OS: Retrospective and Prospective

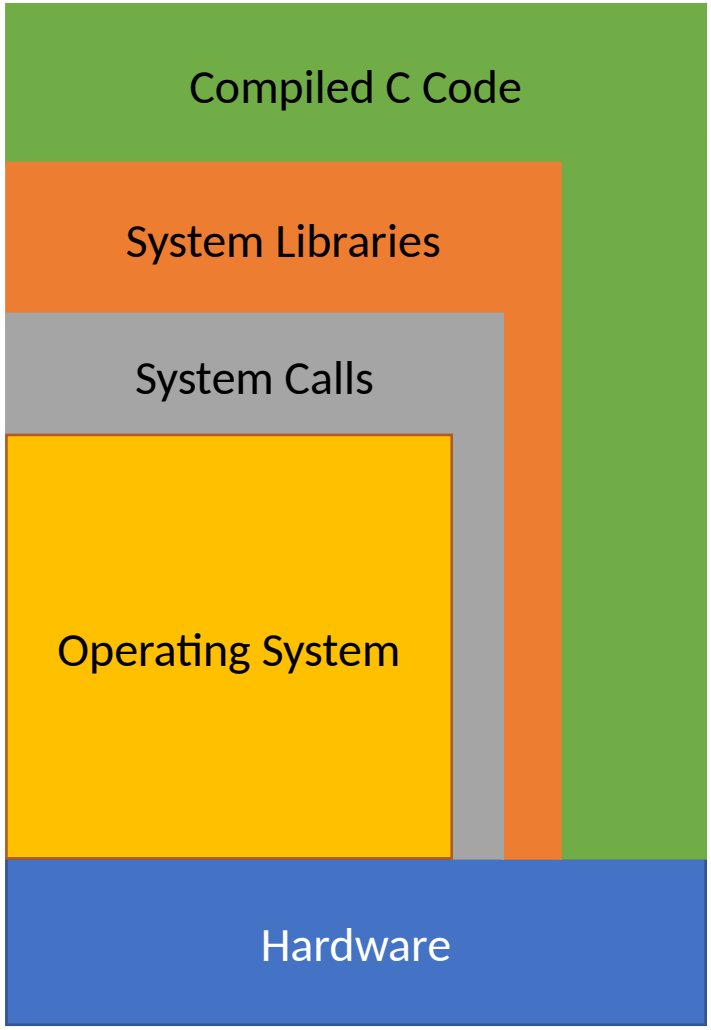
THE UNIVERSITY OF NEW SOUTH WALES

Recap and Consider

- Revision of past content.
- Thoughts about how things are going.
- Options for the future.

Rewind: Why Study OS?

- Understand the whole software stack
- Develop OS code
- Develop code in a challenging environment
  - Concurrency issues
  - Security issues
- Application performance
  - Understand operating system behaviour and how best to interface with it.
  - Diagnose system performance issues.



A System Call

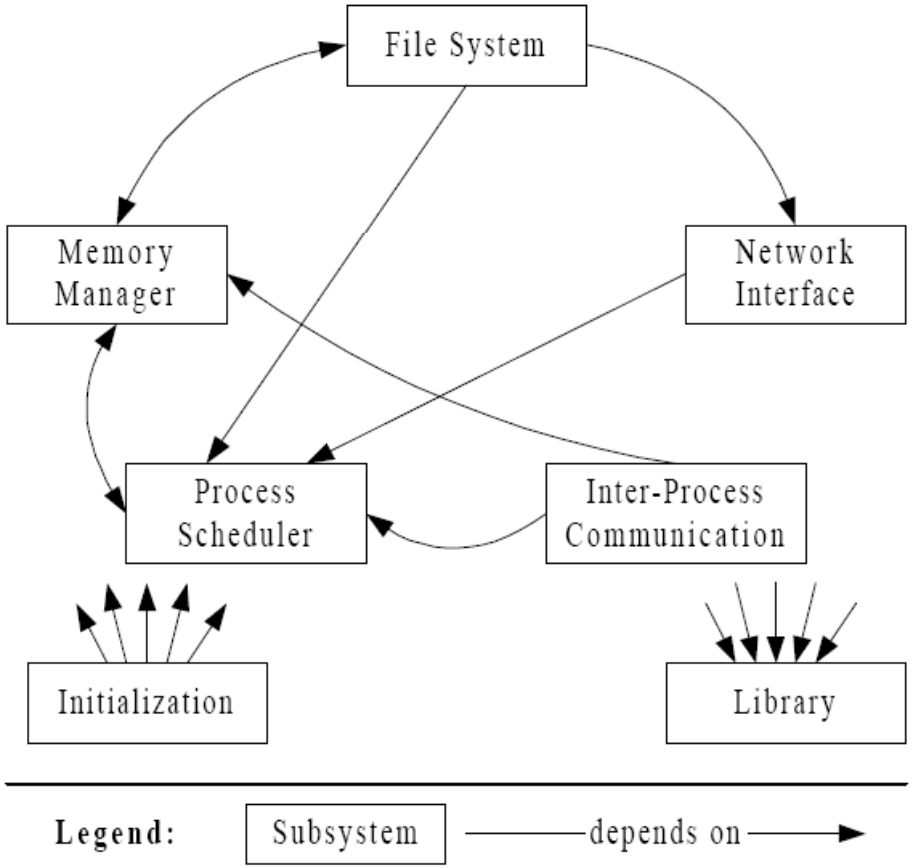
```
void log_special_event(int x, int y) {
  FILE *f;
  int log_count;

  get_log_state(&f, &log_count);
  fprintf(f, "%d: SPEC(%d, %d)\n", log_count, x, y);
}
```

