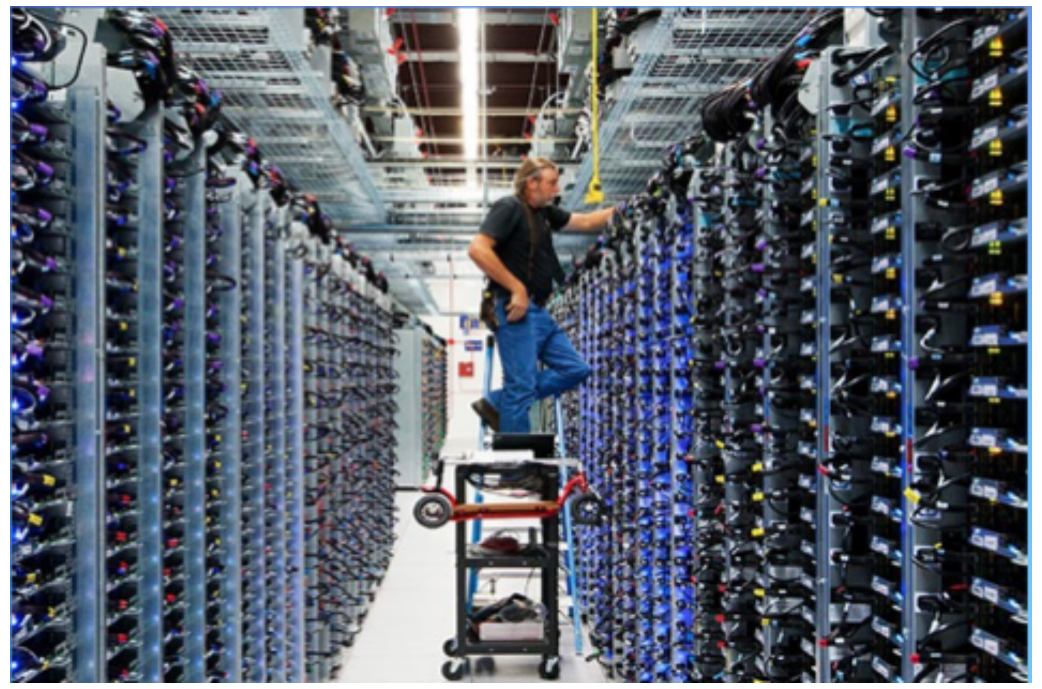
Systems Programming

Prof. Shih-Wei Li

Department of Computer Science and Information Engineering
National Taiwan University

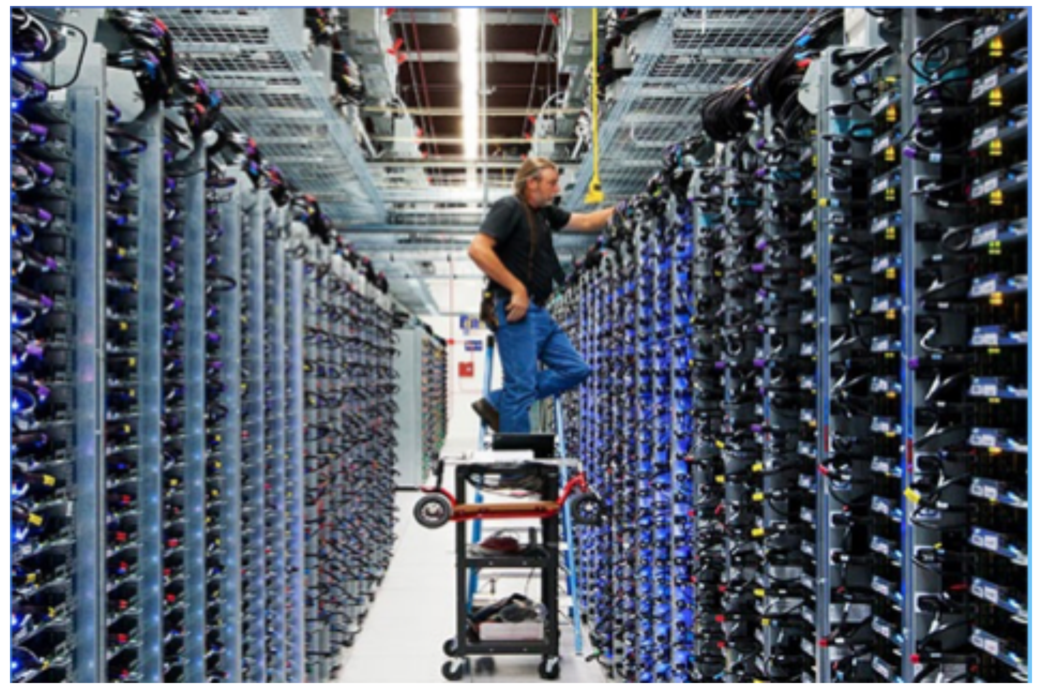
Buzzwords

- Cloud computing, mobile systems, computer security, server/network security/ administration, virtual machines, compilers, etc.



Course Objectives

- Q: I have learned many programming languages, what do I still have to learn from the course?
  - Learn about what features are provided by the (Unix-like) Operating System (OS)
  - Learn how to write programs that interact with the OS



Course Objectives

- Learn basic OS concepts and Unix-based systems:
    - design concepts and implementations
- Throughout the course lectures and assignments, you will learn systems programming in Unix systems using C

- By the end of the semester, you should know the following pretty well:
    - How to write systems programs that interact with the OS and use systems resources: network, files, etc.
    - How to write systems programs with good performance
    - How to write correct and safe systems program

