



**KOÇ  
UNIVERSITY**

# COMP304

## Operating Systems (OS)

### Introduction

Hakan Ayrar

Lecture 1

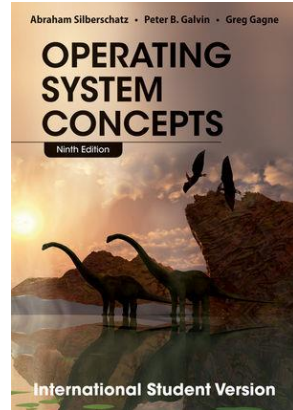
# Course Basics

- Website

- Blackboard : <https://ku.blackboard.com/>
- All course materials will be posted

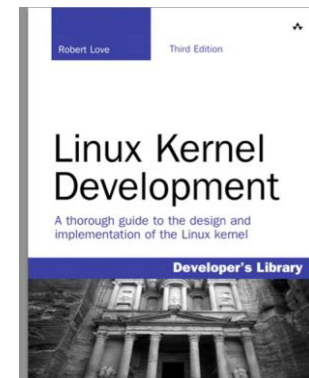
- *Main Book*

- *Operating System and Concepts (10<sup>th</sup> edition)*
  - *By Silberschatz, Galvin and Gagne*



- *Additional Book*

- *Linux Kernel Development (3<sup>rd</sup> Edition)*
  - *By Robert Love*
  - <http://it-ebooks.info/book/819/>



# Linux Operating System

- In all projects, we will be using Linux environment.  
You have two options:

**BACK UP YOUR DATA**

## 1) Install a Linux OS environment (recommended)

Installation package (latest distributions of Ubuntu or Fedora)

Install as dual boot on your own computer

## 2) Install Linux Virtual Machine on your computer

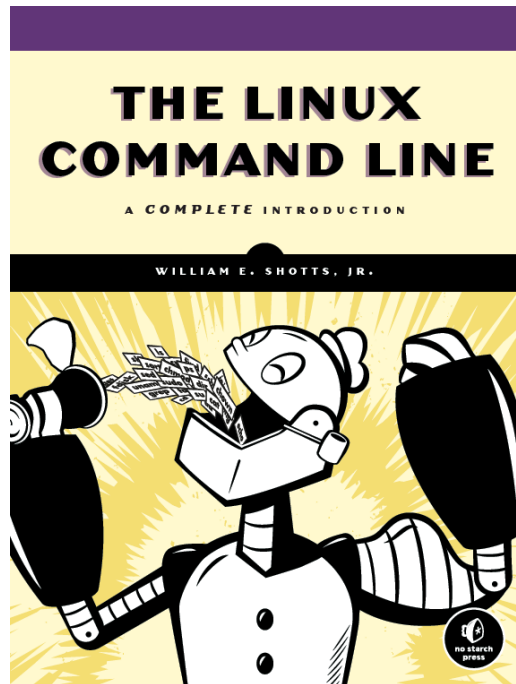
**Have it ready by next week for the PS**, consult your TAs or peers if you have any problems

# Linux Tutorial

- Learning Unix commands
  - <http://www.ee.surrey.ac.uk/Teaching/Unix/>
- Study the intro and first 2 sections by next PS hour
  - Experiment with these basic commands
  - First PS will go over the commands
- First project will require you to have a running Linux environment and basic Unix command knowledge

# Linux Tutorial

- You may also benefit from
  - available for free under CC licence



# Grading

- Grading
  - %5 One Coding Assignment
  - %40 Projects (15+15+10 of them)
  - %20 Midterm
  - %25 Final
  - %10 Attendance

**Final makeup exam and remedial exam will take place on the same day at the same time. A student can take either of them but not both.**

**Midterm makeup exam is on the last week of the instructions at the PS hour. Midterm makeups are not cumulative.**

# TAs and PS Hours

- PS is on Wednesday at 17:30
  - Not every week.
  - We will announce it when it is happening.
- Office Hours
  - Tuesdays 13.00-14.20 or any other time by appointment

# Motivation

- Operating Systems: Major field of Computer Science and Engineering
  - One of the MOST important course
- Around 20% of questions in GRE Computer Science subject test are from the OS concepts
- Forms a good knowledge base for other subject areas
- Provides a complete understanding of software/hardware infrastructure



# Elements

- Good knowledge in
  - C programming
  - Data structures
  - Computer Systems
    - CPU and Memory Subsystem
  - Algorithms

