Shell Interactivity and SSH Basics

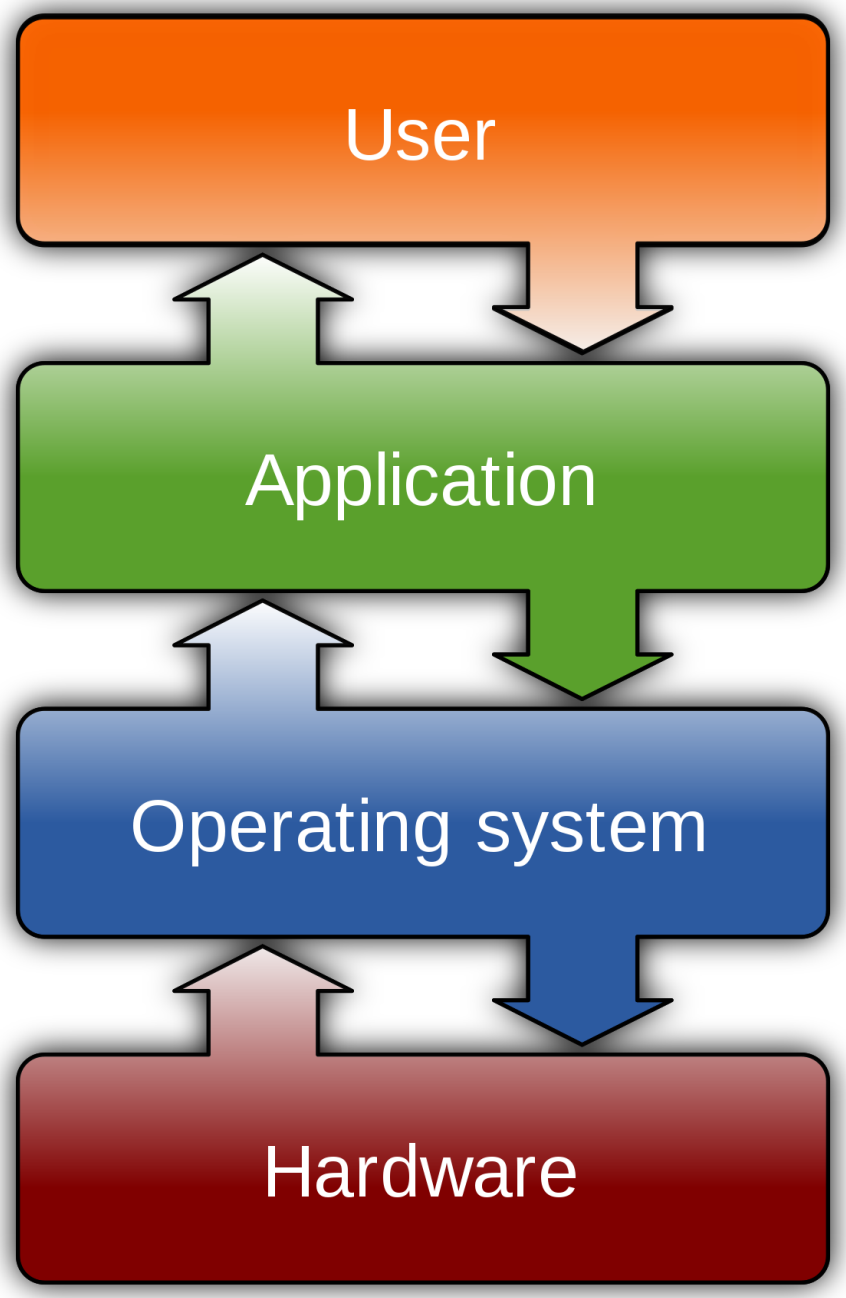
Computer Science Practice and Experience: Development Basics
CS1XC3

Professor: Kevin Browne
E-mail: brownek@mcmaster.ca

Operating systems

- An **operating system** manages computer hardware and software resources and provides access to services for **applications (i.e. computer programs)**
  - Examples include Windows, macOS, Linux
  - Specialized operating systems exist for servers (e.g. Windows Server), and mobile (e.g. iOS)