Course Map

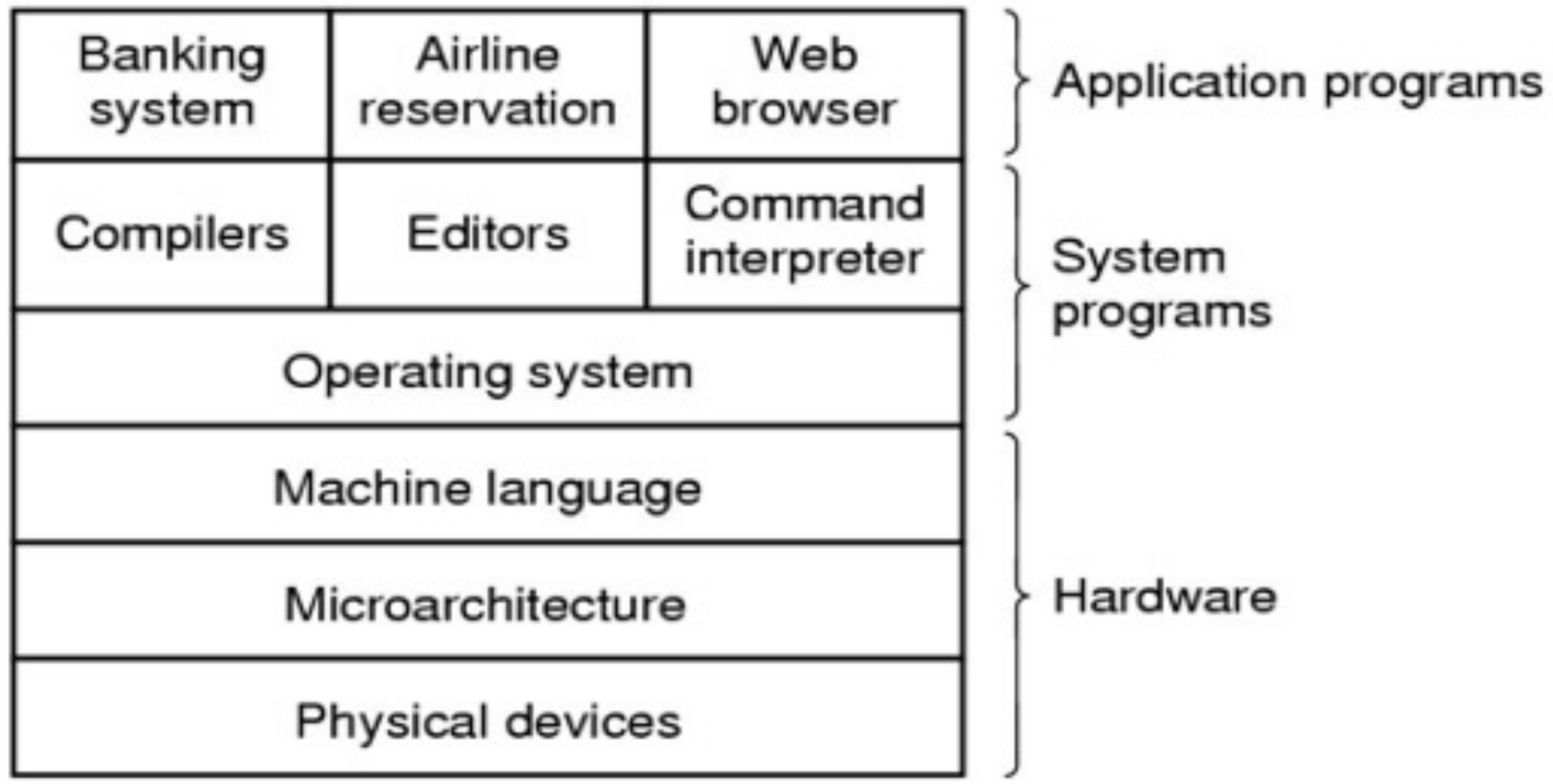
- SP gets you prepared for more advanced courses:
  - Operating Systems
  - Computer Network
  - Computer Security
  - System Design Topic - Design for IoT Middleware
  - Virtual Machines
  - Advanced Operating Systems, Advanced Topics in Software Systems Design and Implementation

About Me

- Shih-Wei Li (黎士瑋). Assistant Professor in the CSIE Department at NTU
- I got my Ph.D. in Computer Science from Columbia University
- Lead Secure Systems Lab
- My current research intersects areas of operating systems, security, computer architecture, and programming languages

