COMP 206
Introduction to Software Systems

Lecture 1 – Course Overview

Sept 5, 2018

What is this course about?

- Comprehend the amazing systems that enable our world
- Practice with the tools used to program these systems
- Case studies of common systems programs

Introducing COMP 206 Team

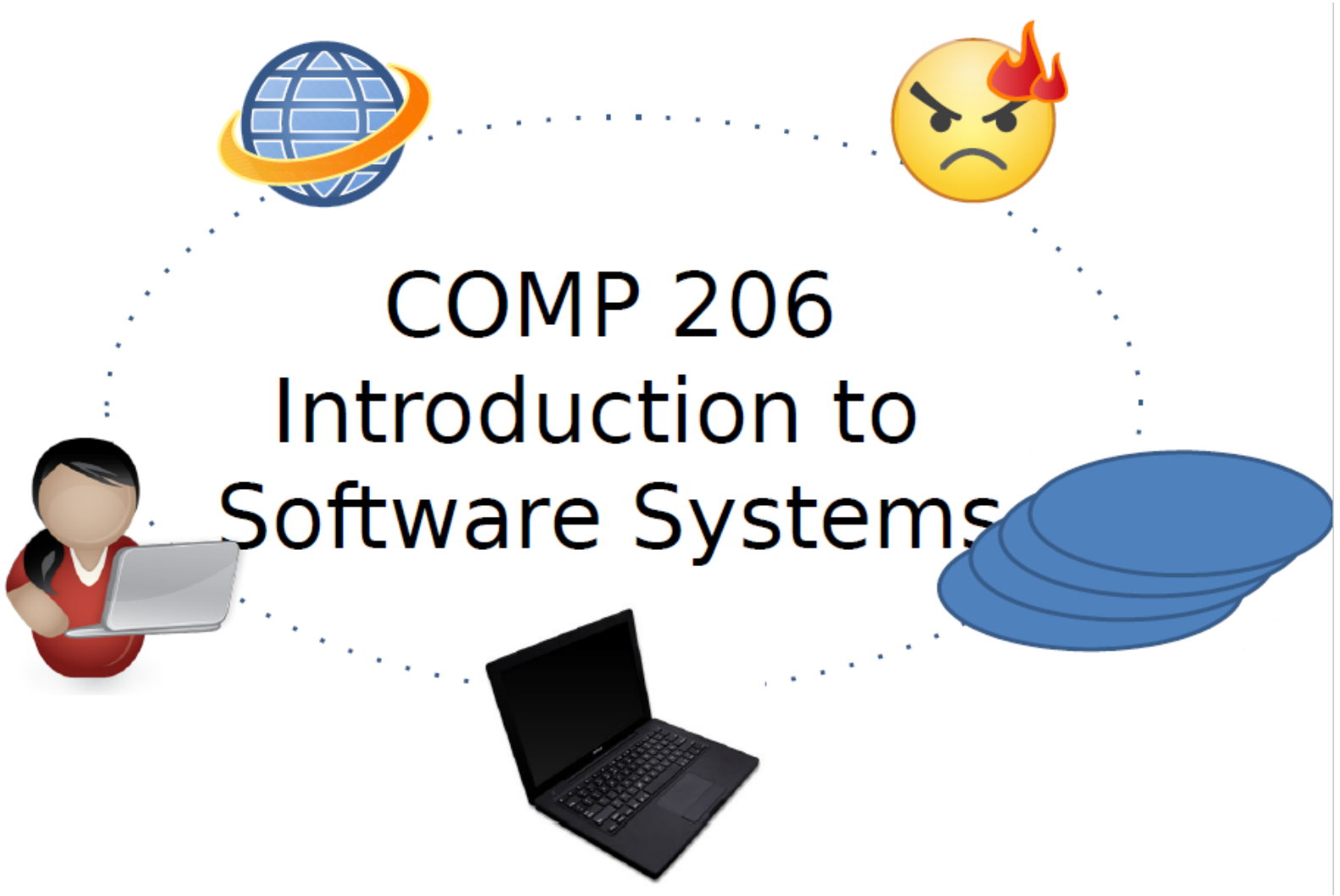
- I’m David Meger (Dave)
  - I run the Mobile Robotics Lab, where I care about making intelligent robots that can care for the elderly, swim, fly, drive and play the xylophone
  - Outside of academia, I previously worked at a start-up who’s goal was to prevent moths from breeding (true story!)
- 15 Graduate Student TAs
- TEAM tutors from previous versions of the course

Software System definition

- (for the purposes of this course) A Software System is a collection of ***specifications*** agreed upon and ***coded*** by several programmers such that multiple elements ***work together effectively***.