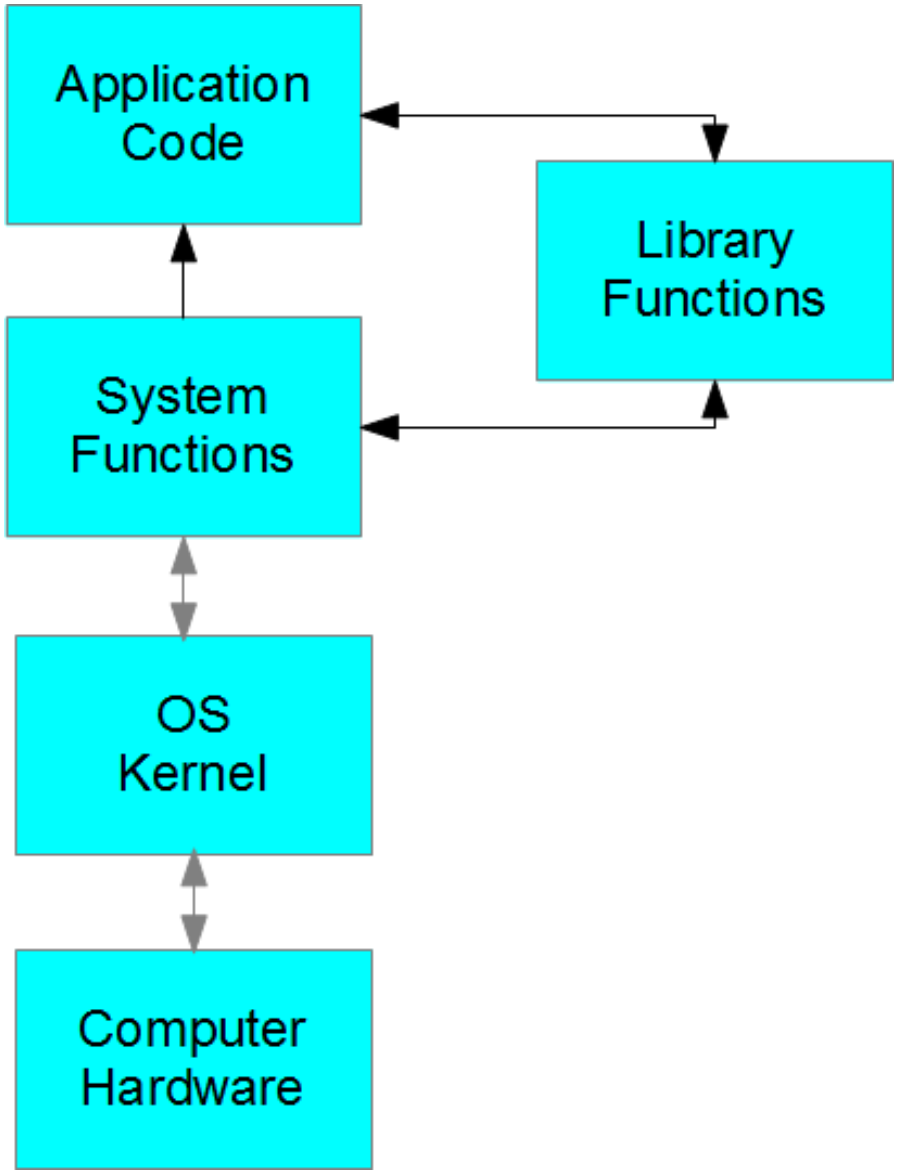
- Services an OS may provide include things like:
  - File access
  - Interprocess communication
  - Network access



