CSE 323: Operating System Design

Introduction

Salman Shamil

North South University (NSU)

Fall 2025

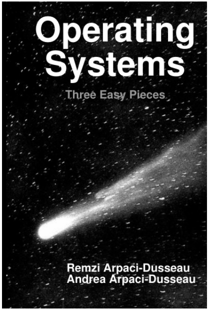
Original slides by Mathias Payer and Sanidhya Kashyap [EPFL]

Lecture Topics

- What you will learn in this course (and how)
- What an OS is and why you want one
- Why you should know about OSes

Class Materials

- **Slides** and **Lectures** on OS Design
- Textbook: Operating Systems: Three Easy Pieces
- Other Books:
  - Operating System Concepts, Silberschatz, Galvin & Gagne, 10th ed. (Wiley, 2019)
  - Modern Operating Systems, Tanenbaum & Bos, 4th ed. (Pearson, 2015)
- Coding Examples & Practice Problems



Grading & Office Hours

- **Assessment Methods**

Assessment Item	**Weight (%)**
Attendance	5
Assignment	5
Class Test	20
Mid Term	30
Final Exam	40

Grading & Office Hours

- **Assessment Methods**

Assessment Item	Weight (%)
Attendance	5
Assignment	5
Class Test	20
Mid Term	30
Final Exam	40

- **Counseling Hours**

Day	Time
Saturday & Tuesday	10:00 AM – 11:00 AM
Sunday & Wednesday	11:00 AM – 01:30 PM

- **Contact**: Office: **719(A)** Email: salman@cse.uiu.ac.bd

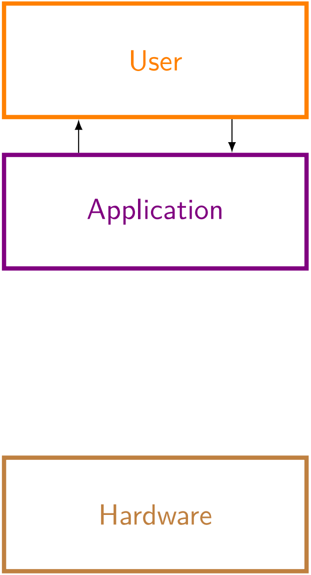
What is an Operating System?

What is an Operating System?

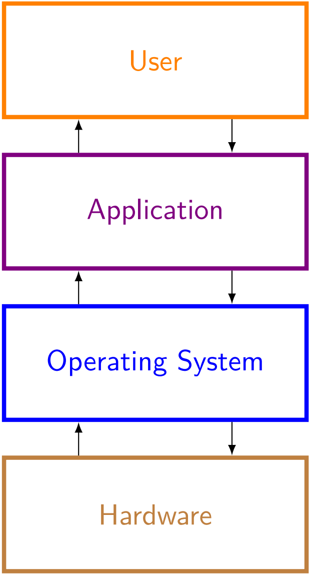
User

Hardware

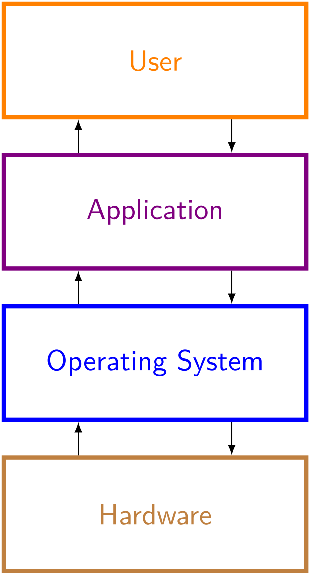
What is an Operating System?



What is an Operating System?



What is an Operating System?



OS is middleware between applications and hardware.

- Provides standardized interface to resources
- Manages hardware
- Orchestrates currently executing processes
- Responds to resource access requests
- Handles access control

OS role #1: Standardized Interface

- Provides **common functionality** to access resources.
- Abstracts hardware, provides a **unified interface**.
    - Example: Network chips A and B are accessed using the same network API that allows sending and receiving packets.
- **Virtualization / Abstraction** of physical resources.

OS role #1: Standardized Interface

- Provides **common functionality** to access resources.
- Abstracts hardware, provides a **unified interface**.
  - Example: Network chips A and B are accessed using the same network API that allows sending and receiving packets.
- **Virtualization / Abstraction** of physical resources.
- **Challenges:**
  - Defining the correct abstractions (e.g., what level)
  - What hardware aspects should be exposed and how much

OS role #2: Resource Management

The OS shares (limited) resources between applications.

OS role #2: Resource Management

The OS shares (limited) resources between applications.

- Isolation: protect applications from each other
- Scheduling: provide efficient and fair access to resources
- Limit: share access to resources

OS Role Analogy

The OS is like a waiter that serves individual clients. The waiter knows the menu, records orders, and delivers food to the right table while keeping track of the bill.

Figure 1: OS as a waiter for processes

What management services does an OS provide?

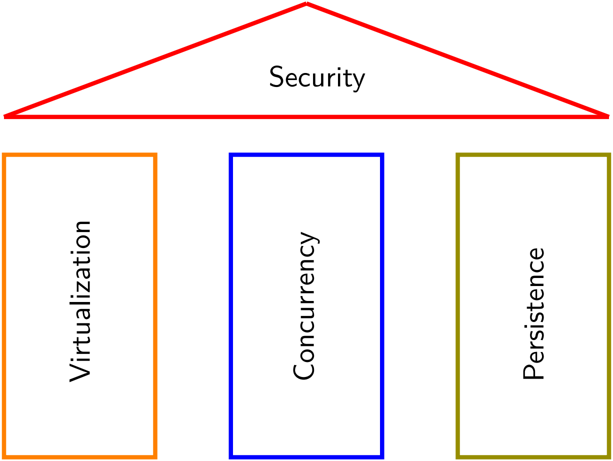
- ***CPU:*** initializes program counter/registers, shares CPU
- ***Program memory:*** initializes process address space, loads program (code, data, heap, stack)
- ***Devices:*** read/write from/to disk; device driver is hardware specific, abstracts to common interface

(Short) History of Operating Systems

- Started as a convenience library of common functions
- Evolved from procedure calls to system calls
- OS code executes at higher privilege level
- Moved from single process to concurrently executing processes

OS Building Blocks

OS design nicely separates into three pillars, with security as a transcendental layer covering/overarching all pillars.



Building block: Virtualization

Each application believes it has all resources for itself

- ***CPU:*** unlimited amount of instructions, continuous execution
- ***Memory:*** unlimited memory is available
- ***Challenge:*** how to share constrained resources

Building block: Concurrency

OS must handle ***concurrent events*** and untangle them as necessary.

- Hide concurrency from ***independent*** processes
- Manage concurrency from ***dependent*** processes by providing synchronization and communication primitives
- ***Challenge:*** providing the right primitives

Building block: Persistence

Lifetime of information is greater than lifetime of a process.

- Enable processes to access ***non-volatile information***
- Abstract how data is stored (through a file system)
- Be ***resilient to failures*** (e.g., power loss)
- Provide ***access control***
- ***Challenge:*** authentication and permissions

Building block: Security

OS is a gatekeeper, it ensures and enforces security. OS is also privileged and therefore frequently attacked.

- ***Isolate*** processes from each other and the OS
- ***Authenticate*** users (who is allowed to do what)
- Protect itself against malicious network/user input
- Harden program execution (through mitigations)
- ***Challenge:*** performance versus security

Why you should study OS!

- Build, modify, or administer an operating system.
- Understand design decisions
- Understand system performance
- Enables understanding of complex systems
- Turns you into a better (systems) programmer