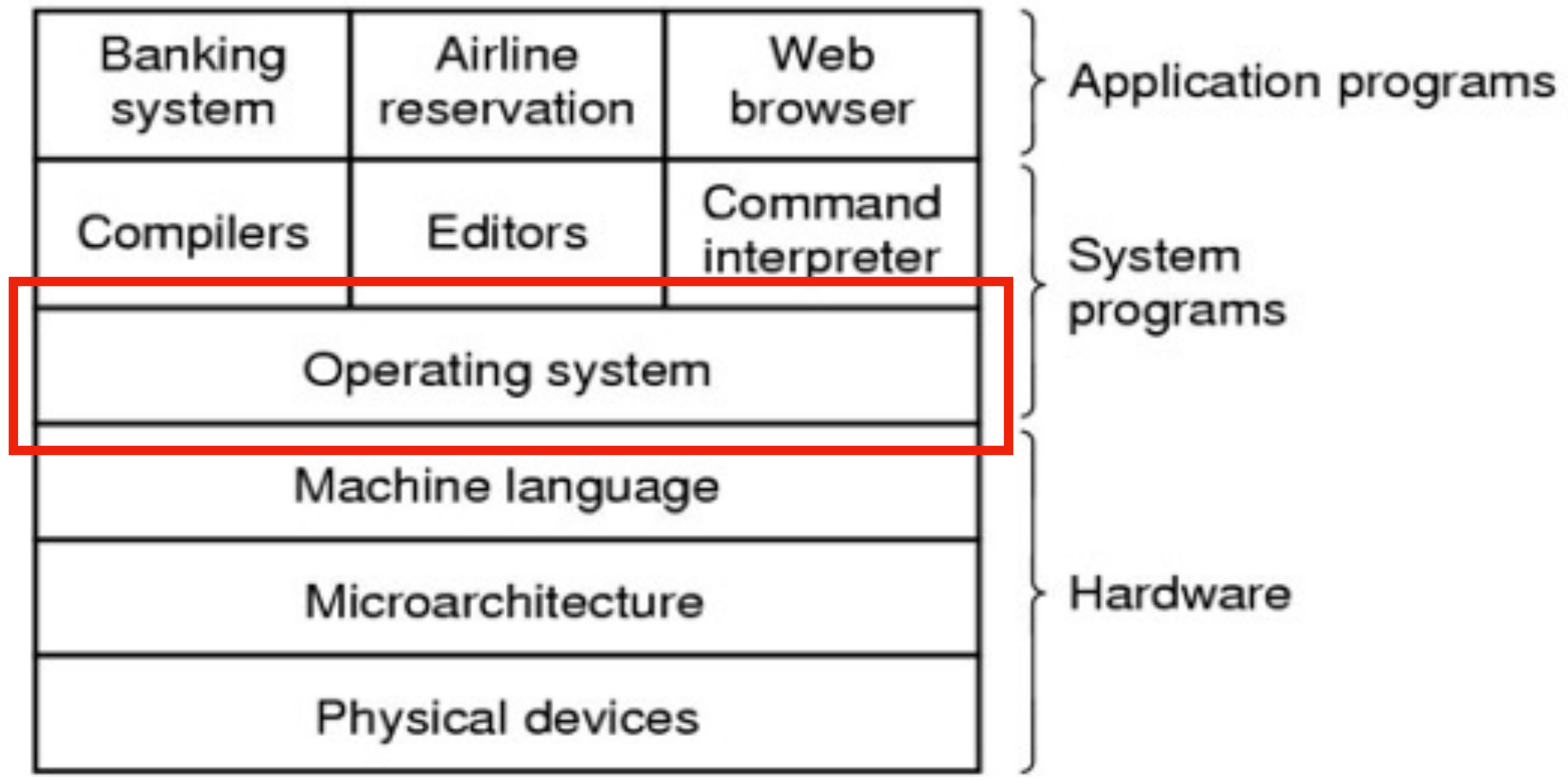
Computer Systems

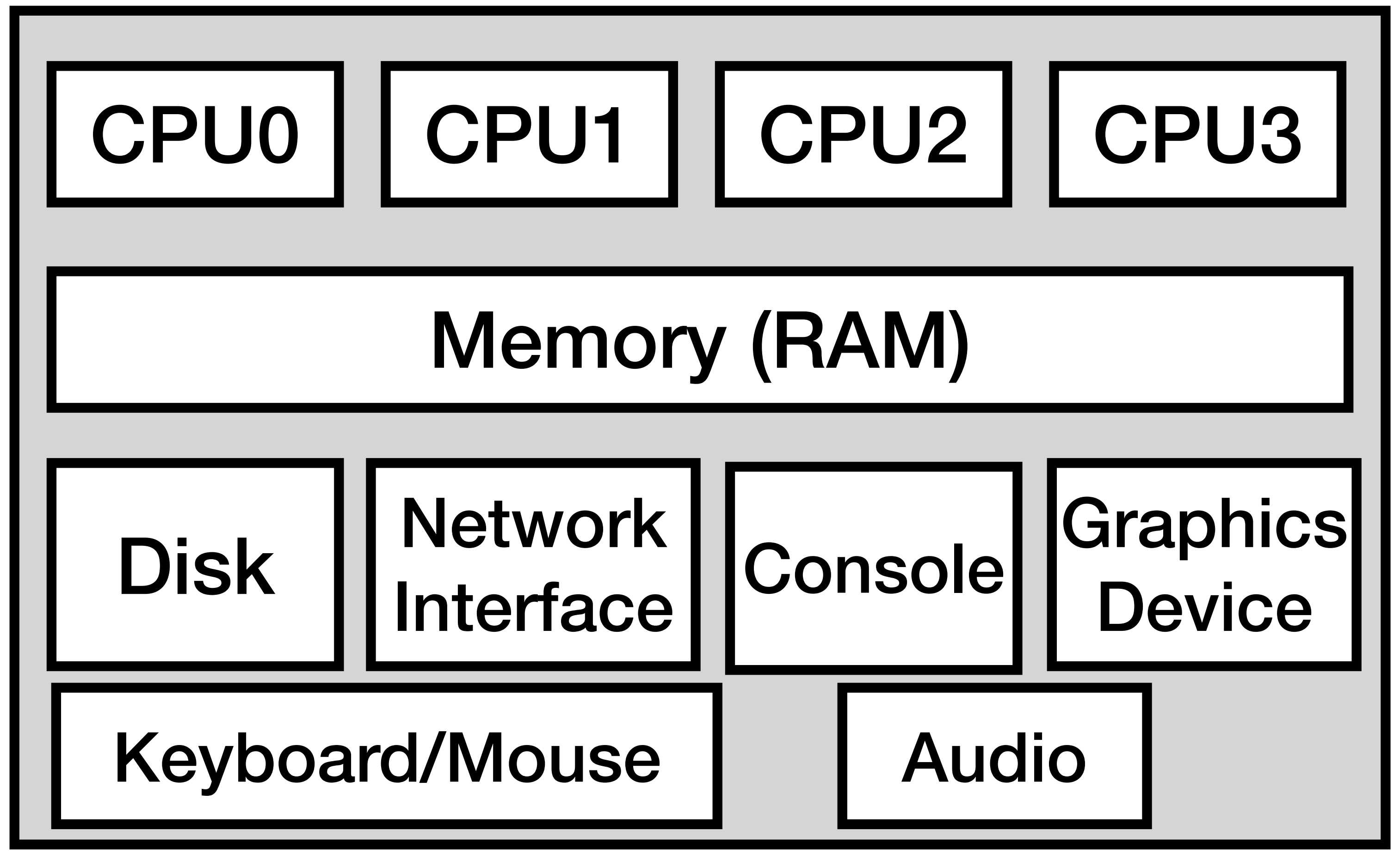
- A computer system consists of:
  - Hardware
  - Systems programs (systems software)
  - Application programs