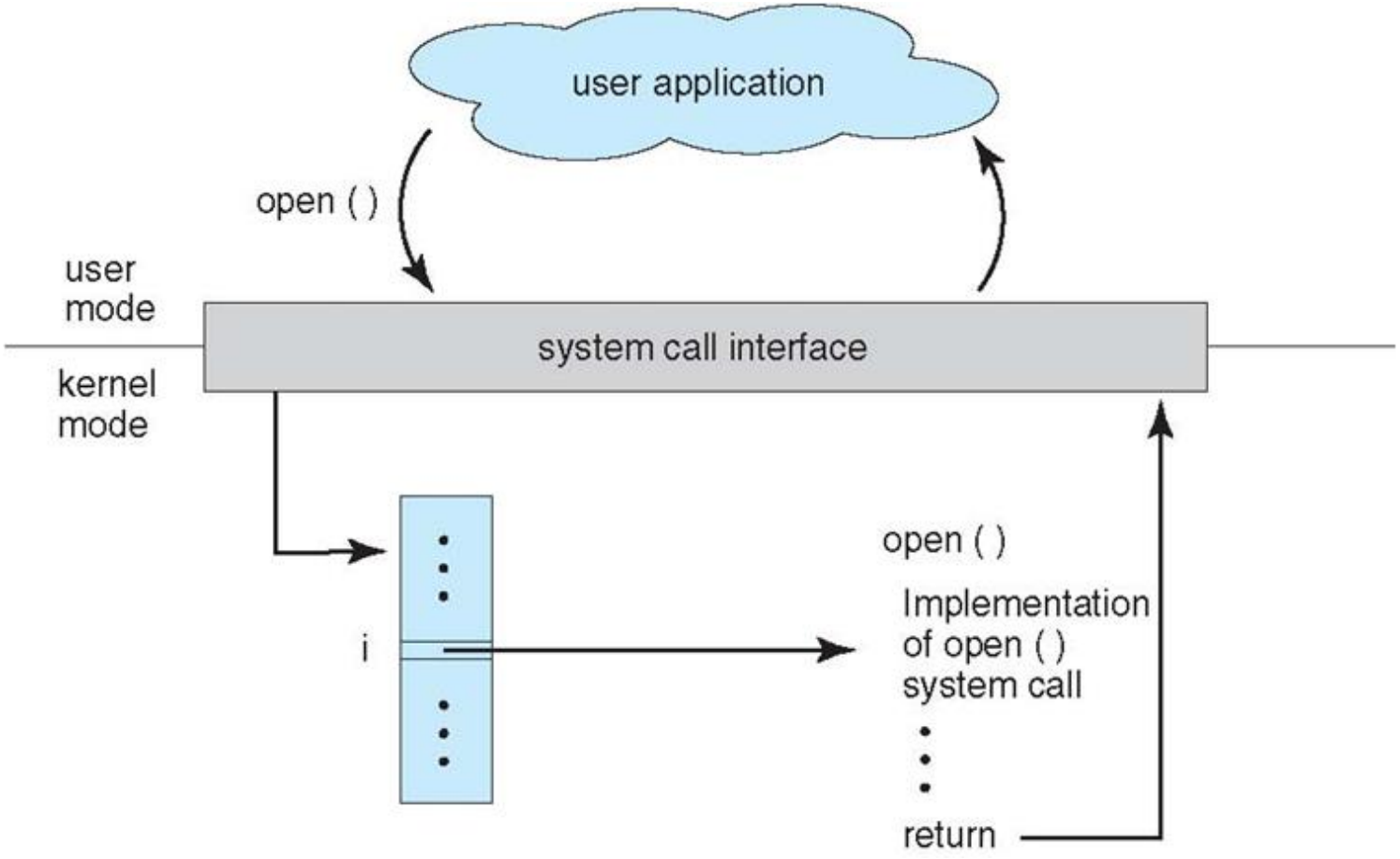
Source: https://upload.wikimedia.org/wikipedia/commons/thumb/e/e1/Operating_system_placement.svg/1200px-Operating_system_placement.svg.png

Operating systems

- Operating systems include a **kernel** which manages access to memory, CPU time, and other resources
  - The kernel is the core of the operating system, all operating systems include a kernel
- Kernels provide an *application programming interface* (API) of services used by OS services and applications
  - e.g. output information to the screen, input information from devices
- These API services are accessed via **system calls**
  - When a system call executes, the kernel takes over to carry out the functionality of the system call
  - user mode -> kernel mode -> user mode transition



Source: https://static.thegeekstuff.com/wp-content/uploads/2012/06/system-library-call.png



Source: https://i.stack.imgur.com/tGPV0.png

Shells

- A shell is an interface for the services of an OS
  - Typically used by humans, but can be an interface for programs too
  - e.g. a shell can be used to run an application, search for files, etc.
- Shells can have different types of interfaces
  - A graphical user interface (GUI) is one type of shell
  - A command-line interface (CLI) is another type of shell
- Graphical user interfaces are easier for users to learn
  - Much better for non-technical users (e.g. running productivity software, games, etc.)