# What is an Operating System?

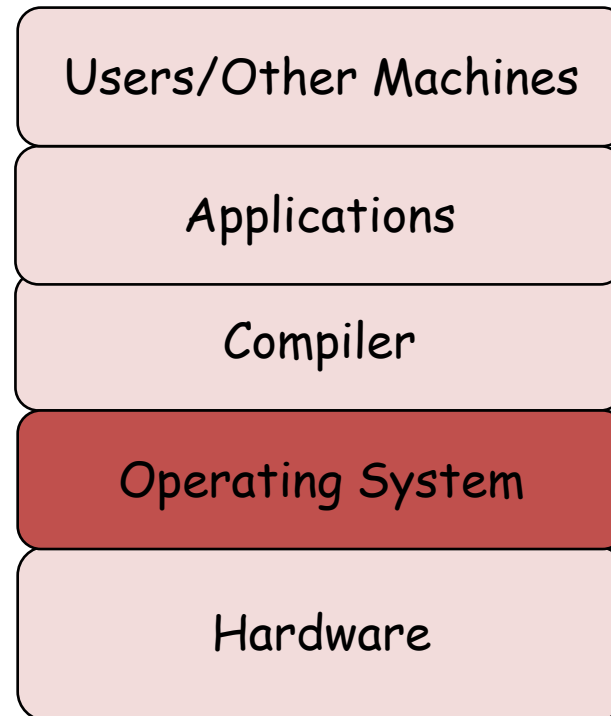
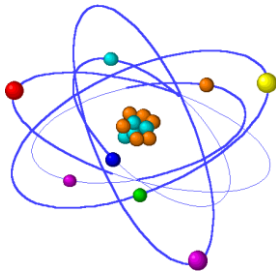
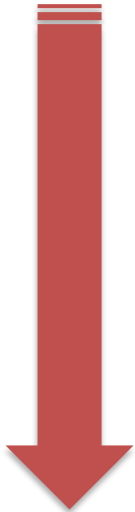
- A program that acts as an intermediary (supervisor) between a user of a computer and the computer resources
- Duties of an OS
  - 1) Provide resource abstraction
  - 2) Manage and coordinate resources
  - 3) Provide *security and protection*
  - 4) Provide *fairness* among users (or programs)

# Computer Startup

- **Bootstrap program** is loaded at power-up or reboot
  - Typically stored in ROM, generally known as **firmware**
  - Initializes all aspects of a system
  - Loads operating system **kernel** into main memory and starts execution
    - The first system process is '**init**' (or systemd) in Linux
  - When the system is fully booted, it waits for some event to occur
- **Kernel**
  - The ``one'' program running at all times (the core of OS)
    - Everything else is an application program
- **Process**
  - An executing program (active program)

# (1) OS creates resource abstractions

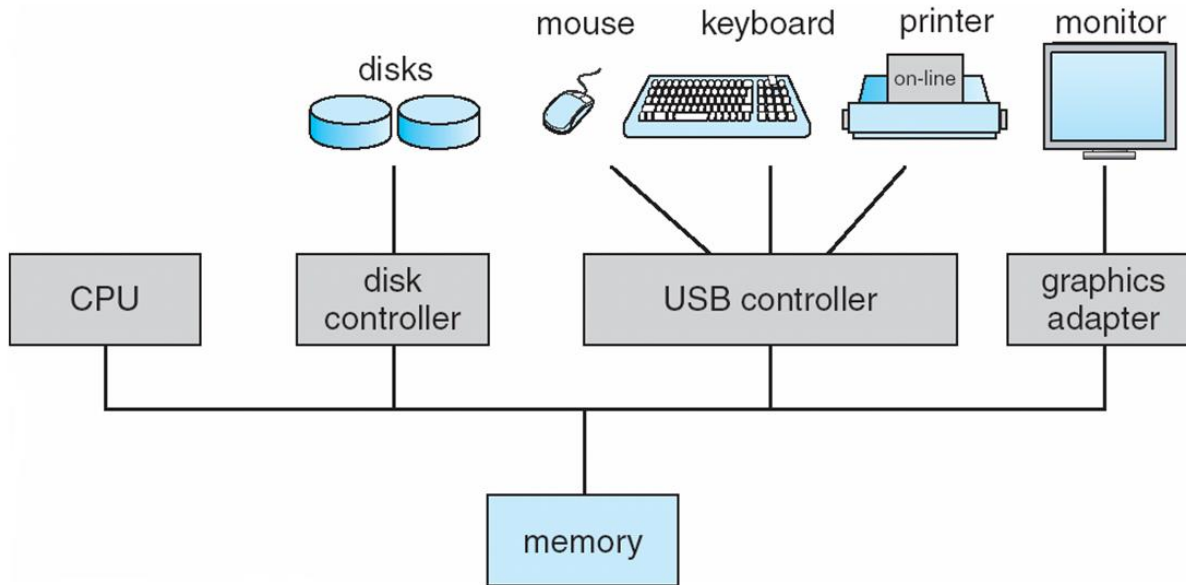
```
foo(int x) {...}
```



There are other layers in software stack such as runtime, libraries etc. Operating System and Compilers are essentials.

