

## EECS 3221: OPERATING SYSTEM FUNDAMENTALS

Hamzeh Khazaei

Department of Electrical Engineering and  
Computer Science

Week 2, Module 1:  
Operating System Structures

# Assignment Project Exam Help

Operating System Concepts – 10<sup>th</sup> Edition

2.1

Silberschatz et al © 2018 updated by Khazaei @ 2021

1

<https://powcoder.com>

Add WeChat powcoder

- Operating System Services
- User and Operating System-Interface
- System Calls
- System Services
- Linkers and Loaders
- Why Applications are Operating System Specific
- Operating-System Design and Implementation
- Operating System Structure
- Building and Booting an Operating System
- Operating System Debugging

Operating System Concepts – 10<sup>th</sup> Edition

2.2

Silberschatz et al © 2018 updated by Khazaei @ 2021

2

## Operating System Services

- Operating systems provide an environment for execution of programs and services to programs and users
- One set of operating-system services provides functions that are helpful to the user:
  - **User interface** - Almost all operating systems have a user interface (**UI**).
    - ▶ Varies between **Command-Line (CLI)**, **Graphics User Interface (GUI)**, **touch-screen**, **Batch**
  - **Program execution** - The system must be able to load a program into memory and to run that program, end execution, either normally or abnormally (indicating error)
  - **I/O operations** - A running program may require I/O, which may involve a file or an I/O device

## Assignment Project Exam Help

Operating System Concepts – 10<sup>th</sup> Edition

2.4

Silberschatz et al © 2018 updated by Khazaei @ 2021

4

<https://powcoder.com>

## Add WeChat powcoder Operating System Services (Cont.)

- One set of operating-system services provides functions that are helpful to the user (Cont.):
  - **File-system manipulation** - The file system is of particular interest. Programs need to read and write files and directories, create and delete them, search them, list file information, permission management.
  - **Communications** – Processes may exchange information, on the same computer or between computers over a network
    - ▶ Communications may be via **shared memory** or through **message passing** (packets moved by the OS)
  - **Error detection** – OS needs to be constantly aware of possible errors
    - ▶ May occur in the CPU and memory hardware, in I/O devices, in user programs
    - ▶ For each type of error, OS should take the appropriate action to ensure correct and consistent computing
    - ▶ Debugging facilities can greatly enhance the user's and programmer's abilities to efficiently use the system

Operating System Concepts – 10<sup>th</sup> Edition

2.5

Silberschatz et al © 2018 updated by Khazaei @ 2021

5

2

## Operating System Services (Cont.)

- Another set of OS functions exists for ensuring the efficient operation of the system itself via resource sharing
  - **Resource allocation** - When multiple users or multiple jobs running concurrently, resources must be allocated to each of them
    - ▶ Many types of resources - CPU cycles, main memory, file storage, I/O devices.
  - **Logging** - To keep track of which users use how much and what kinds of computer resources
  - **Protection and security** - The owners of information stored in a multiuser or networked computer system may want to control use of that information, concurrent processes should not interfere with each other
    - ▶ **Protection** involves ensuring that all access to system resources is controlled
    - ▶ **Security** of the system from outsiders requires user authentication, extends to defending external I/O devices from invalid access attempts

## Assignment Project Exam Help

Operating System Concepts – 10<sup>th</sup> Edition

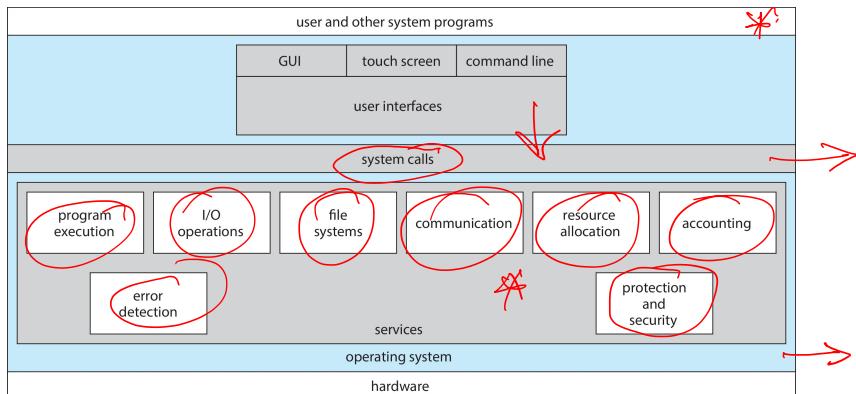
2.6

Silberschatz et al © 2018 updated by Khazaei @ 2021

6

<https://powcoder.com>

## A view of operating system services



Operating System Concepts – 10<sup>th</sup> Edition

2.7

Silberschatz et al © 2018 updated by Khazaei @ 2021

7

## User Operating System Interface - CLI

CLI or **command interpreter** allows direct command entry

- Sometimes implemented in kernel, sometimes by systems program
- Sometimes multiple flavors implemented – **shells**
- Primarily fetches a command from user and executes it
- Sometimes commands built-in, sometimes just names of programs
  - ▶ If the latter, adding new features doesn't require shell modification

## Assignment Project Exam Help

Operating System Concepts – 10<sup>th</sup> Edition

2.8

Silberschatz et al © 2018 updated by Khazaei @ 2021

8

<https://powcoder.com>

## Add WeChat powcoder Bourne Shell command interpreter

```
1. root@r6181-d5-us01:~ (ssh)
root@r6181-d5-us01:~ pbgs ssh root@r6181-d5-us01
root@r6181-d5-us01's password:
Last login: Thu Jul 14 08:47:01 on ttys002
iMacPro:~ pbgs$ ssh root@r6181-d5-us01
root@r6181-d5-us01:~]# uptime
 06:57:48 up 16 days, 10:52,  3 users,  load average: 129.52, 80.33, 56.55
[root@r6181-d5-us01 ~]# df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/mapper/vg_ks-lv_root
                  50G   28G   41G  41% /
tmpfs          127G  520K  127G   1% /dev/shm
/dev/sda1       477M   71M   381M  16% /boot
/dev/dssd0000    1.0T  480G  545G  47% /dssd_xfs
tcp://192.168.150.1:3334/orangefs
                  12T  5.7T   6.4T  47% /mnt/orangefs
/dev/gpfs-test   23T  1.1T   22T   5% /mnt/gpfs
[root@r6181-d5-us01 ~]#
[root@r6181-d5-us01 ~]# ps aux | sort -nrk 3,3 | head -n 5
root     97653 11.2  6.6 42665344 17520636 ? S<Ll Jul13 166:23 /usr/lpp/mmfs/bin/mmfsd
root     69849  6.6  0.0    0    0 ?      S Jul12 181:54 [vpthread-1-1]
root     69850  6.4  0.0    0    0 ?      S Jul12 177:42 [vpthread-1-2]
root     3829  3.0  0.0    0    0 ?      S Jun27 730:04 [rp_thread 7:0]
root     3826  3.0  0.0    0    0 ?      S Jun27 728:08 [rp_thread 6:0]
[root@r6181-d5-us01 ~]# ls -l /usr/lpp/mmfs/bin/mmfsd
-r-x----- 1 root root 20667161 Jun  3 2015 /usr/lpp/mmfs/bin/mmfsd
[root@r6181-d5-us01 ~]#
```