Source: https://tr3.cbsistatic.com/hub/i/r/2016/12/08/7f9849d9-1fc5-4140-8c64-3ea7ec4d28ec/resize/1200x/7b824af44d90ad0e330154819d7793ac/screenshot-from-2016-12-08-143410.png

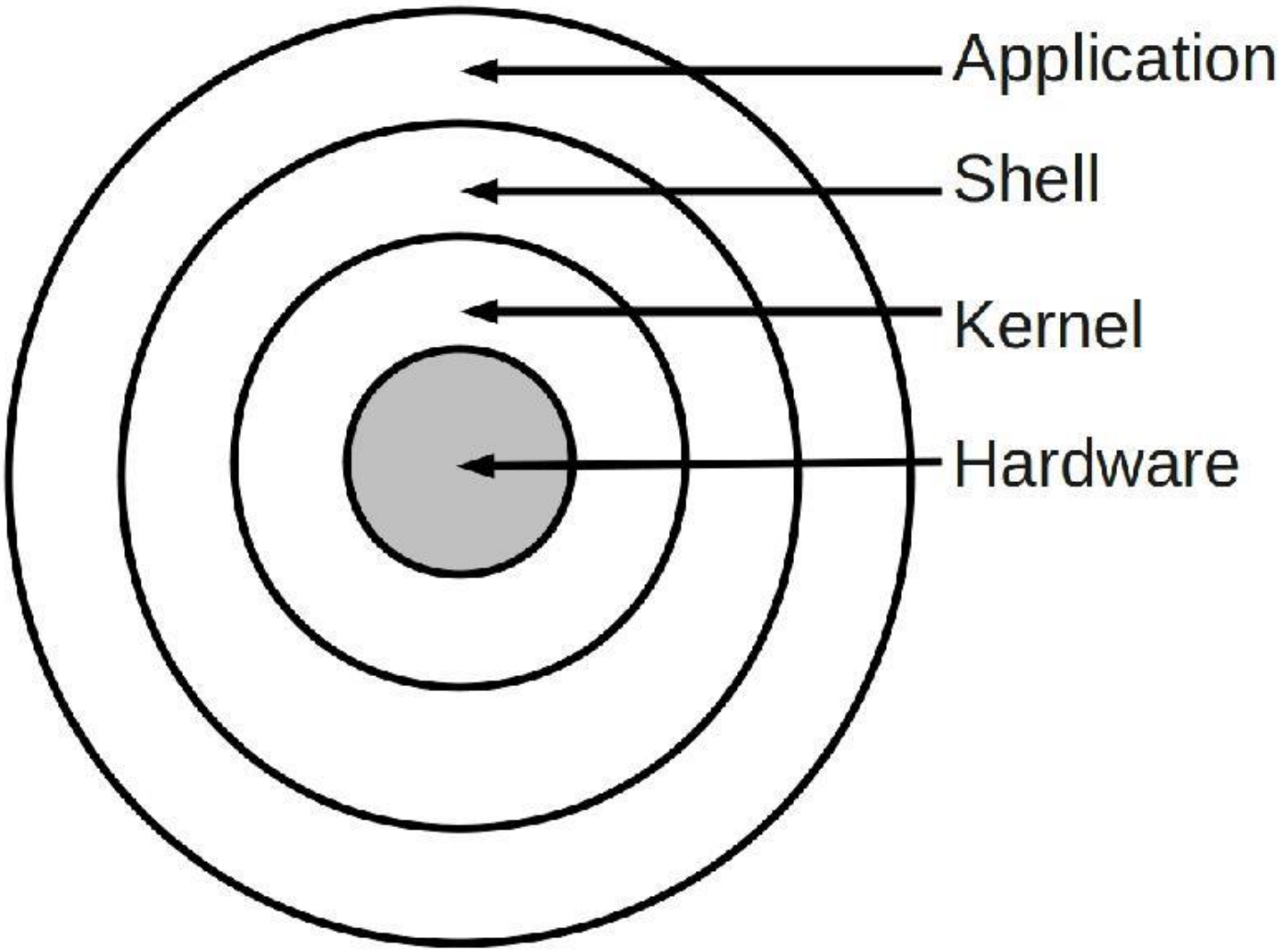
```
chris@ubuntu: ~
chris@ubuntu:~$ bash --version
GNU bash, version 4.3.46(1)-release (x86_64-pc-linux-gnu)
Copyright (C) 2013 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.

This is free software; you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
chris@ubuntu:~$
```