- Nearly all programmers have some vague intuition about what `write()` and `fprintf()` do.
- But what does this really do?

Structure of a Monolithic OS

- Simple standard hierarchical structure doesn't really exist.
- However, some reasonable structure usually prevails



Bowman, I. T., Holt, R. C., and Brewster, N. V. 1999. Linux as a case study: its extracted software architecture. In *Proceedings of the 21st international Conference on Software Engineering* (Los Angeles, California, United States, May 16 - 22, 1999). ICSE '99. ACM, New York, NY, 555-563. DOI= http://doi.acm.org/10.1145/302405.302691

Becoming More Concrete

- Some of these concepts seemed pretty vague on our first look at them

- Hopefully you now have a pretty good idea how a standard modern OS like Linux or Windows or Mac OS X fits together.

OS Concepts

	Seen	?
Processes and Threads	Yes	Yes
Concurrency and Deadlock	Yes	Yes
Syscalls	Yes	Yes
File Systems	Yes	Yes
Memory Management	Yes	Yes
Virtual Memory	Yes	Yes
Virtualisation	Yes	No
Multiprocessors and Locking	Yes	Yes
I/O	Yes	No
Scheduling	Yes	No

What Next?

- Enjoying the OS course?
- COMP9242 Advanced OS.
  - Lots more implementation, using seL4, a modern, unusual, locally-developed OS micro-kernel.
  - More on novel and unusual OS approaches, loosely in the style of the extended lectures.

What Next?

- Enjoying the OS course?

- https://www.trustworthy.systems/
  - Local research group interested in OS, programming languages and provable security.
  - Opportunities for honours theses, Taste of Research scholarship/internships etc.

What Next?

- https://www.trustworthy.systems/

What Next: Exam

- There are lots of details about the exam.
- There is an exam **hurdle** in this course.
- Written-question style exam provided using the Inspera + SEB infrastructure.
  - Mix of multi-choice and written questions.
  - We may ask about C/MIPS code.
  - Typed on a computer, but no actual programming in the exam.
- Past pen-and-paper exams on wiki.
  - See link on course website.

What Next: Exam Content

- Past exams on wiki are a rough guide to content.
  - See link on course website.
  - Note that old exams assess some content that is now not in the course or not assessable.
- What topics and questions are likely to appear?
  - Lecture content is relevant
  - Tutorials questions are a guide.
  - Assignment content is a guide.

What Next: Exam Authority

- The exam is on Fri the 21st for *most* students.
- UNSW Exams authority is organising and invigilating.
  - Different to last year.
  - If they've told you a different date, they're right.
- Limited reference material available in the exam.
  - Students may bring in one A4 page of notes.
  - Must be hand-written, may be double sided.
  - Different rules to last year.

What Next: Exam System

- Exam requires Inspera + SEB.
- We have uploaded two Inspera practice exams, one of which activates SEB. Access via moodle.
- Exam expects you to BYOD.
  - You can bring a recent Windows, Mac OS, or Chromebook device. Search Inspera+SEB.
  - Laptops can be borrowed from the library and CSE foyer. Test-run this before exam day.
  - If technology fails, students will get a supplementary exam.

What Next: Exam Rooms

- Note that there are 4 Inspera exams:
  - COMP3231, COMP3891, COMP9201, COMP9283.
- Inspera process requires you to join an exam and enter a code displayed in the exam room.
  - Make sure you're joining the right exam!
  - Make sure you're using the right code!
  - Make sure you're in the right room!

OS Concepts

	Seen	Exam
Processes and Threads	Yes	Yes
Concurrency and Deadlock	Yes	Yes
Syscalls	Yes	Yes
File Systems	Yes	Yes
Memory Management	Yes	Yes
Virtual Memory	Yes	Yes
Virtualisation	Yes	No
Multiprocessors and Locking	Yes	Yes
I/O	Yes	No
Scheduling	Yes	No

Review: My Perspective

- I (Thomas) am running the OS course for the second time this year.
  - My research work gives me some perspective on OS but it is patchy.
  - I'm a bit more confident this year.
  - The cohort is a bit larger this year.
- Also I took the course long long ago.
- Also I have a long debrief document written by the former lecturer.

Meets Expectations

- Lecturing is, as expected, the challenging and rewarding part of running a course.
  - It helps to have a great audience!
  - Let me encourage you to turn up when you can in your continuing studies.
- Marking and mark admin is, as expected, the least fun part.