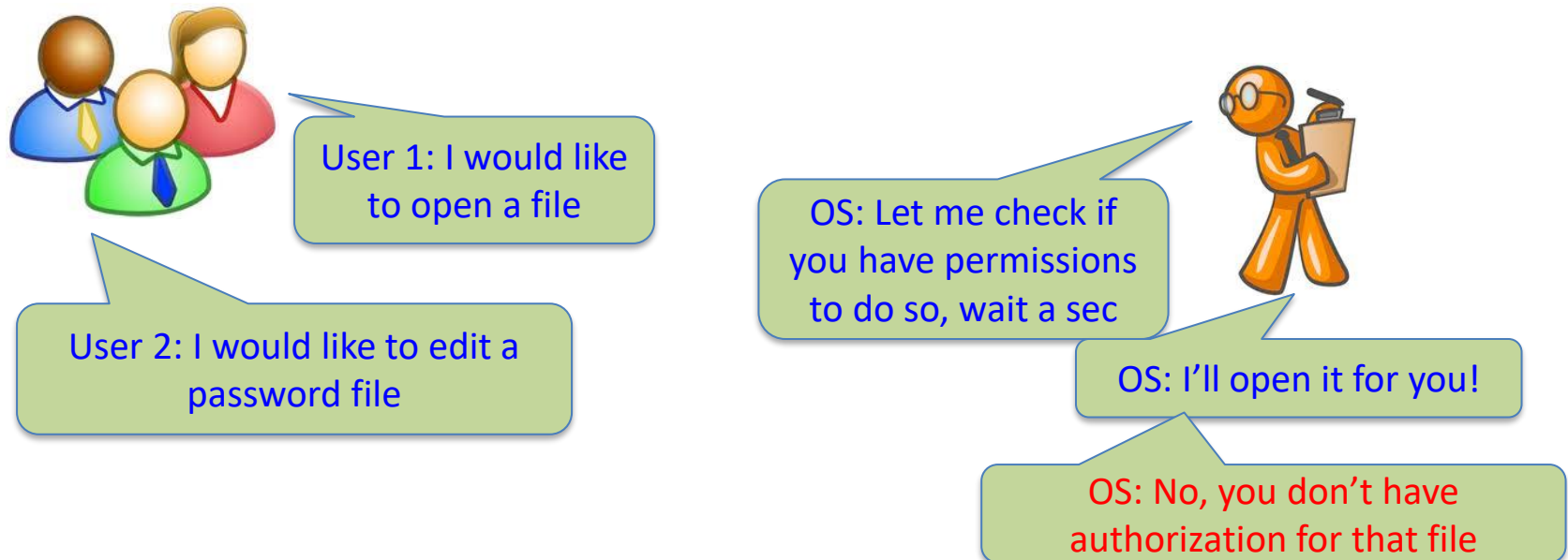
## (2) OS manages resources

- OS is a **resource allocator**
  - Manages all resources for processes
  - Decides between conflicting requests for efficient and fair resource use



# (3) OS provides protection and security

- OS is a **control program**
  - Controls execution of programs to prevent errors and improper/malicious use of the computer
  - Dual mode and Multimode OS
    - User mode and Kernel mode