Source: https://www.howtogeek.com/thumbcache/2/200/81bf2c19ed9586ba20862f5dbbbd16ad/wp-content/uploads/2017/03/img_58c0939c2d487.png

Shells

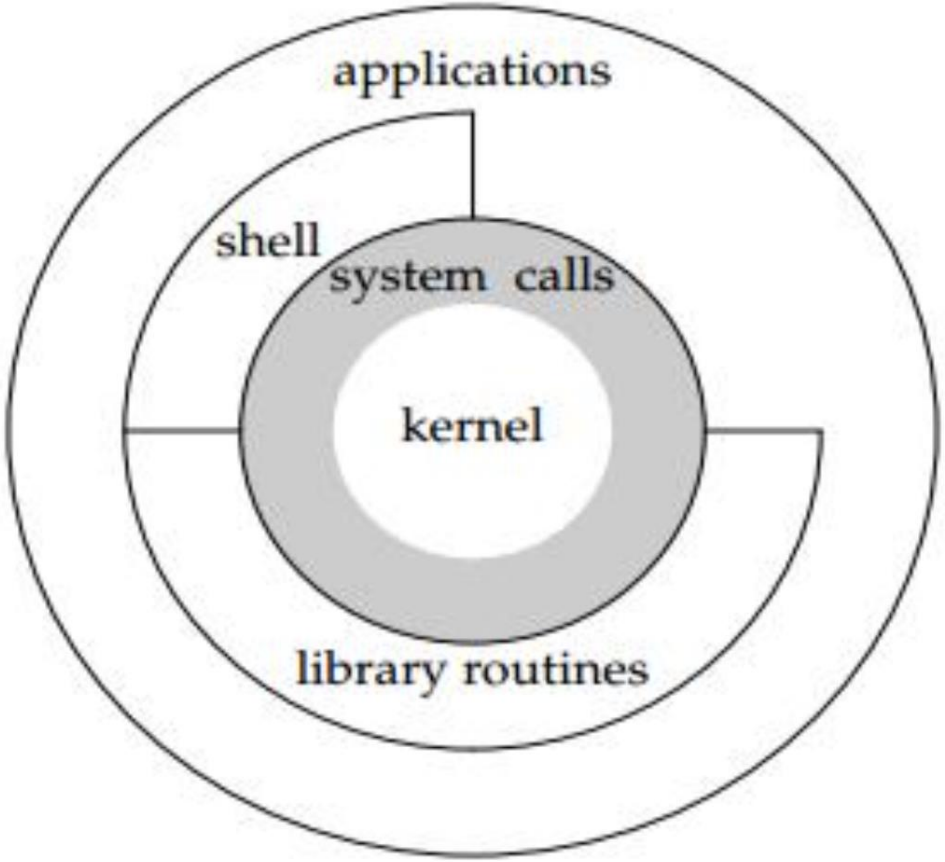
- Command-line interfaces are more powerful, precise, and flexible... very developer-friendly traits
  - e.g. we can run commands with precise sets of options
    - "Find all hidden files that have the file extension png, including all subfolders"
  - We can use the output of one command as the input to another command
  - We can use **shell scripting** which involves running a series of commands in a file... essentially programs of shell commands