Operating System Concepts – 10<sup>th</sup> Edition

2.9

Silberschatz et al © 2018 updated by Khazaei @ 2021

9

# User Operating System Interface - GUI

- User-friendly **desktop** metaphor interface
    - Usually mouse, keyboard, and monitor
    - **Icons** represent files, programs, actions, etc
    - Various mouse buttons over objects in the interface cause various actions (provide information, options, execute function, open directory (known as a **folder**)
      - Invented at Xerox PARC
  - Many systems now include both CLI and GUI interfaces
    - Microsoft Windows is GUI with CLI “command” shell
    - Apple Mac OS X is “Aqua” GUI interface with UNIX kernel underneath and shells available

# Assignment Project Exam Help

Operating System Concepts – 10<sup>th</sup> Edition

2.10

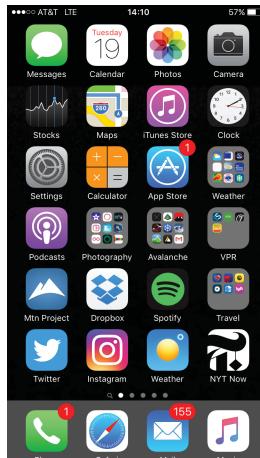
Silberschatz et al © 2018 updated by Khazaei @ 2021

10

<https://powcoder.com>

# Add WeChat powcoder Touchscreen Interfaces

- Touchscreen devices require new interfaces
    - Mouse not possible or not desired
    - Actions and selection based on gestures
    - Virtual keyboard for text entry
  - Voice commands
  - Samsung unveiled a new input method for smartphone users at CES 2020.
    - It's called SelfieType and it was created by Samsung's C-Lab division as a new **virtual keyboard** powered by AI and image recognition technologies.



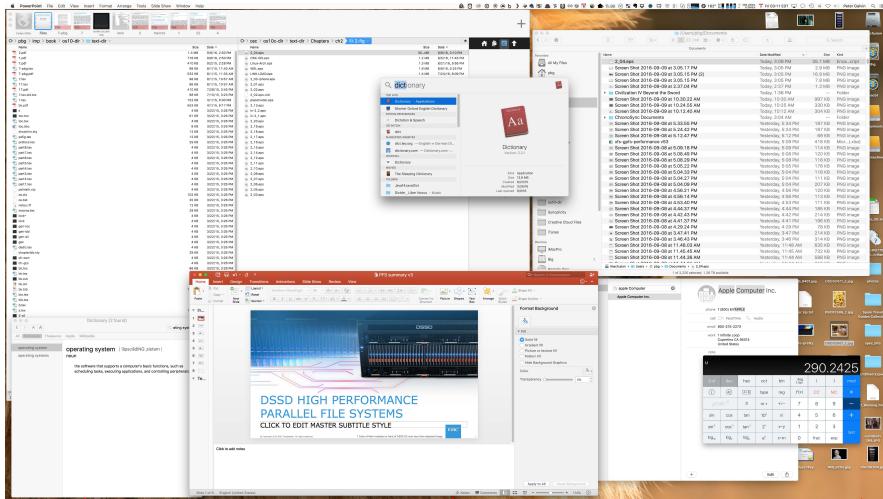
Operating System Concepts – 10<sup>th</sup> Edition

2.11

Silberschatz et al © 2018 updated by Khazaei @ 2021

11

## The Mac OS X GUI



## Assignment Project Exam Help

Operating System Concepts – 10<sup>th</sup> Edition

2.12

Silberschatz et al © 2018 updated by Khazaei @ 2021

12

<https://powcoder.com>

## Add WeChat [powcoder](https://powcoder.com)

- Programming interface to the services provided by the OS
- Typically written in a high-level language (C or C++)
- Mostly accessed by programs via a high-level **Application Programming Interface (API)** rather than direct system call use
- Three most common APIs are **Win32 API** for Windows, **POSIX API** for POSIX-based systems (including virtually all versions of UNIX, Linux, and Mac OS X), and Java API for the Java virtual machine (JVM)

*Note that the system-call names used throughout this text are generic*

Operating System Concepts – 10<sup>th</sup> Edition

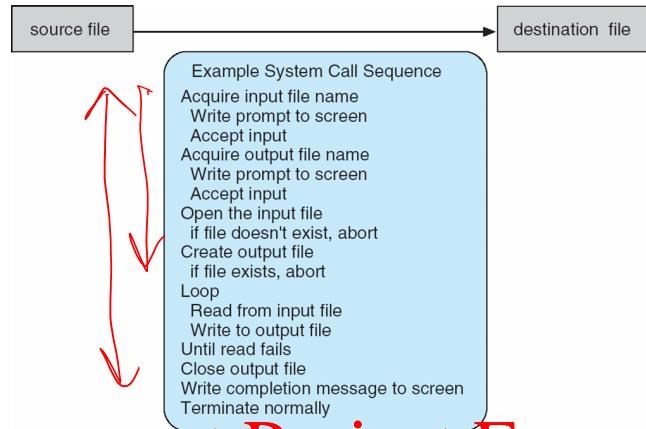
2.13

Silberschatz et al © 2018 updated by Khazaei @ 2021

13

## Example of System Calls

- System call sequence to copy the contents of one file to another file



## Assignment Project Exam Help

Operating System Concepts – 10<sup>th</sup> Edition

2.14

Silberschatz et al © 2018 updated by Khazaei @ 2021

14

<https://powcoder.com>

## Add WeChat powcoder

### EXAMPLE OF STANDARD API

As an example of a standard API, consider the `read()` function that is available in UNIX and Linux systems. The API for this function is obtained from the man page by invoking the command

`man read`

on the command line. A description of this API appears below:

```
#include <unistd.h>
ssize_t read(int fd, void *buf, size_t count)
```

A program that uses the `read()` function must include the `unistd.h` header file, as this file defines the `ssize_t` and `size_t` data types (among other things). The parameters passed to `read()` are as follows:

- `int fd`—the file descriptor to be read
- `void *buf`—a buffer into which the data will be read
- `size_t count`—the maximum number of bytes to be read into the buffer

On a successful read, the number of bytes read is returned. A return value of 0 indicates end of file. If an error occurs, `read()` returns -1.