Probably Meets Expectations

- OS/161 really helps make the course concrete and substantial.

- Explanations that proceed from examples inside OS/161 are generally pretty engaging.
- Explanations from slide examples, animations etc are harder to make work.
- Using “real world” OSs would be clunky.

(Still) Below Expectations

- The “give” system and bundles.
  - There’s a lot we could say about “give”.
  - On balance it has helped in some courses we have run.
  - This year we had a lot of confusion between give/gitlab/branches.

- There isn’t an obvious sensible alternative.

Specifically Below Expectations

- This year we tried to smooth out submission of bundles with a helper script.
  - Trying to address some last-minute student panic and sadness that happened last year.
  - Probably helped the majority of students but also helped confuse a minority.
- This year's batch of "creative" asst1 submissions.
- This year's "doc.txt" documentation format.

Above Expectations

- Students have, in general, done a great job of figuring out the assignments and OS/161 internals.
- This is a big course, but the workload of answering questions on the forum hasn't been especially challenging to manage.

Unsure about Expectations

- Having figured out most of the material in my head, some of the lecture content ran so smoothly we were ahead of time.

- There is even more bonus/extra content than last year. We're keen to hear what you think of it.

myExperience

- That's my perspective; we want to hear yours.
- There's still plenty of time to leave a myExperience review.
  - Please vote even if things are generally OK.
  - Specific (potentially actionable) recommendations are the best feedback.

Extra Content Vote

- That's my perspective; we want to hear yours.
- The Inspera test exam asks you to evaluate some of the course topics, especially the new/bonus content.
  - It's OK to guess.
  - We want to hear what students want to hear.

OS Concepts

	Seen	Exam
Processes and Threads	Yes	Yes
Concurrency and Deadlock	Yes	Yes
Syscalls	Yes	Yes
File Systems	Yes	Yes
Memory Management	Yes	Yes
Virtual Memory	Yes	Yes
Virtualisation	Yes	No
Multiprocessors and Locking	Yes	Yes
I/O	Yes	No
Scheduling	Yes	No

Processes and Threads

Learning Outcomes:

- An understanding of fundamental concepts of processes and threads
- A basic understanding of the MIPS R3000 assembly and compiler generated code.
- An understanding of the typical implementation strategies of processes and threads
  - Including an appreciation of the trade-offs between the implementation approaches
    - Kernel-threads versus user-level threads
- A detailed understanding of “context switching”

Concurrency and Deadlock

Learning Outcomes:

- Understand the issue of concurrency in operating systems and multithreaded applications
- Know the concept of a critical region.
- Understand how mutual exclusion of critical regions can be used to solve concurrency issues
  - Also how mutual exclusion is implemented
- Identify a producer consumer bounded buffer problem.

Concurrency and Deadlock

Also:

- Understand what deadlock is and how it can occur when giving mutually exclusive access to multiple resources.
- Understand broadly some approaches to mitigating the issue of deadlock.
  - Deadlock prevention, deadlock detection.
  - Note: we didn't go into detail about deadlock avoidance (e.g. the banker's algorithm) in lectures & tutorials and won't assess it.

Syscalls

- A high-level understanding of *System Call* interface
  - Mostly from the user's perspective
    - From textbook (section 1.6)
- Understanding of how the application-kernel boundary is crossed with system calls in general
    - Including an appreciation of the relationship between a case study (OS/161 system call handling) and the general case.
- Exposure to architectural details of the MIPS R3000
  - Detailed understanding of the of exception handling mechanism
    - From "Hardware Guide" on class web site

File Systems

- Files and directories from the programmer (and user) perspective

- File allocation methods
  - How files are stored in disk blocks, and what book keeping is required.
- Layout on drive
- Managing free space
- Directories
- Block size trade off
  - Internal vs External fragmentation

31

Memory Management

- Appreciate the need for memory management in operating systems, understand the limits of fixed memory allocation schemes.
- Understand fragmentation in dynamic memory allocation, and understand basic dynamic allocation approaches.
- Understand how program memory addresses relate to physical memory addresses, memory management in base-limit machines, and swapping

Virtual Memory Management

- An understanding of MMU and TLB systems
- Understand TLB refill:
  - in general,
  - and as implemented on the R3000
- An understanding of demand-paged virtual memory in depth, including:
  - Locality and working sets
  - Page replacement algorithms
  - Thrashing

33

Multiprocessors

- An understanding of the structure and limits of multiprocessor hardware.
- Broad understanding of operating system support for multiprocessor machines.
- An understanding of issues surrounding and approaches to construction of multiprocessor synchronisation primitives.
  - For instance test-and-set, test-and-test-and set locks, and the performance differences.

34

Today (Recap Recap)

- We've now covered most of the internal systems of a modern standard OS.
- We've seen my retrospective, and welcome your input (myExperience) and votes on future course content.
- Logistics of the exam.
- Recap of the examinable course content.

- Thanks for your attention!