Operating Systems

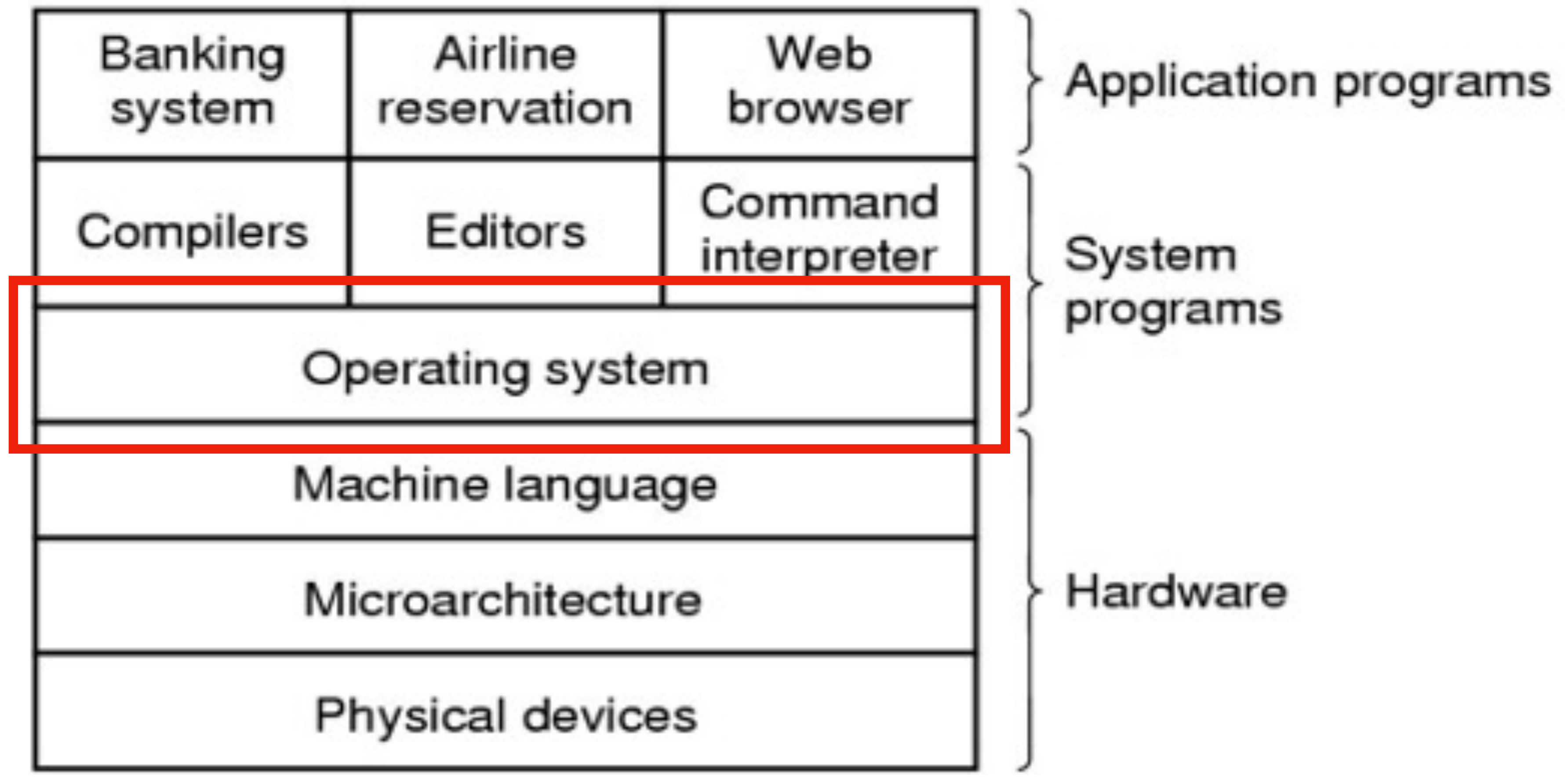
- An Operating System:
  - Runs on the physical hardware/machine to manage resources for each different application/programs on systems
  - Abstracts the machine resources and provides an interface for others to use its functionality
  - Hides messy details/complexities from the applications and (some) system programs; why is this important?





Operating Systems

- All Operating Systems provide ***services*** for programs that run on top of them:
  - Executing/stopping a program
  - File systems: opening/reading/writing/closing/creating/deleting a file
  - Getting the current time of the day
  - System utilities and management tools
  - Many more!



Operating Systems: Case Study

- Operating Systems (OS) v.s. Operating Systems Kernels
  - OS: encompasses a broader set of functionalities and usually includes an OS kernel
  - OS Kernel: directly manages hardware resources and provides core services
- You might have heard of the following: Ubuntu, Linux, Android OS, macOS, Windows
  - How do they fit in the picture?