Operating System Concepts – 10<sup>th</sup> Edition

2.15

Silberschatz et al © 2018 updated by Khazaei @ 2021

15

## System Call Implementation

- Typically, a number associated with each system call
  - **System-call interface** maintains a table indexed according to these numbers
- The system call interface invokes the intended system call in OS kernel and returns status of the system call and any return values
- The caller need know nothing about how the system call is implemented
  - Just needs to obey API and understand what OS will do as a result call
  - Most details of OS interface hidden from programmer by API
    - ▶ Managed by run-time support library (set of functions built into libraries included with compiler)

## Assignment Project Exam Help

Operating System Concepts – 10<sup>th</sup> Edition

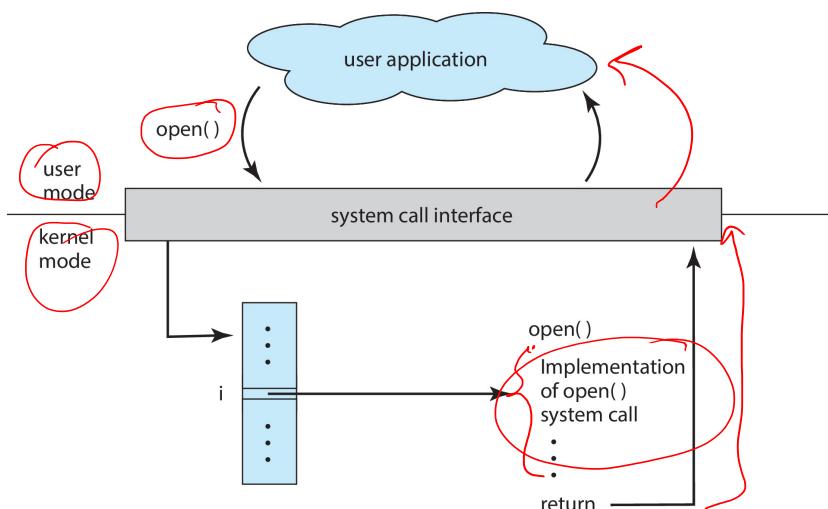
2.16

Silberschatz et al © 2018 updated by Khazaei @ 2021

16

<https://powcoder.com>

Add WeChat powcoder



Operating System Concepts – 10<sup>th</sup> Edition

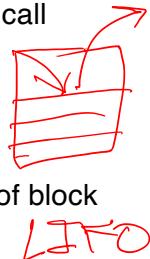
2.17

Silberschatz et al © 2018 updated by Khazaei @ 2021

17

## System Call Parameter Passing

- Often, more information is required than simply identity of desired system call
  - Exact type and amount of information vary according to OS and call
- Three general methods used to pass parameters to the OS
  - Simplest: pass the parameters in registers
    - ▶ In some cases, may be more parameters than registers
  - Parameters stored in a block, or table, in memory, and address of block passed as a parameter in a register
    - ▶ This approach taken by Linux and Solaris
  - Parameters placed, or **pushed**, onto the **stack** by the program and **popped** off the stack by the operating system
    - ▶ Block and stack methods do not limit the number or length of parameters being passed.



## Assignment Project Exam Help

Operating System Concepts – 10<sup>th</sup> Edition

2.18

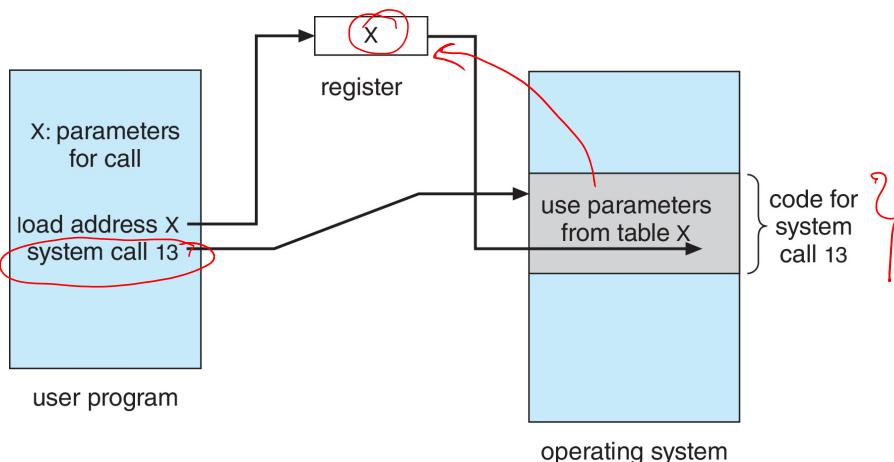
Silberschatz et al © 2018 updated by Khazaei @ 2021

18

<https://powcoder.com>

## Add WeChat powcoder

### Parameter Passing via Table



Operating System Concepts – 10<sup>th</sup> Edition

2.19

Silberschatz et al © 2018 updated by Khazaei @ 2021

19

## Types of System Calls

- Process control
    - create process, terminate process
    - end, abort
    - load, execute
    - get process attributes, set process attributes
    - wait for time
    - wait event, signal event
    - allocate and free memory
    - Dump memory if error
    - Debugger for determining bugs, single step execution
    - Locks for managing access to shared data between processes
- Assignment Project Exam Help**

Operating System Concepts – 10<sup>th</sup> Edition

2.20

Silberschatz et al © 2018 updated by Khazaei @ 2021

20

<https://powcoder.com>

## Add WeChat powcoder

### Types of System Calls (cont.)

- File management
  - create file, delete file
  - open, close file
  - read, write, reposition
  - get and set file attributes
- Device management
  - request device, release device
  - read, write, reposition
  - get device attributes, set device attributes
  - logically attach or detach devices