- We'll specifically study the following interactions between elements:
  - A user program with the operating system
  - A program with its human user
  - Two programs with a common set of data
  - Two programs over the Internet or World Wide Web

The Big Picture



Software Systems in the News

HOME SEARCH

The New York Times

SUBSCRIBE NOW

TECHNOLOGY

Researchers Discover Two Major Flaws in the World's Computers

查看简体中文版 | 查看繁體中文版 | Leer en español

By CADE METZ and NICOLE PERLROTH JAN. 3, 2018

208

RECENT COMMENTS

Cathy January 4, 2018
Well if for the most part you look at what in memory 99 percent will be useless to anybody .

NML January 4, 2018
Looks like my faith in Pilot, Bic, Clairefontaine & Moleskine has been rewarded.

dunbar7376 January 4, 2018
I have found a third major flaw: not one woman in your RSA conference pic.

SEE ALL COMMENTS

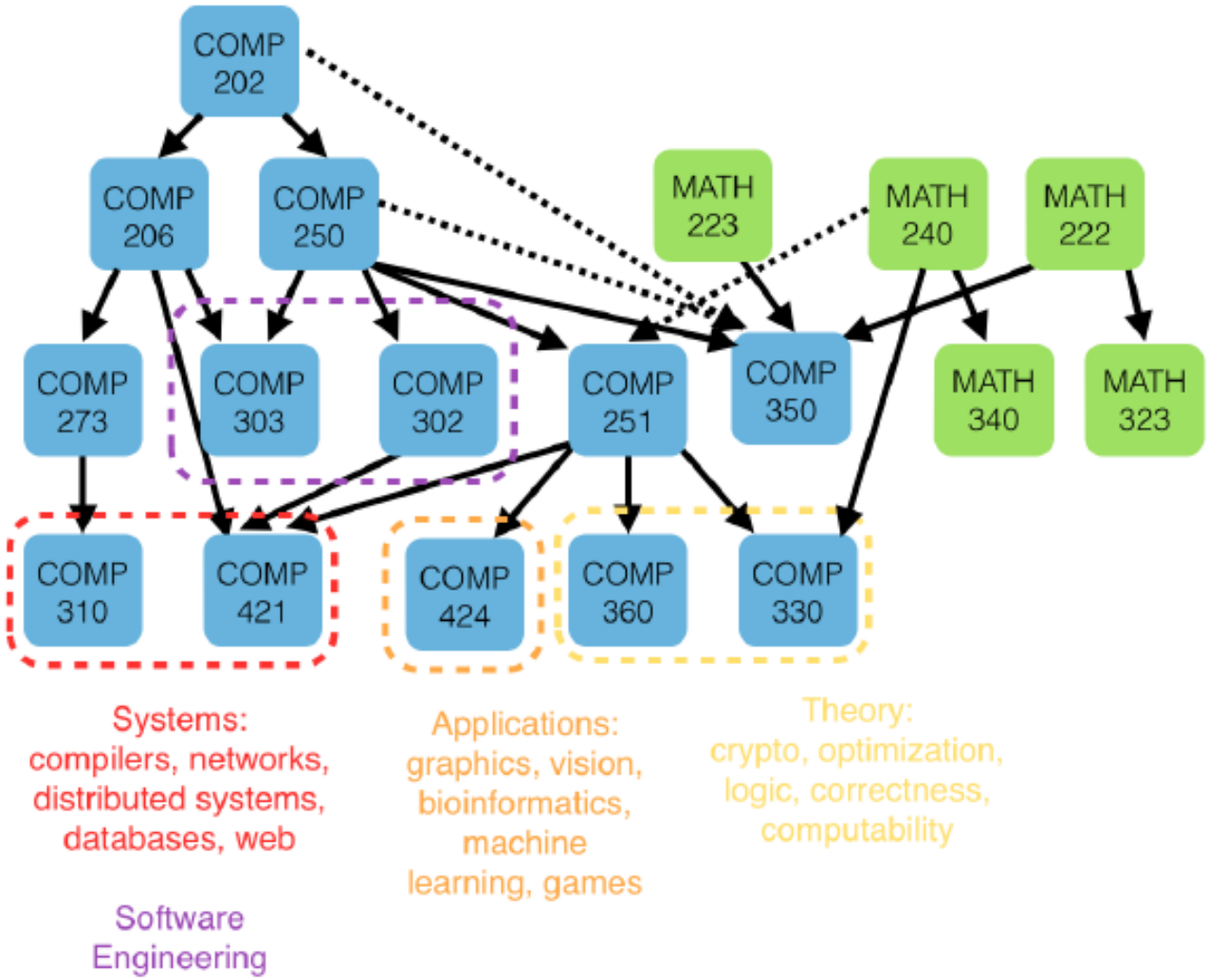
What is this course about?