- While the above might be ***technically possible*** with GUIs, it's slower and less programmatic / automatable



Source: https://cis2.oc.ctc.edu/oc_apps/Westlund/xbook/111_u04e/OS_Layers.jpg

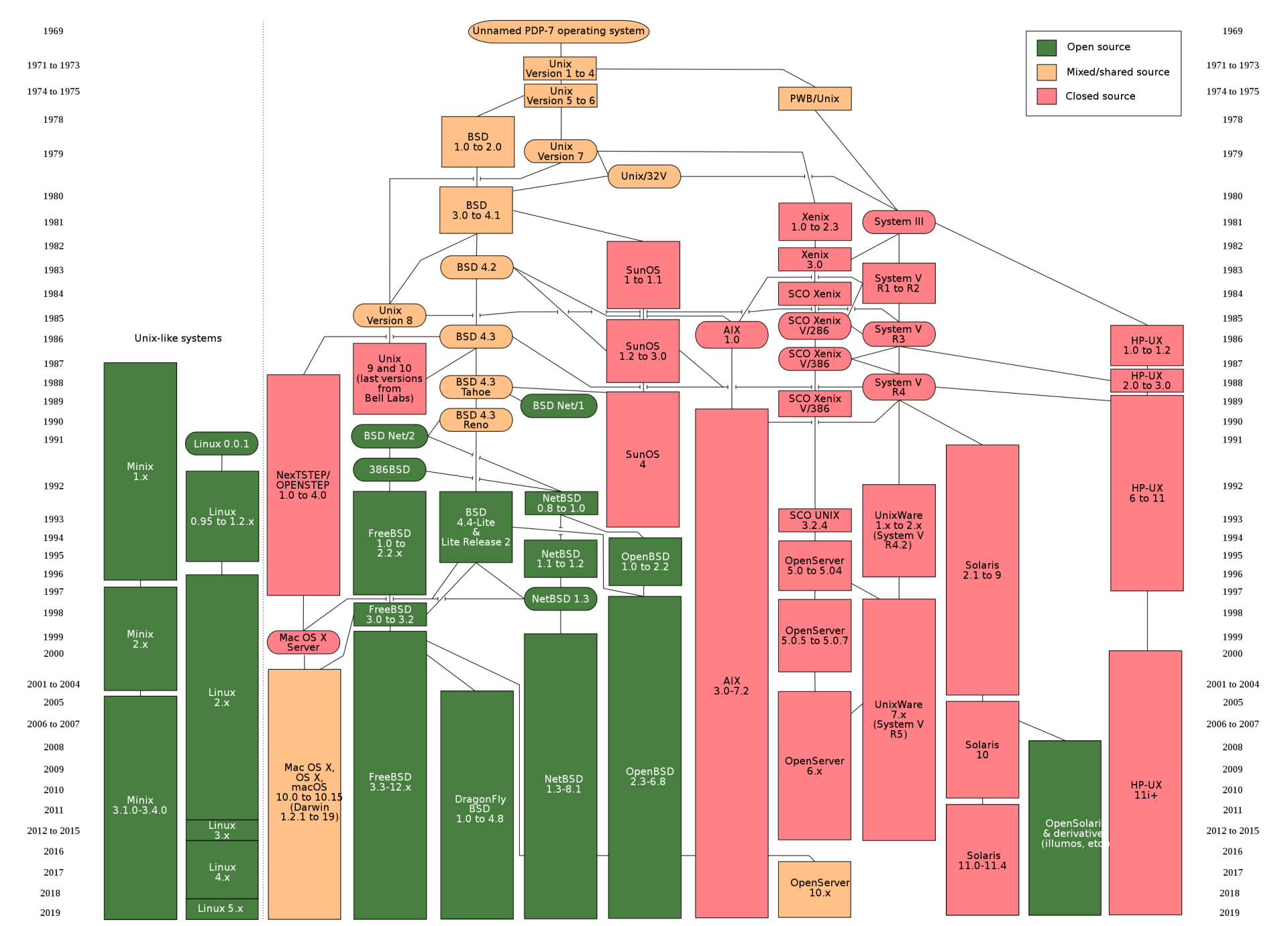
Shell and applications both use system calls... so while we can think of applications as being another layer, they also share a layer with the shell in accessing the kernel via system calls...