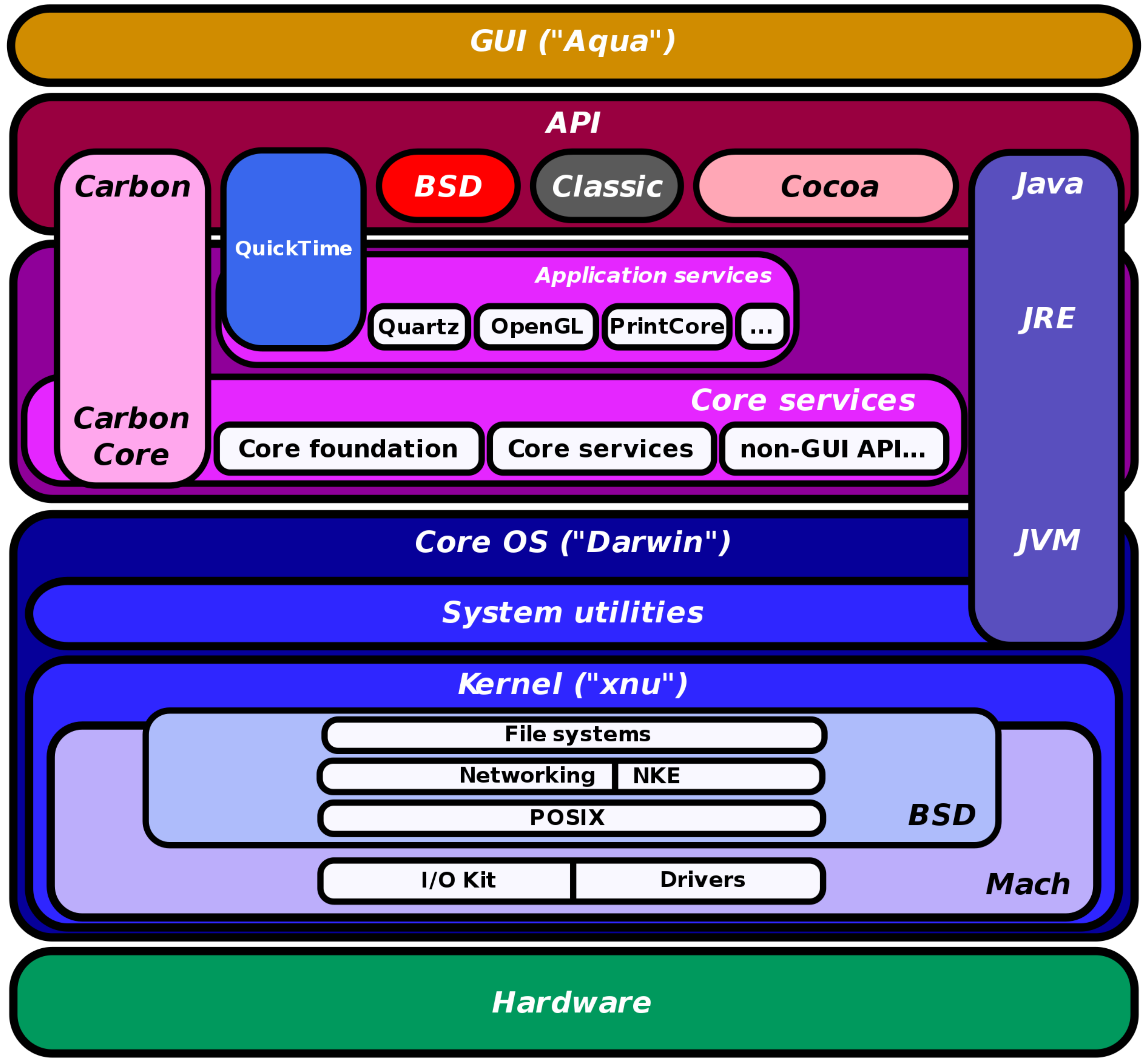
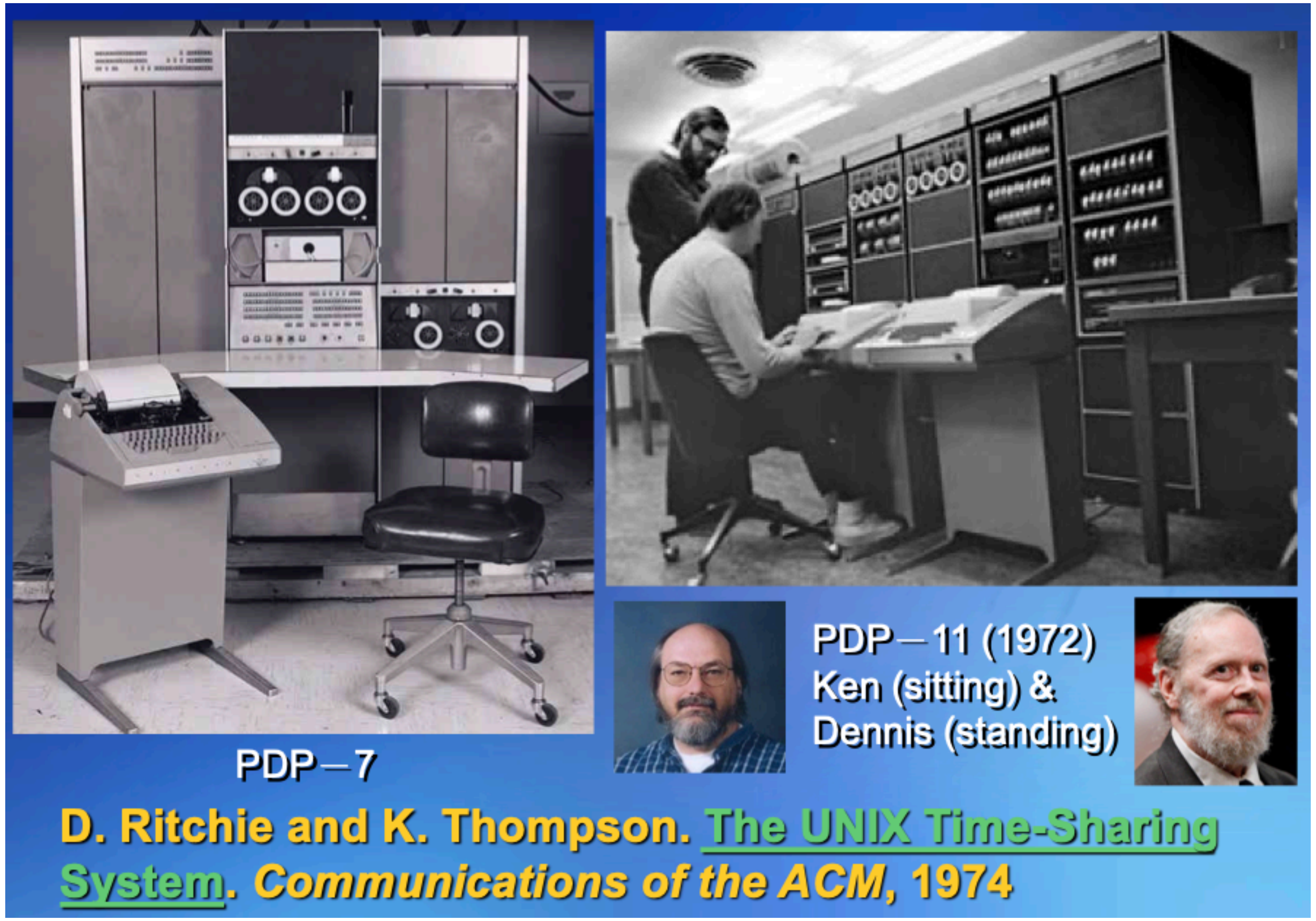


Diagram of macOS architecture: https://upload.wikimedia.org/wikipedia/commons/thumb/f/f2/Diagram_of_Mac_OS_X_architecture.svg/1920px-Diagram_of_Mac_OS_X_architecture.svg.png

Unix Operating System

- Unix was developed in 1969 by Ken Thompson and Dennis Ritchie and others at Bell Labs
  - Unix version 0 runs on PDP-7
  - Unix was initially written in assembly; rewritten in C in 1973

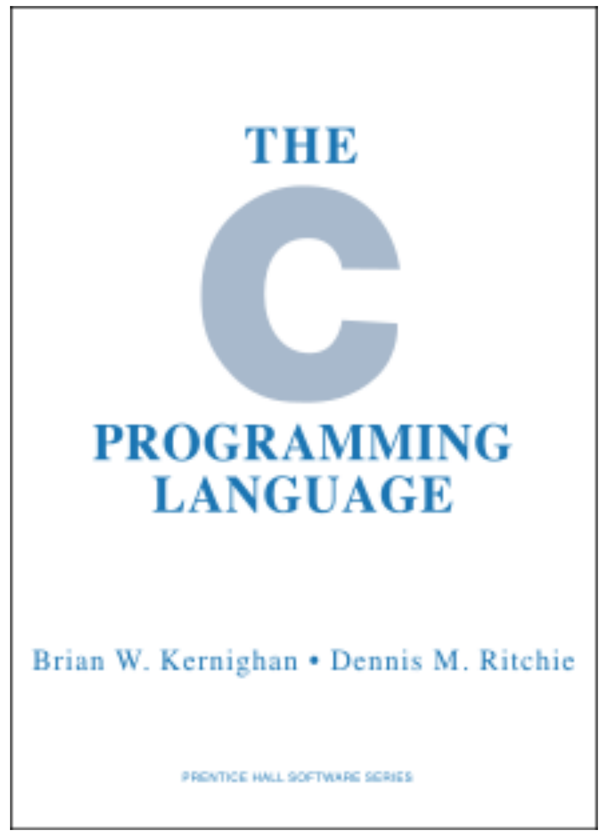


From: https://www.csie.ntu.edu.tw/%7Epjcheng/course/sp2022/sp_overview.pdf

Unix Operating System

https://hackaday.com/2019/11/17/unix-version-0-running-on-a-pdp-7-in-2019/

Unix Operating System



- Fun facts:
  - Ken Thompson & Dennis Ritchie were the ACM Turning award winners for their development of generic operating systems theory and specifically for the implementation of the UNIX operating system
  - Dennis Ritchie is also the inventor of the C programming language