Operating System Concepts – 10<sup>th</sup> Edition

2.21

Silberschatz et al © 2018 updated by Khazaei @ 2021

21

10

## Types of System Calls (Cont.)

- Information maintenance
    - get time or date, set time or date
    - get system data, set system data
    - get and set process, file, or device attributes
  - Communications
    - create, delete communication connection
    - send, receive messages if **message passing model** to **host name** or **process name**
      - ▶ From **client** to **server**
    - **Shared-memory model** create and gain access to memory regions
    - transfer status information
    - attach and detach remote devices
- Ls -l | grep Vir*
- IPC*

Operating System Concepts – 10<sup>th</sup> Edition

2.22

Silberschatz et al © 2018 updated by Khazaei @ 2021

22

<https://powcoder.com>

## Add WeChat powcoder

- ### Types of System Calls (Cont.)
- Protection
    - Control access to resources
    - Get and set permissions
    - Allow and deny user access

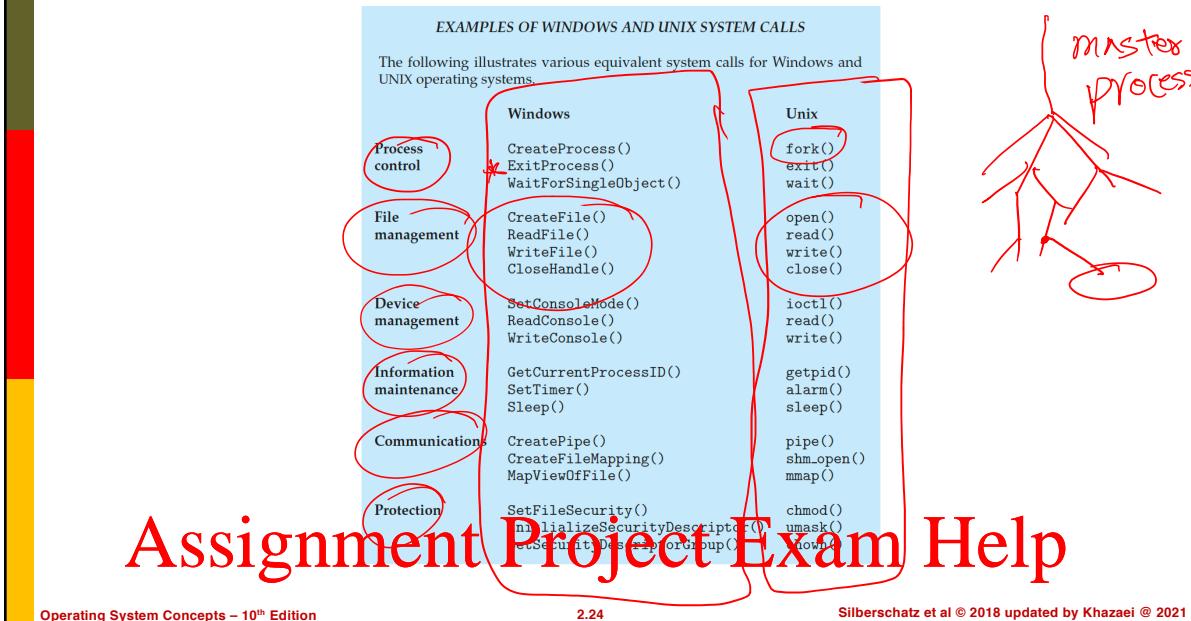
Operating System Concepts – 10<sup>th</sup> Edition

2.23

Silberschatz et al © 2018 updated by Khazaei @ 2021

23

## Examples of Windows and Unix System Calls

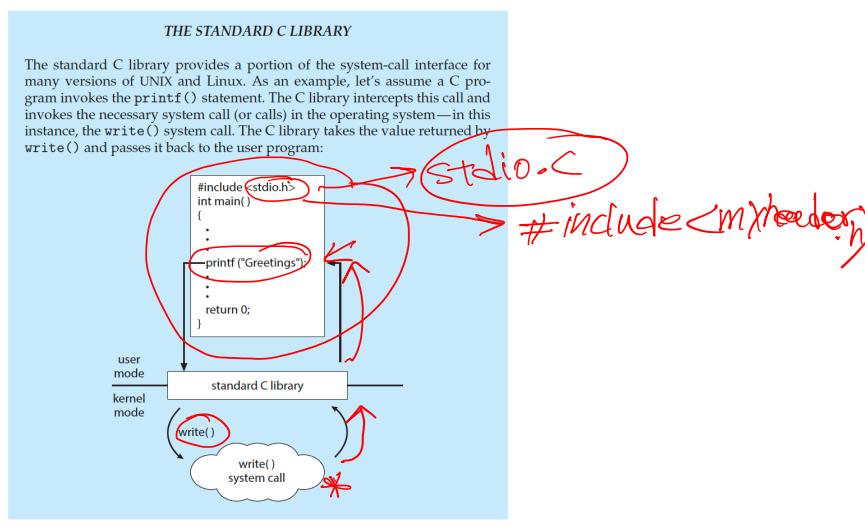


24

<https://powcoder.com>

## Add WeChat powcoder

- C program invoking printf() library call, which calls write() system call



25

## System Services

- System programs provide a convenient environment for program development and execution. They can be divided into:

- File manipulation
- Status information sometimes stored in a file
- Programming language support
- Program loading and execution
- Communications
- Background services
- Application programs

- Most users' view of the operation system is defined by **system programs**, not the actual system calls

Assignment Project Exam Help

<https://powcoder.com>

## Add WeChat powcoder System Services (cont.)

- Provide a convenient environment for program development and execution
  - Some of them are simply user interfaces to system calls; others are considerably more complex
- **File management** - Create, delete, copy, rename, print, dump, list, and generally manipulate files and directories
- **Status information**
  - Some ask the system for info - date, time, amount of available memory, disk space, number of users
  - Others provide detailed performance, logging, and debugging information
  - Typically, these programs format and print the output to the terminal or other output devices
  - Some systems implement a **registry** - used to store and retrieve configuration information

## System Services (Cont.)

### ■ File modification

- Text editors to create and modify files
- Special commands to search contents of files or perform transformations of the text

### ■ Programming-language support - Compilers, assemblers, debuggers and interpreters sometimes provided

### ■ Program loading and execution- Absolute loaders, relocatable loaders, linkage editors, and overlay-loaders, debugging systems for higher-level and machine language

### ■ Communications - Provide the mechanism for creating virtual connections among processes, users, and computer systems

- Allow users to send messages to one another's screens, browse web pages, send electronic-mail messages, log in remotely, transfer files from one machine to another.