Source: https://www.programmersought.com/images/741/ed36f9661e90f05e83a86c85eb750545.png

Unix-like systems

- Unix is a family of related operating systems that conform to a specification
- Unix-like systems include systems *similar to* Unix such as Linux and macOS
- The term *Unix-like* is informal, not technical/precise
- But Unix-like systems will share a similar modular design and characteristics, such as shells



Source: https://en.wikipedia.org/wiki/Unix-like#/media/File:Unix_history-simple.svg

Linux

- Linux is itself a family of operating systems which all share the Linux kernel
- Many distributions of Linux exist, for example: Ubuntu, Fedora, CentOS, Debian, Gentoo
- Linux is famously **open source software**
  - Open source software allows users to view and edit source code
  - Different licenses exist that allow for different forms of sharing modified versions of the software

Fun fact: Ancaster-native (town located 5 mins from McMaster) Bob Young co-founded Linux tech giant Red Hat... named after the red hats he would wear at the time...

Source: https://i2.wp.com/dennis-blog.appelon.net/wp-content/uploads/2008/11/Why-Red-Hat_only_pic.jpg?fit=474%2C382&ssl=1

Bash shell

- Bash shell is a (very) common shell for Unix-like systems
- Bash availability...
  - It is available on macOS systems (open a terminal, enter the command "bash")
  - It is the default shell on most Linux distributions
  - It's even available on ***Windows 10*** now via the Windows Subsystem for Linux:
    - https://en.wikipedia.org/wiki/Windows_Subsystem_for_Linux
- Other shells exist... Z shell (zsh), Korn shell (ksh), etc.
  - Bash command syntax is actually a superset of the Bourne shell... people conflate the two as a result

Commands

- A large set of commands are supported by shells in all Unix systems and virtually all Unix-like systems
  - https://en.wikipedia.org/wiki/List_of_Unix_commands
- Examples:
  - **cd** – change the working directory
  - **ls** – list directory contents
  - **mkdir** – make directory
- We'll get into more advanced shell features like scripting and pipes/filters soon enough!
  - But we'll start with some basic interactivity first...

Let's try out a Bash shell!

Some commands

- **cat –** outputs content of the file
- **cd** – change the working directory
- **cp –** copy a file
- **ls** – list directory contents
- **man –** show a command's man page (i.e. manual)
- **mkdir** – make directory
- **mv –** move a file
- **ps –** lists all processes
- **pwd –** outputs current working directory
- **rm** – removes a file
- **rmdir –** removes a directory if it is empty
- **grep –** searches file contents

Commands

- If you're ever unsure how a command works you can always run man to bring up instructions, but be mindful the instructions will be lengthy and not beginner-friendly:

  man *commandName*

- Commands can have ***arguments***
  - cp source_file target_file

- Commands can have ***option flags*** that modify the behaviour of the command
  - cp -R /home/kevin/new /home/kevin/old

- We'll go over specific commands in more detail in lab next week!

SSH (Secure Shell)

- The **Secure Shell (SSH) Protocol** is a network protocol for secure remote login and other secure network services over an insecure network
  - A network protocol is a set of rules that allow computers on a network to transmit information to each other

- SSH is most typically used to access a shell on a *remote machine* from your *local machine*
  - i.e. access *another computer's* shell from *your computer*
  - But SSH has other issues such as file transfer via SCP, SFTP, technologies we'll talk about soon

