Lecture 1 —Introduction, Operating Systems, Security

Jeff Zarnett
jzarnett@uwaterloo.ca

Department of Electrical and Computer Engineering
University of Waterloo

December 6, 2022

Course Syllabus

As our first order of business, let's go over the course syllabus.

Collaborative Course

The source material for the SE 350 notes and slides is now open-sourced via Github.

If you find an error in the notes/slides, or have an improvement, go to `https://github.com/jzarnett/se350` and open an issue.

If you know how to use `git` and LaTeX, then you can go to the URL and submit a pull request (changes) for me to look at and incorporate!

Introduction to Operating Systems

Operating systems are those programs that interface the machine with the applications programs. The main function of these systems is to dynamically allocate the shared system resources to the executing programs.

- What Can Be Automated?: The Computer Science and Engineering Research Study, MIT Press, 1980

Introduction to Operating Systems

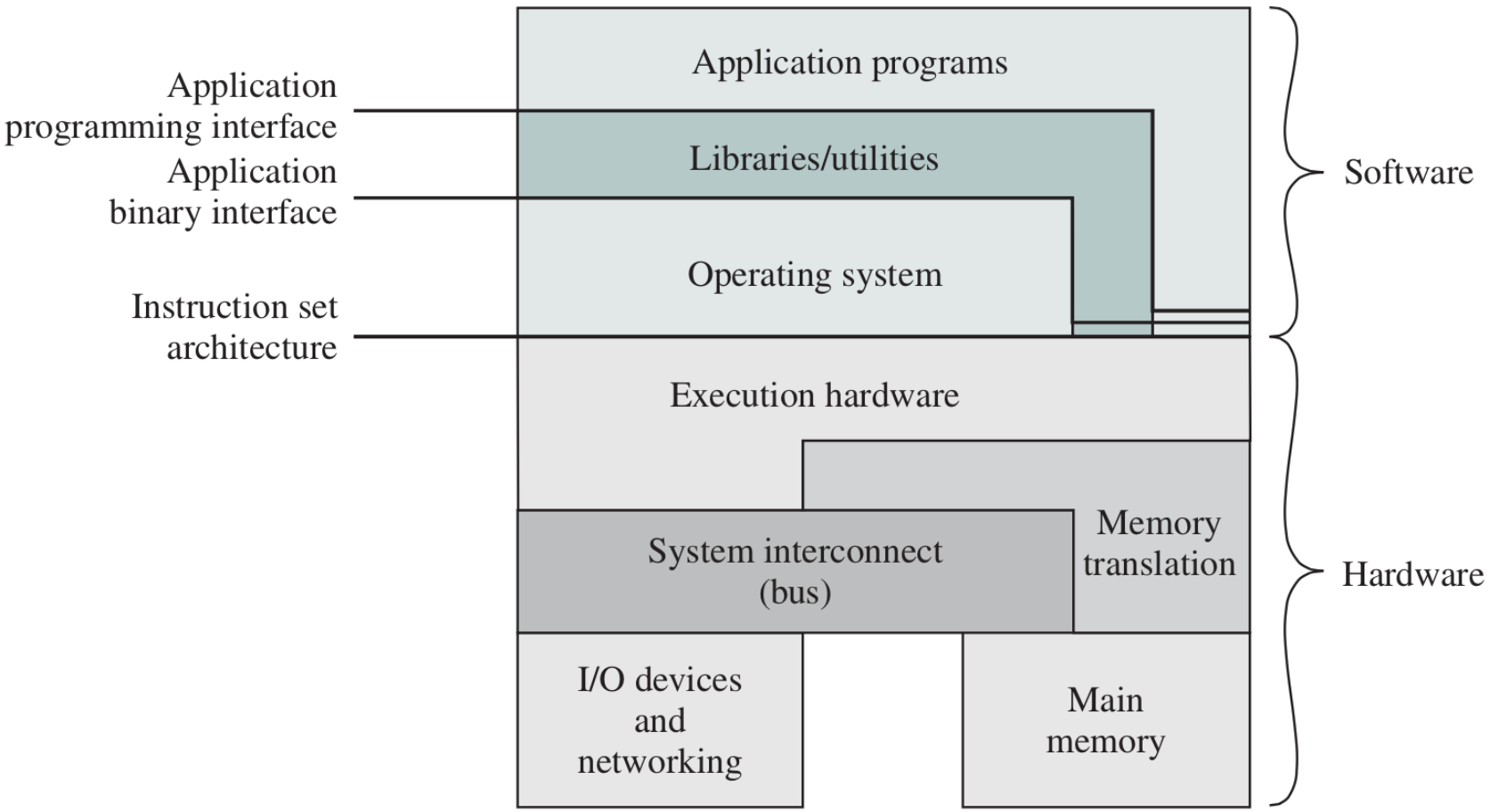
An operating system (OS) sits between the hardware and programs.

It does many different things.

It has many often-conflicting goals.

You might think of the OS as the “secretary” of the system.

Structural Diagram of a Modern Computer



OS: Resource Manager

The OS is responsible for resource management and allocation.

Resources like CPU time or memory space are limited.

The OS must decide how to allocate & to keep track of system resources.

In the event of conflicting requests, choose the winner.

OS: Environment Provider

The OS enables useful programs like Photoshop or Microsoft Word to run.

The OS is responsible for abstracting away the details of hardware.

This is so program authors do not have to worry about the specifics.

Imagine Hello World had to be written differently for different hardware.

OS: Multitasking

Multiple programs means some resources are shared.

$\rightarrow$ A source of conflicts!

OS creates and enforces the rules so all can get along.

Sometimes processes want to co-operate and not compete.

The OS can help them to do so.

OS: Efficiency

Another goal may be to use the computer hardware efficiently.

Any moment when the supercomputer is not doing useful work is a waste.

There are always programs eager to run.

OS: What is it, really?

Operating systems tend to be large and do a lot of things.

We expect now that an OS comes with a web browser, an e-mail client, some method for editing text, et cetera.

The part of the operating system we will study is the Kernel.

The kernel is the “core”; the portion of the OS that is always present in main memory and the central part that makes it all work.

OS: Evolution

Operating systems will evolve over time.

There will be new hardware released, new types of hardware, new services added, and bug fixes.

Evolution is constrained: a need to maintain compatibility for programs.

Microsoft Windows: strict devotion to not breaking binary compatibility.

Systems Programming

Operating systems themselves are not the whole picture.

Systems programming: the next layer above the OS itself.

Useful tools, but they are not part of the kernel.

Some examples of systems programming:

- **File Manipulation**
- **Status Information**
- **Programming-Language Support**
- **Communication**

Systems Programming

Programming at this level is more difficult than regular programs.

It may require knowledge of the hardware, or perhaps programming facilities like debugging are limited.

Systems programs must take concurrency into account.