Unix Operating System

- History of Unix: https://Unix.org/what_is_Unix/history_timeline.html
- Bell Labs made the first Unix (V6) release in 1975 - used by researchers at the universities
  - Ken Thompson and graduate students at UC Berkeley developed the Berkeley Software Distribution (BSD) based on V6 in 1977
- Unix diverges into to main versions: BSD and System V (by AT&T) — two main branches for later Unix-based OS implementations
- List of Unix variants that you may have heard of: ***Linux*** (RedHat, Ubuntu, Fedora, ArchLinux, CentOS), ***BSD*** (FreeBSD, NetBSD), ***Others*** (macOS, Minix, Solaris)

Unix System Architecture

- The *Unix OS kernel* provides machine bootstrap, systems initialization, interrupt and exception handling, process scheduling, memory management, and I/Os
- The kernel exposes services via the *system calls* interface

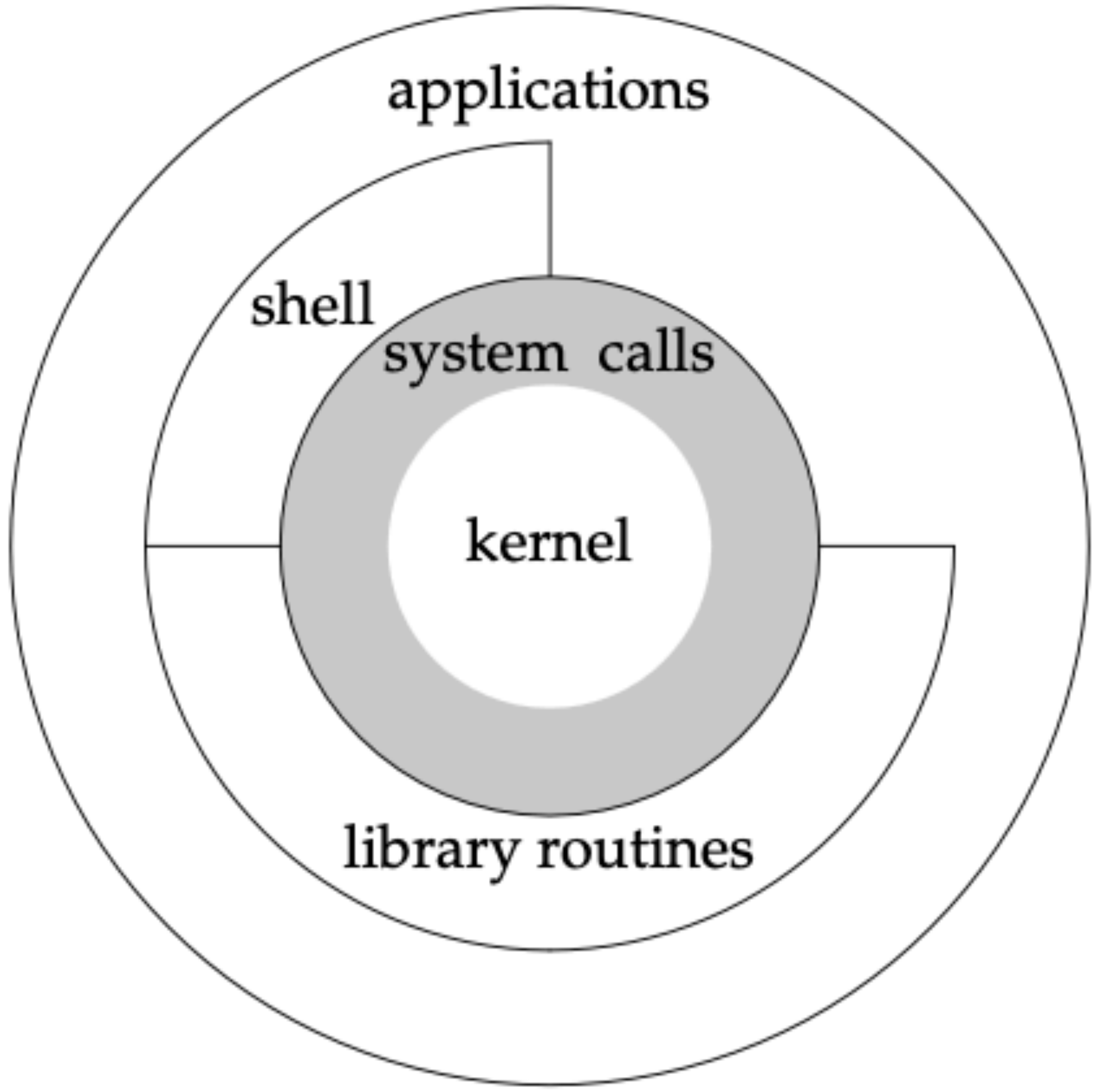


Figure 1.1 Architecture of the UNIX operating system

Unix System Architecture

- *Applications* are written and compiled by programmers into binaries
- *Library routines* provide pre-compiled binaries (e.g. header files like “stdio.h”)
- *Shell* exposes an interactive interface for users to issue commands to run applications
  - Note: “applications” that we refer to here and later include both “applications and some of the system programs” shown earlier