Assignment Project Exam Help

<https://powcoder.com>

## Add WeChat powcoder

### ■ Background Services

- Launch at boot time
  - ▶ Some for system startup, then terminate
  - ▶ Some from system boot to shutdown
- Provide facilities like disk checking, process scheduling, error logging, printing
- Run in user context not kernel context
- Known as **services, subsystems, daemons**

### ■ Application programs

- Don't pertain to system
- Run by users
- Not typically considered part of OS
- Launched by command line, mouse click, finger poke

## Linkers and Loaders

- Source code compiled into object files designed to be loaded into any physical memory location – **relocatable object file**
- **Linker** combines these into single binary **executable** file
  - Also brings in libraries
- Program resides on secondary storage as binary executable
- Must be brought into memory by **loader** to be executed
  - **Relocation** assigns final addresses to program parts and adjusts code and data in program to match those addresses
- Modern general purpose systems don't link libraries into executables
  - Rather, **dynamically linked libraries** (in Windows, **DLLs**) are loaded as needed, shared by all that use the same version of that same library (loaded once)
- Object, executable files have standard formats, so operating system knows how to load and start them

## Assignment Project Exam Help

Operating System Concepts – 10<sup>th</sup> Edition

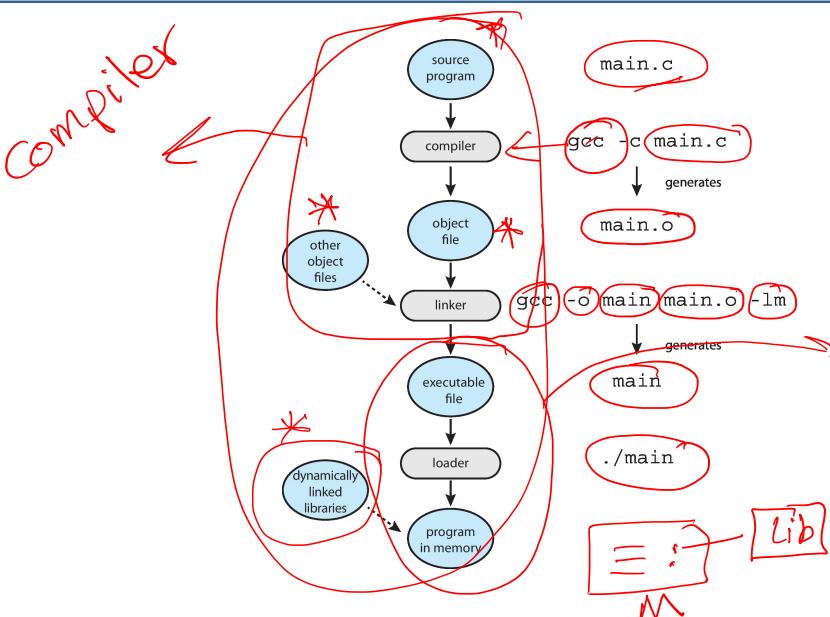
2.32

Silberschatz et al © 2018 updated by Khazaei @ 2021

32

<https://powcoder.com>

## Add WeChat powcoder The Role of the Linker and Loader



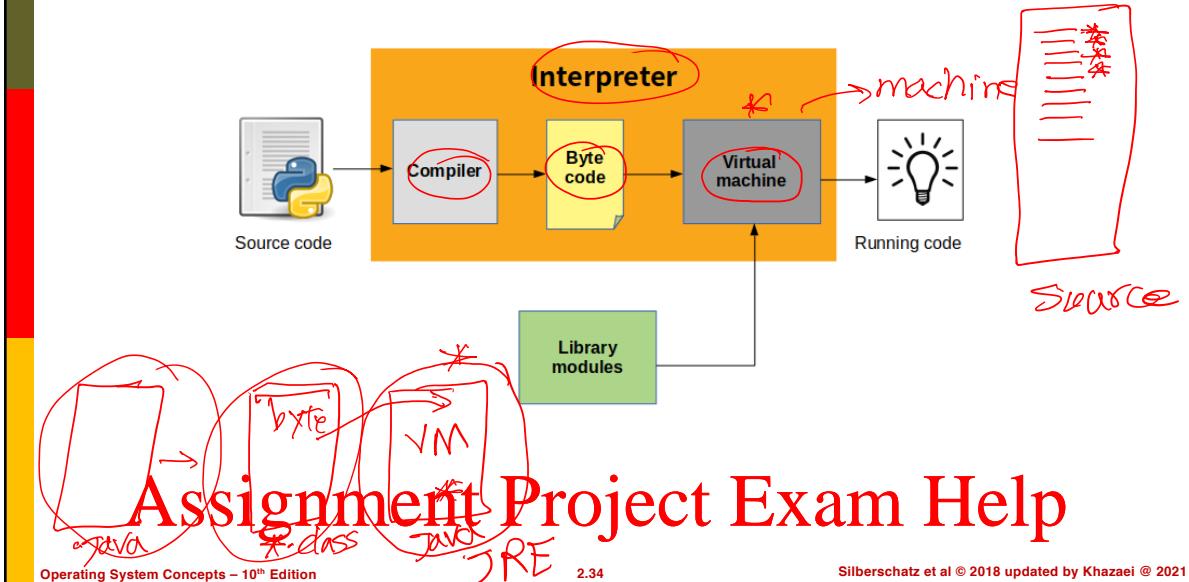
Operating System Concepts – 10<sup>th</sup> Edition

2.33

Silberschatz et al © 2018 updated by Khazaei @ 2021

33

# Python Interpreter



34

<https://powcoder.com>

Add WeChat powcoder  
Why Applications are Operating System Specific

# Add WeChat powcoder

## Why Applications are Operating System Specific

- Apps compiled on one system usually not executable on other operating systems
  - Each operating system provides its own unique system calls *\*Up to here.*
    - Own file formats, etc
  - Apps can be multi-operating system
    - Written in interpreted language like Python, Ruby, and interpreter available on multiple operating systems
    - App written in language that includes a VM containing the running app (like Java)
    - Use standard language (like C), compile separately on each operating system to run on each
  - **Application Binary Interface (ABI)** is architecture equivalent of API, defines how different components of binary code can interface for a given operating system on a given architecture, CPU, etc