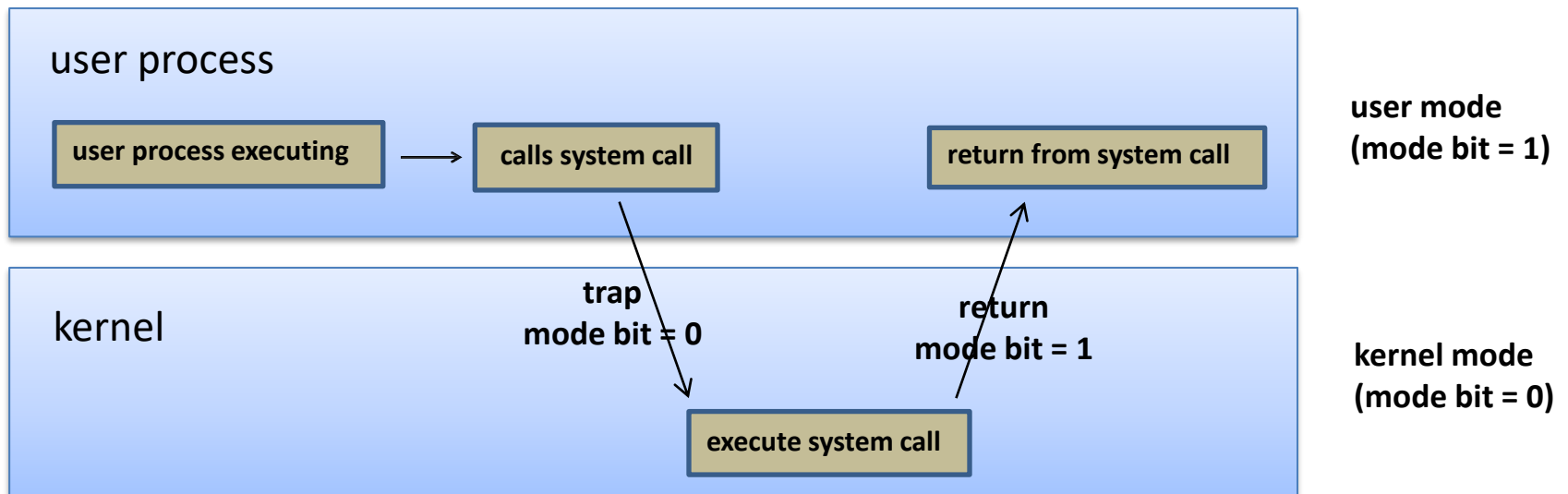


# (3) OS provides protection and security

- **System Call**

- How a program requests a service from an OS
- Results in a transition from user to kernel mode
- Return from call resets it to user mode