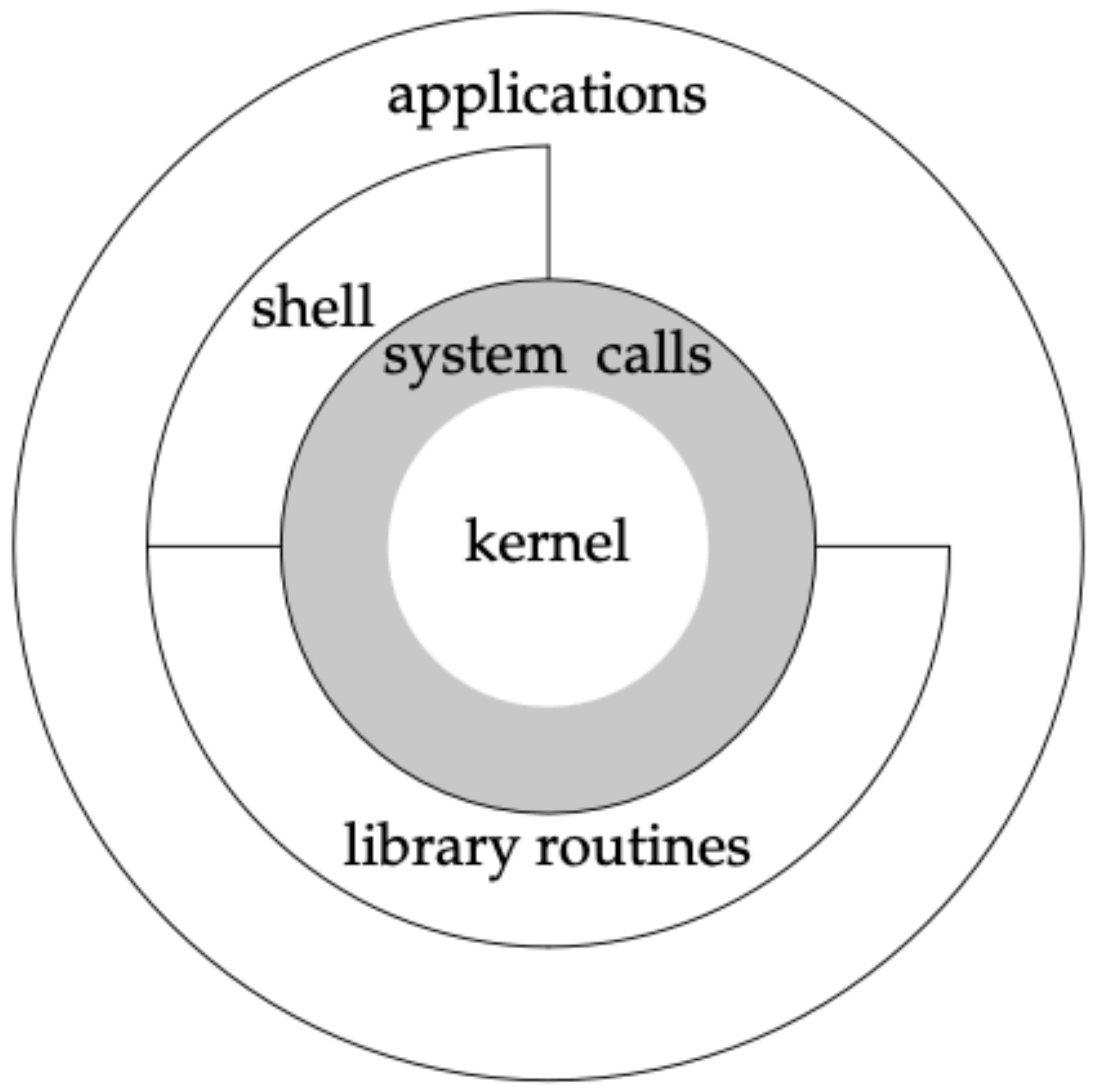


Figure 1.1 Architecture of the UNIX operating system

Unix Shell

```
shihwei — shihwei@linux1:~ — ssh parallels@10.211.55.4 — 130×40
parallels@ubuntu-linux-20-04-desktop:~$ ssh shihwei@linux1.csie.ntu.edu.tw
shihwei@linux1.csie.ntu.edu.tw's password:

##########################################################
#        Public Domain Workstation Lab (R217).           #
##########################################################
#   UNIX Login Service:                                  #
#     FreeBSD - bsd1                                     #
#     Linux   - linux1, linux2, linux3, ... linux15      #
#             - [NEW!] meow1, meow2 (with GPU)           #
#             - oasis1, oasis2, oasis3 (non-computing)   #
#     Run `ws-status` for current machine status         #
#                                                        #
#   Office open time:                                    #
#     08:30 ~ 17:00, otherwise please use accesscards    #
#                                                        #
#   Contact information:                                 #
#     Web:    https://wslab.csie.ntu.edu.tw/             #
#     E-Mail (linux): ta217@csie.ntu.edu.tw              #
#     E-Mail (bsd)  : lantw44@csie.ntu.edu.tw            #
#                                                        #
########################## Last Update: Sep  26 2018 ###
Last login: Fri Jul 15 13:36:53 2022 from 140.112.29.31
mail: /var/spool/mail/shihwei: No such entry, file or directory
西元2022年07月15日 (週五) 13時37分13秒 CST

shihwei@linux1 [~]
```

OpenSSH SSH client (program to login to remote shell)

Shell prompt: where you type commands

Unix Shell

shihwei — parallels@ubuntu-linux-20-04-desktop: ~ — ssh parallels@10.211.55.4 — 130×40

```
shihwei@linux1 [~] cd sp21/
shihwei@linux1 [~/sp21] cat sp01-demo1.c
#include <unistd.h>
#include <sys/syscall.h>
#include <stdio.h>
#include <string.h>

int main()
{
        char *hello = "hello world\n";

        write(1, hello, strlen(hello));
        syscall(SYS_write, 1, hello, strlen(hello));
        printf("%s", hello);
}
shihwei@linux1 [~/sp21] gcc -Wall sp01-demo1.c -o syscall
shihwei@linux1 [~/sp21] ./syscall > out
shihwei@linux1 [~/sp21] cat out
hello world
hello world
hello world
shihwei@linux1 [~/sp21] ls -al out
-rw-r--r-- 1 shihwei faculty 36  7月  15 13:59 out
shihwei@linux1 [~/sp21] date
西元2022年07月15日 (週五) 14時00分21秒 CST
shihwei@linux1 [~/sp21] exit
登出
Connection to linux1.csie.ntu.edu.tw closed.
parallels@ubuntu-linux-20-04-desktop:~$
```

Open sp01-demo1.c and output the contents to the terminal

Call the C compiler (GCC) to compile the program into “syscall”

Run the “syscall” program and put its output to the file “out”