## Operating System Design and Implementation

- Design and Implementation of OS not “solvable”, but some approaches have proven successful
- Internal structure of different Operating Systems can vary widely
- Start the design by defining goals and specifications
- Affected by choice of hardware, type of system
- **User** goals and **System** goals
  - User goals – operating system should be convenient to use, easy to learn, reliable, safe, and fast
  - System goals – operating system should be easy to design, implement, and maintain, as well as flexible, reliable, error-free, and efficient

## Assignment Project Exam Help

Operating System Concepts – 10<sup>th</sup> Edition

2.36

Silberschatz et al © 2018 updated by Khazaei @ 2021

36

<https://powcoder.com>

## Add WeChat powcoder Operating System Design and Implementation (Cont.)

- Important principle to separate
  - Policy:** *What* will be done?
  - Mechanism:** *How* to do it?
- Mechanisms determine how to do something; policies decide what will be done
- The separation of policy from mechanism is a very important principle, it allows maximum flexibility if policy decisions are to be changed later (example – timer)
- Specifying and designing an OS is highly creative task of **software engineering**

Operating System Concepts – 10<sup>th</sup> Edition

2.37

Silberschatz et al © 2018 updated by Khazaei @ 2021

37

17

## Implementation

- Much variation
  - Early OSes in assembly language
  - Then system programming languages like Algol, PL/I
  - Now C, C++
- Usually a mix of languages
  - Lowest levels in assembly
  - Main body in C
  - Systems programs in C, C++, scripting languages like PERL, Python, shell scripts
- More high-level language easier to **port** to other hardware
  - But slower
- ~~Emulation can allow an OS to run on non-native hardware~~

<https://powcoder.com>

## Add WeChat powcoder Operating System Structure

- General-purpose OS is very large program
- Various ways to structure ones
  - Simple structure – MS-DOS
  - More complex – UNIX
  - Layered – an abstraction
  - Microkernel – Mach

## Monolithic Structure – Original UNIX

- UNIX – limited by hardware functionality, the original UNIX operating system had limited structuring. The UNIX OS consists of two separable parts
  - Systems programs
  - The kernel
    - Consists of everything below the system-call interface and above the physical hardware
    - Provides the file system, CPU scheduling, memory management, and other operating-system functions; many functions for one level

## Assignment Project Exam Help

Operating System Concepts – 10<sup>th</sup> Edition

2.40

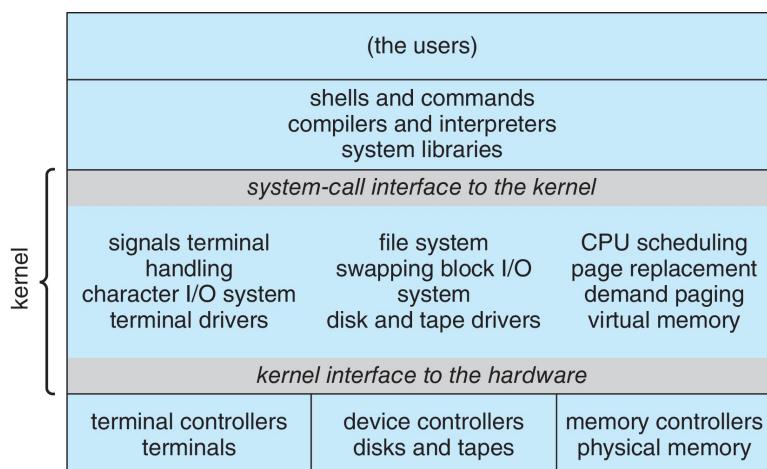
Silberschatz et al © 2018 updated by Khazaei @ 2021

40

<https://powcoder.com>

## Add WeChat powcoder

Beyond simple but not fully layered



Operating System Concepts – 10<sup>th</sup> Edition

2.41

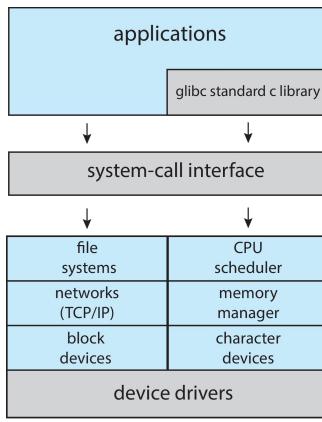
Silberschatz et al © 2018 updated by Khazaei @ 2021

41

19

## Linux System Structure

Monolithic plus modular design



Assignment Project Exam Help

Operating System Concepts – 10<sup>th</sup> Edition

2.42

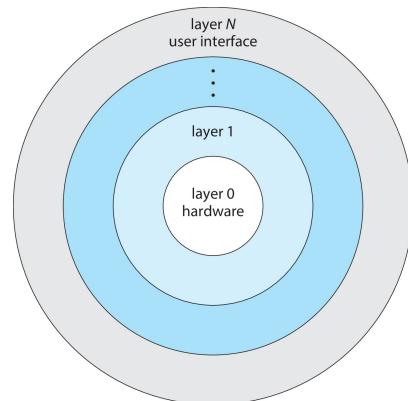
Silberschatz et al © 2018 updated by Khazaei @ 2021

42

<https://powcoder.com>

## Add WeChat powcoder

- The operating system is divided into several layers (levels), each built on top of lower layers. The bottom layer (layer 0), is the hardware; the highest (layer N) is the user interface.
- With modularity, layers are selected such that each uses functions (operations) and services of only lower-level layers



Operating System Concepts – 10<sup>th</sup> Edition

2.43

Silberschatz et al © 2018 updated by Khazaei @ 2021

43

## Microkernels

- Moves as much from the kernel into user space
- **Mach** example of **microkernel**
  - Mac OS X kernel (**Darwin**) partly based on Mach
- Communication takes place between user modules using **message passing**
- Benefits:
  - Easier to extend a microkernel
  - Easier to port the operating system to new architectures
  - More reliable (less code is running in kernel mode)
  - More secure
- Detriments:
  - Performance overhead of user space to kernel space communication

