think you deserve. If I agree, you could get up to this many extra points. If I disagree, you will lose this much points.

# Late policy

- None – Projects/homework/critiques are due at 11:00 am (right before the beginning of class) and 5 pm prior day. **I do not accept late submissions** (not even a second)

- Please contact me regarding <u>unforeseen emergencies</u>

# Academic Honesty

- Freedom of information rule:
  - Collaboration is acceptable
  - To assure that all collaboration is on the level, **you must always write the name(s) of your collaborators on your assignment.** Failure to adequately acknowledge your contributors is at best a lapse of professional etiquette, and at worst it is plagiarism. Plagiarism is a form of cheating.

# Academic Honesty – Gilligans Island Rule

- This rule says that you are free to meet with fellow students(s) and discuss assignments with them. Writing on a board or shared piece of paper is acceptable during the meeting; however, you may **not take any written (electronic or otherwise) record away from the meeting**. This applies when the assignment is supposed to be an individual effort. After the meeting, engage in half hour of mind-numbing activity (like watching an episode of Gilligan's Island), before starting to work on the assignment. This will assure that you are able to reconstruct what you learned from the meeting, by yourself, using your own brain.

# How to Read a Research Paper

- Typical paper
  - Abstract
  - Introduction
  - Motivation, problem description
  - Research questions that are being addressed by this paper
  - Experiment Setup
  - Results
  - Conclusions and Future work

# Why do you read a paper?

- Understand and learn new contributions

- However
  - Not all papers are “good”
  - Not all papers are “interesting”
  - Not all papers are “worthwhile” for you

- You have to learn to identify a good paper and spend your time wisely
  1. Breadth
  2. Depth
  3. React

# How to Read a Research Paper

- Ask yourself, what is this paper about? (breadth)
  - Read the title and the abstract

    If you still don't know what this paper is about, then this is a bad paper.
  - Read the conclusion

    Are you now sure you know what this paper is about? If not it is a BAD paper. We will try not to read such papers in this course
- Read the introduction
- Read the section headings
- Read tables and graphs and captions. See what they say

# How to read a paper (cont)

- See who wrote it, where it was published, when was it written (credibility)
- Skim bibliography to see if the authors are aware of relevant related work. See if you know the relevant work. See if you know a relevant work that they didn't refer

# How to read a paper - depth

- Approach with scientific skepticism
- Examine the assumptions
  - Do their results rely on any assumptions about trends in environments?
  - Are these assumptions reasonable?
    - E.g. “Lets assume that there are billions of powerful computers, connected by a high speed network, spread across the world, our system will …”
    - E.g. “Our system can enable you to run Windows 98 on a 33Mhz Intel 386 with 640K main memory”

# How to read a paper - depth

- Examine the methods:
  - Did they measure what they claim?
  - Can they explain what they observed?
    - It is easy to dump your experimental results on the paper. As a reader you want an analysis of why the system behaves a certain way, not the raw data
      - This is why your final reports will be 10 pages long
  - Did they have adequate controls
  - Were tests carried out in a standard way? Were the performance metrics standard? If not, do they explain their metrics clearly?

# How to read a paper - depth

- Examine the statistics: (there is truth, lies and then there is statistics!!)
  - Were appropriate statistical tests applied properly?
  - Did they do proper error analysis?
  - Are the results statistically significant?
    - Common mistake: “We performed our experiment once at 4 am and noticed a ten fold improvement. Thus we conclude that our system is better”
  - Be very careful with percentages
    - Method A: 0.01 seconds, our Method: 0.005 seconds
    - Our method shows 100% improvement over method A!!

# How to read a paper - depth

- Examine the conclusions:
  - Do the conclusions follow logically from the experiments
    - We performed our experiments with 8 palm pilots and saw a 10 fold improvement. Hence we conclude that our system will scale to millions of palm pilots
  - What other explanations are there for the observed effects
  - What other conclusions or correlations are there in the data that they did not point out
    - Earlier work performed experiments using a 2 Mbit wireless network. Our system (incidentally) used a 11 Mbit network and saw a 5 fold improvement. So our technique works!!

# How to read a paper - react

- Take notes
- Highlight major points
- React to the points in the paper
  - Place this work with your own experience
  - If you doubt a statement, note your objection
  - If you find a pleasing quotation, write it down
- Construct your own example
- Summarize what you read

- Maintain your own bibliography of all papers that you ever read

# Sample bibliography - bibtex

```
@Book{stevens98,
        author = {W. Richard Stevens},
        title = {UNIX Network Programming:
                Networking APIs: Sockets and XTI},
        publisher = {Prentice Hall},
        year = 1998,
        volume = 1,
        series = {ISBN 0-13-490012-X},
        note = {Sample code from this book is available at
                \url{http://www.kohala.com/start/unpv12e.html}},
        edition = 2,
}
```

….. You can refer to the Computer Network books by W. Richard Stevens \cite{stevens98} for sample ….

# How to Write a Research Paper

- Write it such that anyone who reads it using the method we discussed understands your ideas.
- Clearly explain what problem you are solving, why it is interesting and how your solution solves this interesting problem
- Be crisp. Explain what your contributions are, what your ideas are and what are others' ideas

# First paper – experiences

- **Hints for Computer System Design** - Butler W. Lampson
  - http://en.wikipedia.org/wiki/Butler_Lampson
  - 1992 – Turing Award, 1994 – ACM Fellow
  - Xerox PARC -> DEC SRC -> MSR Research

# Functionality

- Keep it simple (Keep it simple, Stupid (KISS)), (If in doubt, leave it out), (Exterminate features)
  - Do one thing at a time and do it well. Don't generalize; generalizations are generally wrong.
- Don't hide power
- Leave it to the client
  - End to end argument in networks
- Keep basic interfaces stable
- Plan to throw one away (Multics -> UNIX)
- Keep secrets of the implementation
- Divide and conquer (UNIX style system programs)

# Functionality (cont)

- Use a good idea again instead of generalizing it
- Cache answers to expensive computations
  - Makes much more sense where resources are plentiful
  - 4 GB main memory in laptops!!
- When in doubt, use brute force
  - There is no point optimizing for a 2000 computer in 2008
- Fault tolerance: End-to-end argument