OS: Retrospective and Prospective

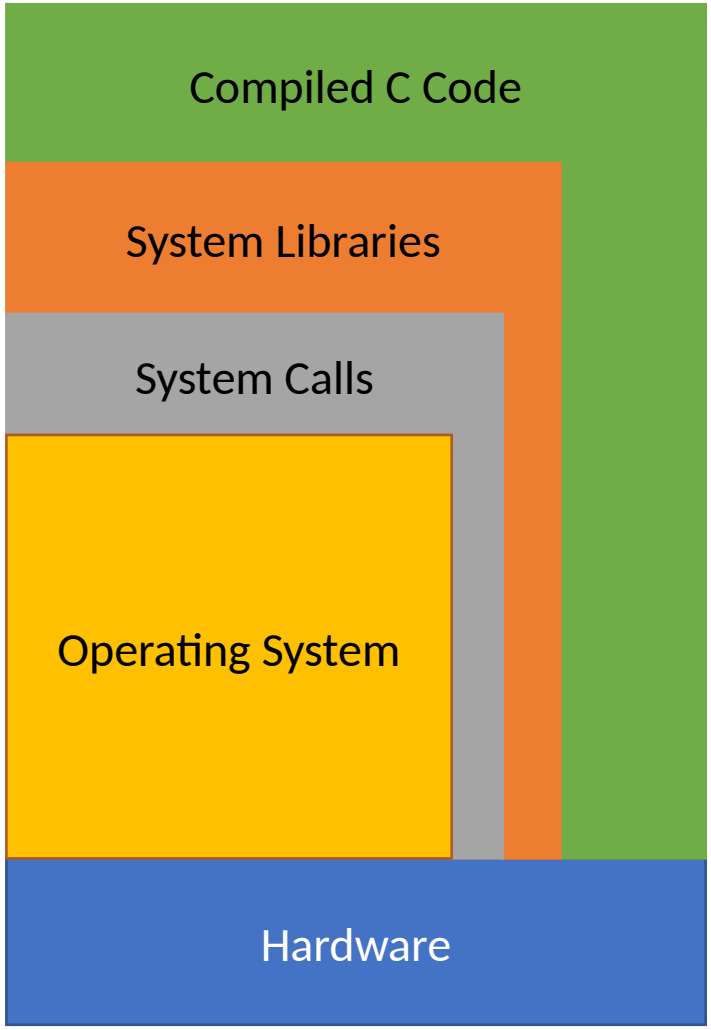
THE UNIVERSITY OF NEW SOUTH WALES

Recap and Consider

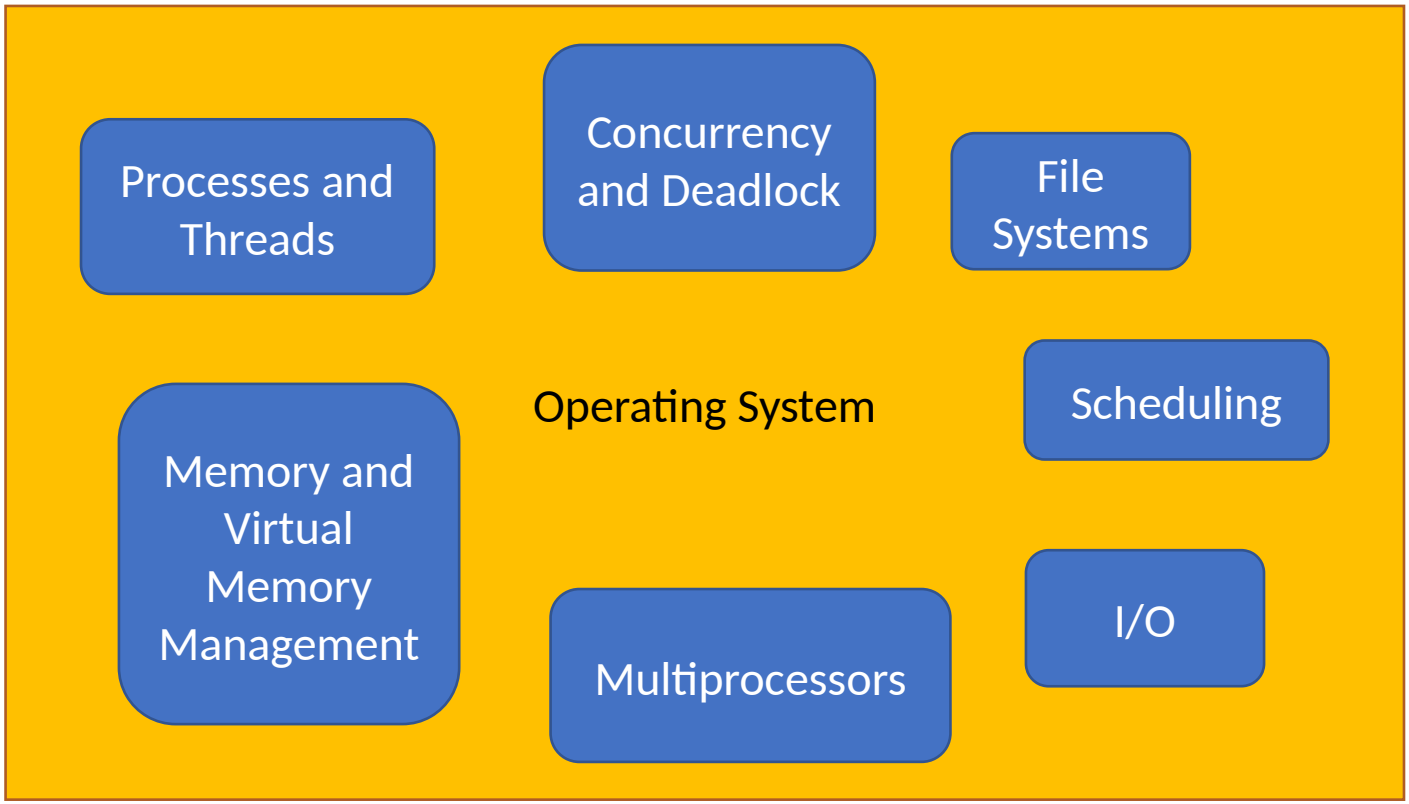
- Revision of past content.
- Thoughts about how things are going.
- Options for the future.