- Software error or request creates **exception or trap**



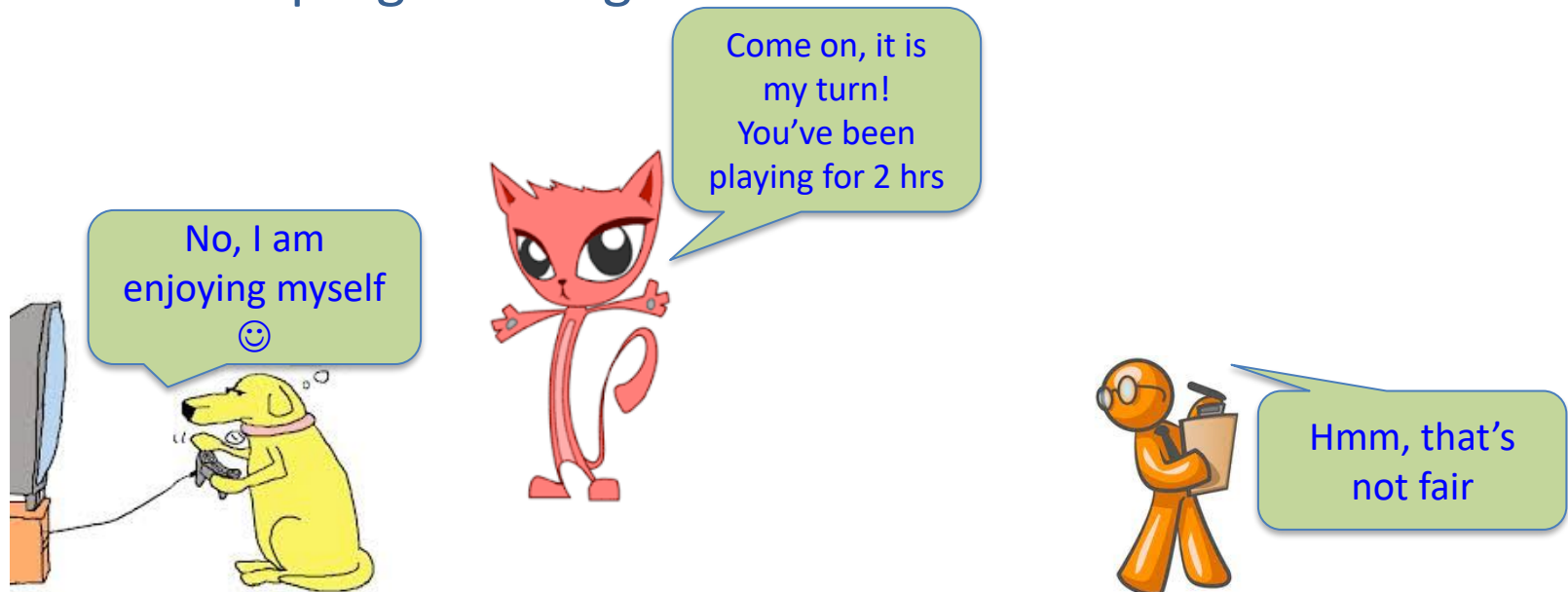
# Interrupts

- An operating system is **interrupt driven**
  - It sits and waits for an event to occur
- Device or hardware interrupts
  - I/O device is done or
  - Hardware throws an exception (e.g. overflow)
- Software interrupts
  - A **trap** or **exception** is a software-generated interrupt caused either by an error or a user request (system call)
- OS has an **interrupt vector**, which contains the addresses of all the service routines for interrupt handling



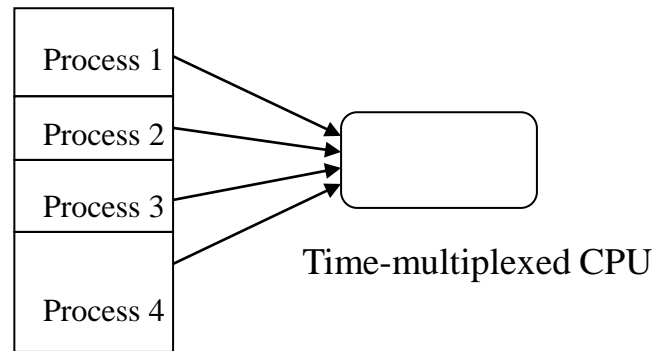
# (4) OS provides *fair* execution

- OS provides fair execution and resource sharing between users and programs
  - via multiprogramming



# How Multiprogramming Works

- **Multiprogramming** needed for efficiency
  - Single user or program cannot keep CPU and I/O devices busy at all times
  - Organize processes so that CPU always has one process to execute
  - A subset of total jobs is kept in main memory
- One job selected and run via **CPU scheduling**
  - When it has to wait (for I/O for example), OS switches to another job



Space-multiplexed Memory

# Brief History of OS

- No operating system 1940s
  - Computers are exotic
  - Program in machine language
  - Programs manually loaded
  - No concurrency: no multiple jobs, no multiple users
- 1950s
  - First compiler is developed
  - OS uses batch scheduling
    - No human-computer interaction
    - Still used in servers, clusters and data centers today

# Brief History of OS

- 1960s
  - Multics – one of the most important real OS
    - Hierarchical file system (directory structure)
    - Access control list and protection
    - <https://multicians.org/>
- 1970s
  - Computers became affordable
  - UNIX is born at Bell Labs by Ken Thompson and Dennis Ritchie
    - Written in C, allows people to experiment

# Brief History of OS

- 1980s
  - MS-DOS
    - IBM needed software for their personal computers
    - Approached Bill Gates (Microsoft) and he created MS-DOS
  - BSD Unix
    - University of California developed BSD Unix
    - Became open source later
  - Mach
    - Carnegie Mellon Univ. developed Mach to replace Unix
    - Apple chose BSD/Mach as the foundation for MacOS X
- 1983
  - Richard Stallman started the GNU project
    - Advocates free, open-source UNIX compatible operating system
    - GNU General Public License (GPL) is now a common license under which free software is released

# Brief History of OS

- 1990s
  - Linux
    - Developed by a student (Linus Torvalds) in Finland
    - Unix-based
    - Several distributions: SUSE, Fedora, Ubuntu, Redhat ....
    - Open-source operating system under GNU General Public License
  - Windows 95 and MacOS X became mature and complex
- 2000s
  - Mobile devices: Android (based on Linux)
  - Trend is to have a smaller OS (network storage)
  - Virtualization has become common (Vmware Player, VirtualBox etc.)

# Reading

- From text book
  - Read 1.1, 1.4-1.10 (OS Structure – Kernel Data Structures)
  - Read 1.12 (Open-Source OS)
  - Read 1.2-1.3 if you want to refresh your Computer Architecture knowledge
- Install the Linux Distribution or Virtual Machine by next PS
- Subscribe to Blackboard Discussion Forum

# Acknowledgments

- Original slides are by Didem Unat which were adapted from
  - Öznur Özkasap (Koç University)
  - Operating System and Concepts (9<sup>th</sup> edition) Wiley