SSH Client

- An **SSH client** allows us to access a remote machine's shell using SSH

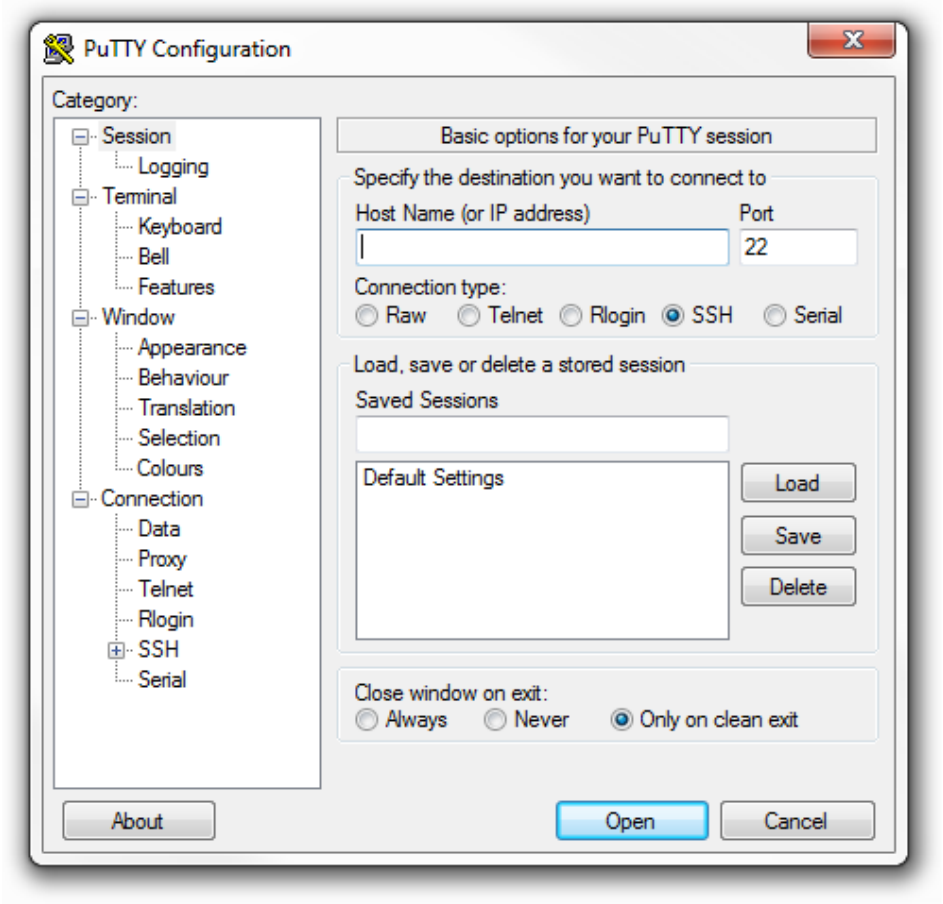
- OpenSSH comes with macOS and Linux
  - An open source command-line tool
  - https://en.wikipedia.org/wiki/OpenSSH
  - It's now being included with newer versions of Windows

- PuTTY is a popular SSH client for Windows
  - https://www.putty.org/

OpenSSH usage

- Basic usage: ssh username@remote.machine.com
- Where username is an account on the remote machine
- And remote.machine.com is the hostname or IP address for the remote machine
  - An IP address is a numerical address for identifying machines on a network (such as the Internet)
  - A hostname is a more human-friendly identifier, for example something like server.mcmaster.ca

PuTTY has a GUI interface for entering hostname and username information…



Source: https://kbimages.dreamhosters.com/images/01_Putty_config.png

moore server

- Every student in this course should have access to an account (and bash shell) on the moore server maintained by the department

- To access your account:
  - Visit this URL and enter your MacID (e.g. smithj) and your MacID password when asked: https://www.cas.mcmaster.ca/macid
  - This will create an account for you on Moore with a username set to your MacID and your MacID password
  - You can then login using ssh:
    - e.g. ssh yourmacid@moore.mcmaster.ca

Let's try out an SSH client!

Try accessing the Moore server

- Try accessing your Moore server account via an SSH client
- Follow the steps in the previous slide
  - i.e. make sure you visit that URL and enter your username and password
- You'll need access to a bash shell for the first assignment... Moore is an easy way to get access to one
  - Everyone should be able to access Moore throughout the course
  - We can also run C programs on Moore and other tools throughout the course...