**Assignment Project Exam Help**

Operating System Concepts – 10<sup>th</sup> Edition

2.44

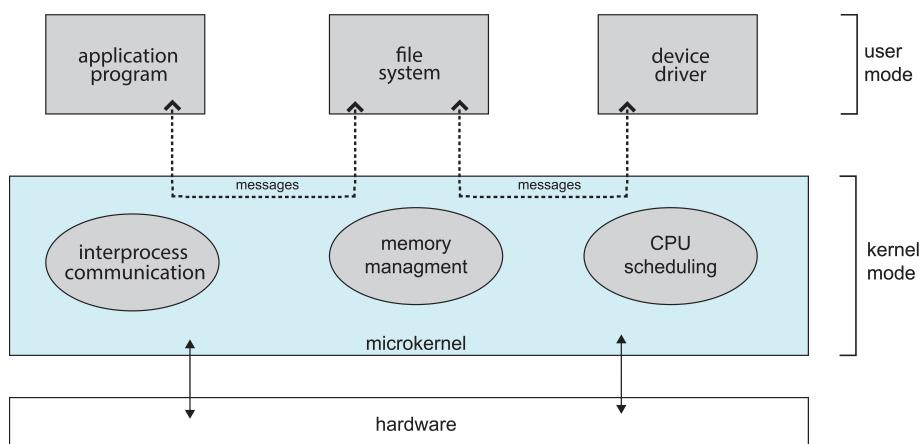
Silberschatz et al © 2018 updated by Khazaei @ 2021

44

<https://powcoder.com>

## Add WeChat powcoder

### Microkernel System Structure



Operating System Concepts – 10<sup>th</sup> Edition

2.45

Silberschatz et al © 2018 updated by Khazaei @ 2021

45

## Modules

- Many modern operating systems implement **loadable kernel modules (LKMs)**
  - Uses object-oriented approach
  - Each core component is separate
  - Each talks to the others over known interfaces
  - Each is loadable as needed within the kernel
  
- Overall, similar to layers but with more flexibility
  - Linux, Solaris, etc

## Assignment Project Exam Help

Operating System Concepts – 10<sup>th</sup> Edition

2.46

Silberschatz et al © 2018 updated by Khazaei @ 2021

46

<https://powcoder.com>

## Add WeChat powcoder Hybrid Systems

- Most modern operating systems are actually not one pure model
  - Hybrid combines multiple approaches to address performance, security, usability needs
  - Linux and Solaris kernels in kernel address space, so monolithic, plus modular for dynamic loading of functionality
  - Windows mostly monolithic, plus microkernel for different subsystem **personalities**
  
- Apple Mac OS X hybrid, layered, **Aqua** UI plus **Cocoa** programming environment
  - Below is kernel consisting of Mach microkernel and BSD Unix parts, plus I/O kit and dynamically loadable modules (called **kernel extensions**)

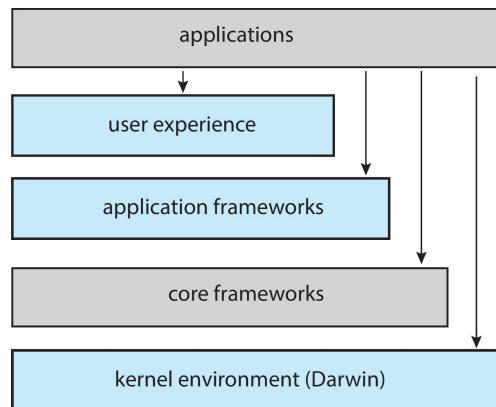
Operating System Concepts – 10<sup>th</sup> Edition

2.47

Silberschatz et al © 2018 updated by Khazaei @ 2021

47

## macOS and iOS Structure



## Assignment Project Exam Help

Operating System Concepts – 10<sup>th</sup> Edition

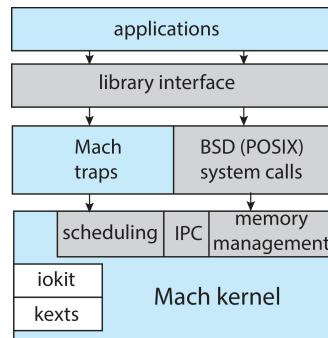
2.48

Silberschatz et al © 2018 updated by Khazaei @ 2021

48

<https://powcoder.com>

Add WeChat  
Darwin



Operating System Concepts – 10<sup>th</sup> Edition

2.49

Silberschatz et al © 2018 updated by Khazaei @ 2021

49

# iOS

- Apple mobile OS for *iPhone, iPad*

- Structured on Mac OS X, added functionality
- Does not run OS X applications natively
  - ▶ Also runs on different CPU architecture (ARM vs. Intel)
- **Cocoa Touch** Objective-C API for developing apps
- **Media services** layer for graphics, audio, video
- **Core services** provides cloud computing, databases
- Core operating system, based on Mac OS X kernel

Cocoa Touch

Media Services

Core Services

Core OS

## Assignment Project Exam Help

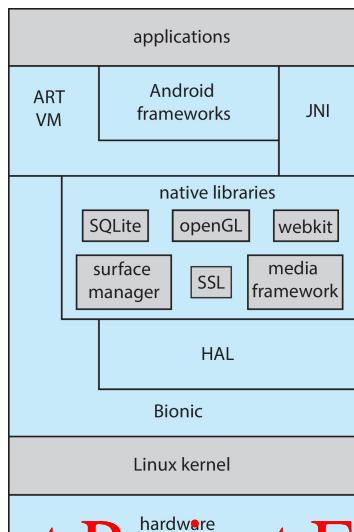
<https://powcoder.com>

# Add WeChat, powcoder

Android

- Developed by Open Handset Alliance (mostly Google)
  - Open Source
- Similar stack to IOS
- Based on Linux kernel but modified
  - Provides process, memory, device-driver management
  - Adds power management
- Runtime environment includes core set of libraries and Dalvik virtual machine
  - Apps developed in Java plus Android API
    - ▶ Java class files compiled to Java bytecode then translated to executable than runs in Dalvik VM
- Libraries include frameworks for web browser (webkit), database (SQLite), multimedia, smaller libc

## Android Architecture



Assignment Project Exam Help

<https://powcoder.com>

## Add WeChat powcoder Building and Booting Linux

- Download Linux source code (<http://www.kernel.org>)
- Configure kernel via “make menuconfig”
- Compile the kernel using “make”
  - Produces vmlinuz, the kernel image
  - Compile kernel modules via “make modules”
  - Install kernel modules into vmlinuz via “make modules\_install”
  - Install new kernel on the system via “make install”
- **Optional Exercise:** try to build a new kernel on your VM and install it?