Unix Shell: System Calls

shihwei — parallels@ubuntu-linux-20-04-desktop: ~ — ssh parallels@10.211.55.4 — 130×40

```
shihwei@linux1 [~] cd sp21/
shihwei@linux1 [~/sp21] cat sp01-demo1.c
#include <unistd.h>
#include <sys/syscall.h>
#include <stdio.h>
#include <string.h>

int main()
{
        char *hello = "hello world\n";

        write(1, hello, strlen(hello));
        syscall(SYS_write, 1, hello, strlen(hello));
        printf("%s", hello);
}
shihwei@linux1 [~/sp21] gcc -Wall sp01-demo1.c -o syscall
shihwei@linux1 [~/sp21] ./syscall > out
shihwei@linux1 [~/sp21] cat out
hello world
hello world
hello world
shihwei@linux1 [~/sp21] ls -al out
-rw-r--r-- 1 shihwei faculty 36  7月  15 13:59 out
shihwei@linux1 [~/sp21] date
西元2022年07月15日 (週五) 14時00分21秒 CST
shihwei@linux1 [~/sp21] exit
登出
Connection to linux1.csie.ntu.edu.tw closed.
parallels@ubuntu-linux-20-04-desktop:~$
```

The three lines here each makes a call to output the string hello

Unix Shell

- For more on this, you can checkout: http://cslibrary.stanford.edu/107/UnixProgrammingTools.pdf
- *Section 5* covers the commonly used commands in Unix Shells

Common Unix shells

Read Chap. 1.3 for more details about shells

Name	Path	FreeBSD 8.0	Linux 3.2.0	Mac OS X 10.6.8	Solaris 10
Bourne shell	`/bin/sh`	•	•	copy of `bash`	•
Bourne-again shell	`/bin/bash`	optional	•	•	•
C shell	`/bin/csh`	link to `tcsh`	optional	link to `tcsh`	•
Korn shell	`/bin/ksh`	optional	optional	•	•
TENEX C shell	`/bin/tcsh`	•	optional	•	•

Figure 1.2 Common shells used on UNIX systems

From APUE 3rd Edition: Figure 1.2

Summary: Course Objectives

- In this course, you will learn how to write programs on top of the Unix OS kernel
  - We will discuss Unix OS design, but that is only to help you understand how to do systems programming
  - You will learn more about the ***OS kernel*** internals in the Operating Systems class
- After taking SP, you should expect to know:
  - How to write a shell
  - How to write system services/commands that you use in the Unix environment

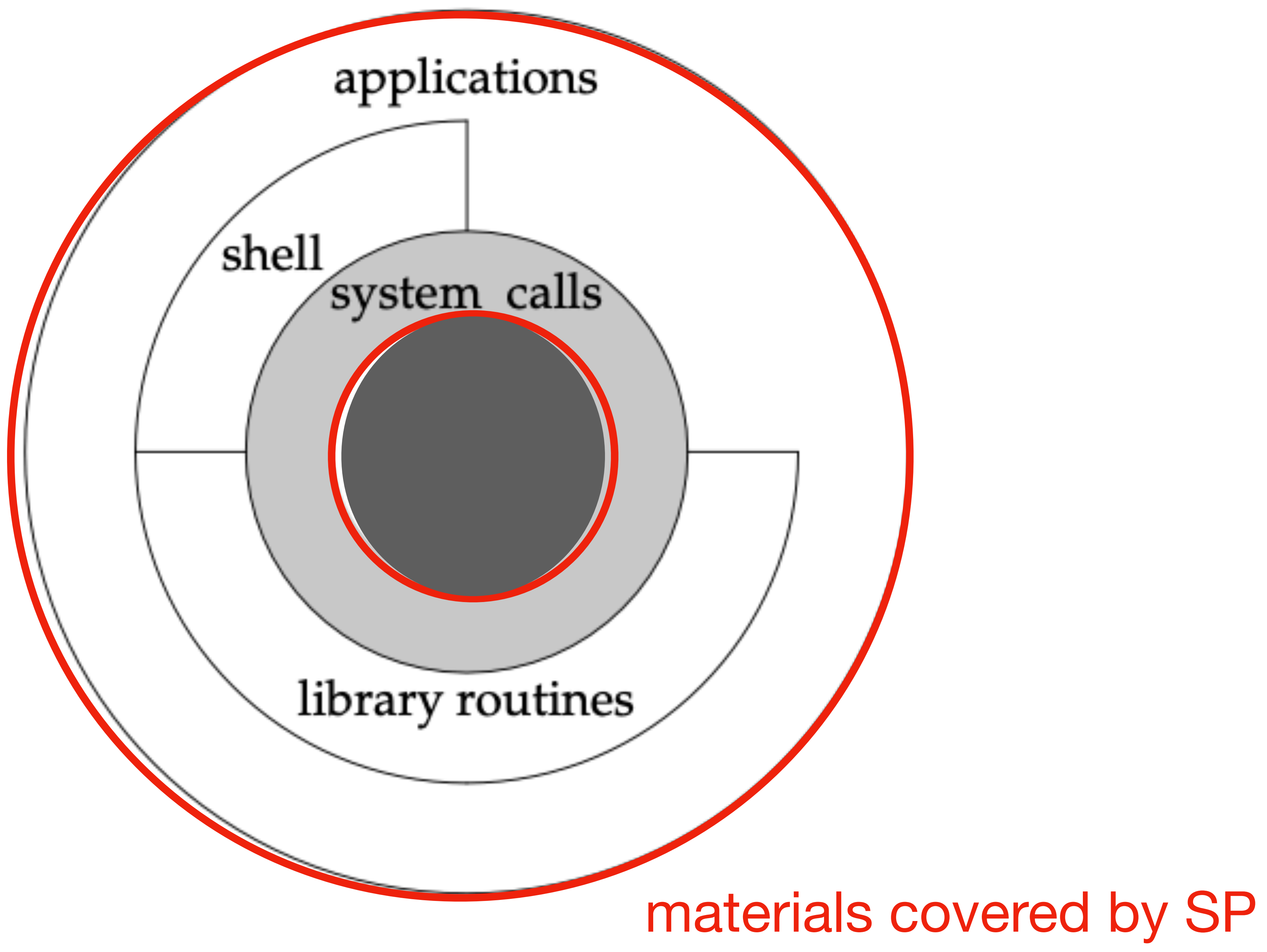


Figure 1.1 Architecture of the UNIX operating system