- Comprehend the amazing systems that enables our world
- Practice with the tools used to program these systems:
  - The C programming language
  - The Linux operating system
  - C programs using Linux system calls
  - Compilers, debuggers, editors, scripting languages, version control and build tools that make it possible to build larger systems
- Case studies of common systems programs:
  - User interaction
  - Data interaction
  - Networking and web

History of Unix and Linux

1960s	MULTICS project (MIT, Bell Labs, GE, AT&T)
1970s	AT&T Bell Labs
1970s/80s	UC Berkeley
1980s	DOS imitated many Unix ideas
	Commercial Unix fragmentation
	GNU Project
1990s	Linux
now	Unix, Linux and derivatives (including Android and OS X) dominate worldwide use

COMP Course Context



Who is this course for?

- Pre-requisite: COMP 202 or equivalent prior programming experience
  - You should have written multiple programs of reasonable size
  - The language doesn't matter, any of Python, Java, C, C++, Pascal, Go, Ruby, Lisp, COBOL, Fortran will do (and many others!)
  - You should understand variables and datatypes
  - You should be familiar with control flow (if/else, for, while)
  - Nice if you've seen recursion, but OK if not
  - Nice if you've seen object oriented (classes) but OK if not
- If you aren't sure, let's talk in office hours
- Beyond this, the course is open to everyone, and will be taught in an accessible way. My goal is to transfer you this important knowledge!

Course Outline

- External on web

Tips for Success

- Use my exercises and readings. They aren't for marks, but I try to make them very targeted to the assignments, so you'll have a head start.
- Don't underestimate the programming assignments:
  - They are meant to be doable, but they can take time
  - C makes almost everyone cry… but it ages well. If you are here at McGill, you are by far smart enough to do well. Perseverance is key!
- Visit the TAs and I early and often during office hours
  - Tell me your name (perhaps a few times, sorry this class is huge)

Exercises

- Access Ubuntu 16.04. Several options:
  - Go to Trottier 3rd floor. Create an account, sit in front of a machine directly
  - Install Linux at home the safe way
  - Use your home computer to connect to SOCS servers
  - Install Linux at home the scary way
- Try out the tutorial commands described here:
  - https://www.cs.mcgill.ca/docs/ (in particular the tutorials under “UNIX”)

Installing Linux at Home – The Safe Ways

- Run Linux as a Virtual Machine (Windows or OSX)
  - Some limits on performance, but all features available
  - http://www.osboxes.org/ubuntu/
- On Windows 10, enable the “Windows Subsystem for Linux”
  - Perfect performance of terminal interface, but awkward to start graphical programs
  - Install instructions
- On OSX, get familiar with the terminal
  - Perfect performance, but there are some subtle differences. Can you find any?

Remote Login to SOCS Linux

- SSH stands for “Secure Shell”. It allows access from remote users over the network and behaves almost as though you were there in person
- There are even ways to “forward” graphical applications

- Decent instructions here: https://www.cs.mcgill.ca/docs/labs/ssh
- Let’s try it live!

Living Life on the Edge

- Linux can be run as a native operating system that dual boots “side-by-side” with any other Operating System

WARNING: Installing in this way gives you the absolute best functionality but has the potential to damage or destroy your existing data and operating system. Do not start this without backing up all important files.

- If you feel brave, proceed very carefully search the web for a guide suited to your specific machine. Please read a few different pages before you do anything and ensure you pay very close attention to detail.
- We’ll check in next Tuesday – prizes for anyone who can share an interesting story!