## Operating-System Debugging

- Debugging is finding and fixing errors, or bugs
- Also performance tuning
- OS generate log files containing error information
- Failure of an application can generate core dump file capturing memory of the process
- Operating system failure can generate crash dump file containing kernel memory
- Beyond crashes, performance tuning can optimize system performance
  - Sometimes using **trace listings** of activities, recorded for analysis
  - **Profiling** is periodic sampling of instruction pointer to look for statistical trends

**Kernighan's Law:** "Debugging is twice as hard as writing the code in the first place. Therefore, if you write the code as cleverly as possible, you are, by definition, not smart enough to debug it."

## Assignment Project Exam Help

Operating System Concepts – 10<sup>th</sup> Edition

2.56

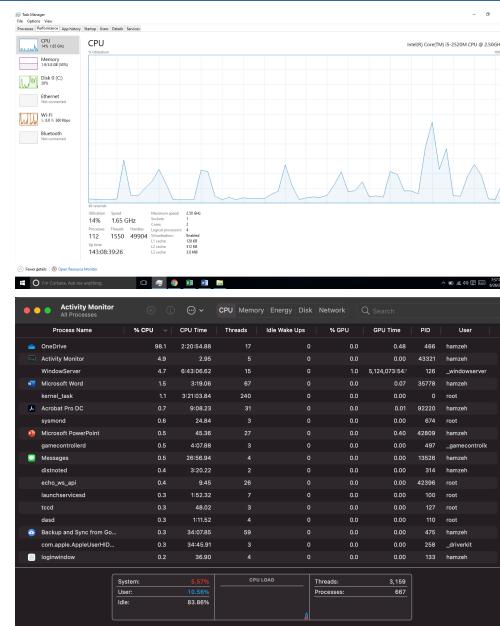
Silberschatz et al © 2018 updated by Khazaei @ 2021

56

<https://powcoder.com>

## Add WeChatpowcoder

- Improve performance by removing bottlenecks
- OS must provide means of computing and displaying measures of system behavior
- For example, “top” program or Windows Task Manager, Activity Monitor for Mac



Operating System Concepts – 10<sup>th</sup> Edition

2.57

Silberschatz et al © 2018 updated by Khazaei @ 2021

57

## Tracing

- Collects data for a specific event, such as steps involved in a system call invocation
- Tools include
  - strace – trace system calls invoked by a process
  - gdb – source-level debugger
  - perf – collection of Linux performance tools
  - tcpdump – collects network packets

## Assignment Project Exam Help

Operating System Concepts – 10<sup>th</sup> Edition

2.58

Silberschatz et al © 2018 updated by Khazaei @ 2021

58

<https://powcoder.com>

## Add WeChat powcoder

- Debugging interactions between user-level and kernel code nearly impossible without toolset that understands both and can instrument their actions
- BCC (BPF Compiler Collection) is a rich toolkit providing tracing features for Linux
  - See also the original DTrace
- For example, disksnoop.py traces disk I/O activity

TIME(s)	T	BYTES	LAT(ms)
1946.29186700	R	8	0.27
1946.33965000	R	8	0.26
1948.34585000	W	8192	0.96
1950.43251000	R	4096	0.56
1951.74121000	R	4096	0.35

- Many other tools (next slide)

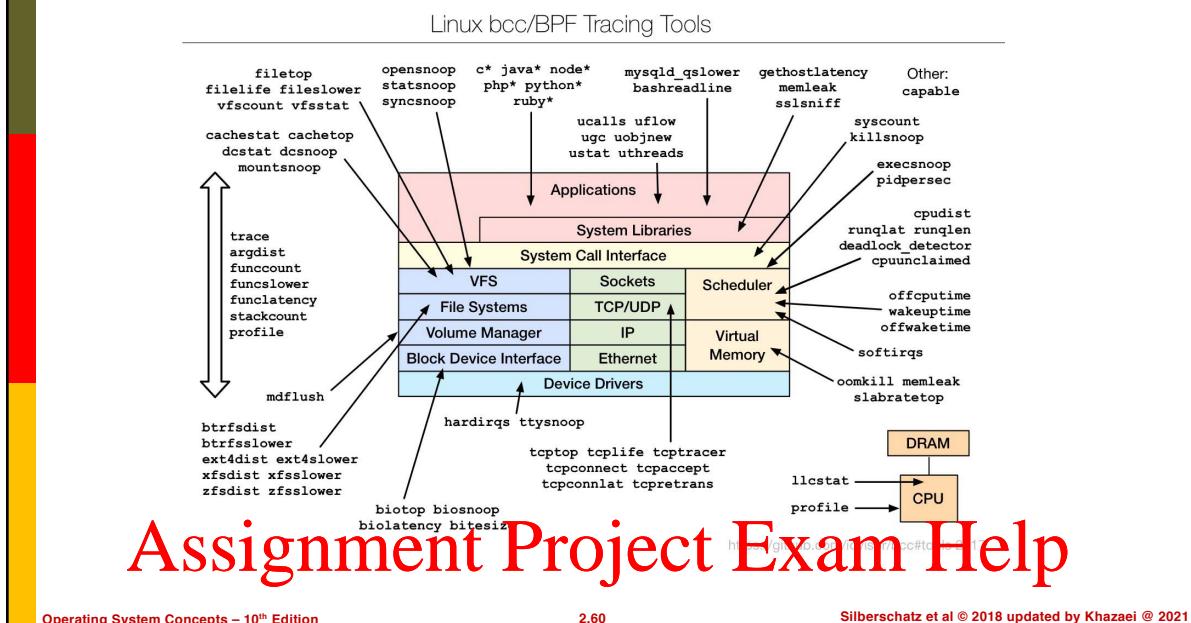
Operating System Concepts – 10<sup>th</sup> Edition

2.59

Silberschatz et al © 2018 updated by Khazaei @ 2021

59

## Linux bcc/BPF Tracing Tools



60

<https://powcoder.com>

Add WeChat powcoder  
End of Chapter 2

Any Question?