Rewind: Why Study OS?

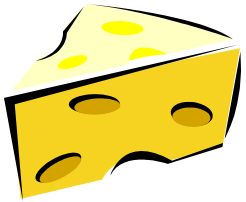
- Understand the whole software stack
- Develop OS code
- Develop code in a challenging environment
  - Concurrency issues
  - Security issues
- Application performance
  - Understand operating system behaviour and how best to interface with it.
  - Diagnose system performance issues.



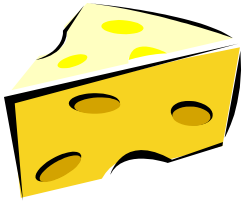
Major OS Topics



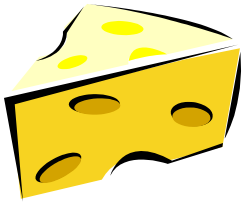
Disk



Memory

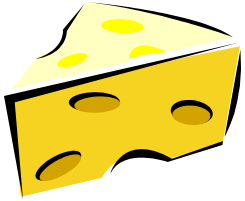


CPU



Network

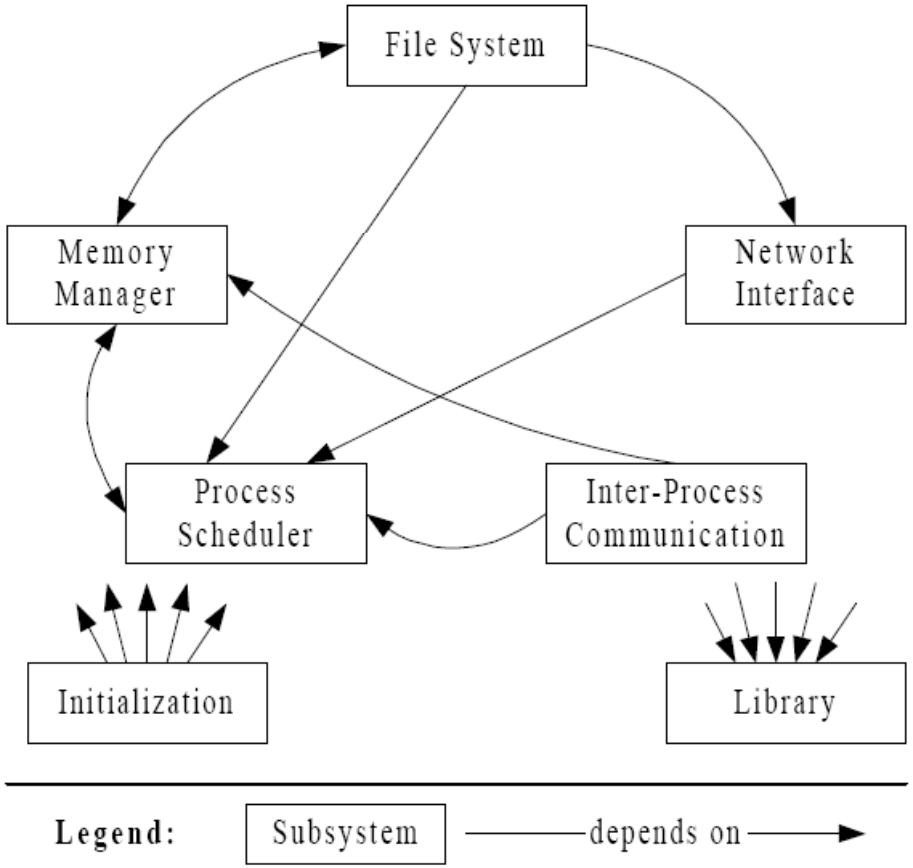
Bandwidth



Users

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

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

What Next?

- Enjoying the OS course?
- COMP9242 Advanced OS.
  - Lots more implementation, using seL4, a modern, unsual, locally-developed OS kernel.
  - More on novel and unusual OS approaches, 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: Exam

- Lots of questions about the exam details.
- Thanks to CSE Exams crew:
  - Actual exam Wed 27th, 9:45-12:00
  - In lecture theatres, mostly Patricia O'Shane
  - TODO: other rooms & times for ELPs
  - BYO device if you can, or contact us
  - TODO: detailed pre-exam instructions
- Students can bring paper/printed notes

Review: My Perspective

- I (Thomas) am running OS for the first time this year.
  - My research work gives me some perspective on OS but it is patchy.
- 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!
- Marking and mark admin is, as expected, the least fun part.

Probably Meets Expectations

- OS/161 is basically the star of the show.

- Explanations that proceed from examples inside OS/161 are generally pretty engaging.
- Explanations from slides, animations etc tend to be dry or insubstantial.

Below Expectations

- The “give” system and bundles.
  - There’s a lot we could say about “give”.
  - On balance it has helped in the past.
  - It really doesn’t seem to combine well with optional components.
    - The advanced parts.

Way Above Expectations

- The capacity students have shown to figure out OS/161 internals.

- For instance, the forum is actually getting easier to manage as time goes on.

myExperience

- That's my perspective; we want to hear yours.
- There's still plenty of time to leave a myExperience review.

OS Concepts

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

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

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.

Today (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).
- Recap of the rest of the course content.

- Thanks for your attention!