

Computer Science Fundamentals

Hovag Abramian



Session II

Outline

We are going to learn about:

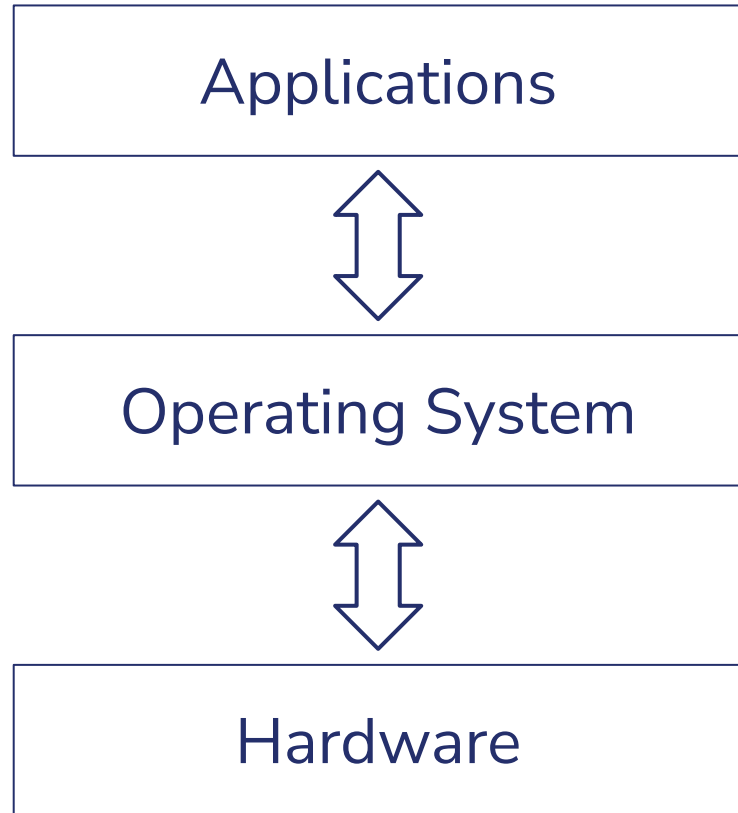
- Operating Systems
- Processes
- Networking

Learning Objectives

At the end of the session, you will be able to:

- Enumerate the responsibilities of operating systems
- Identify running processes
- Classify network protocols

Operating Systems, Responsibilities



- Abstraction
- Resource Management
- Multitasking
- Security
- User interface

Abstraction

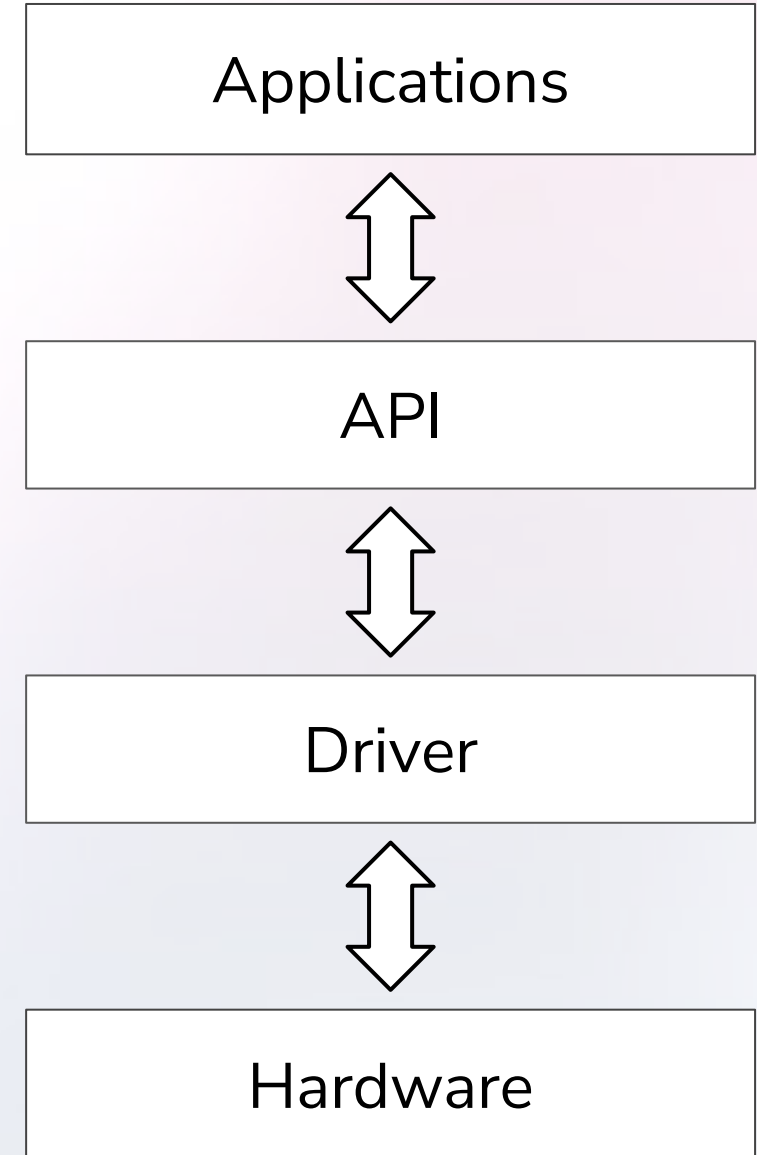
Accounting application that helps you calculate income tax, needs to save and print documents.

vs.

“Accounting” for the correct voltage through the right pin that regulates the motion mechanism for paper, for every printer model ever built.

Abstraction

- Accounting application asks OS to print a document for it.
- OS passes it along to the [device driver](#) for the printer.
- Device driver doesn't care about what it is printing, applications don't care about how it is printed.
- OS hides the complexities of hardware from user applications.



Resource Management

Accounting application that helps you calculate income tax, needs to work with large amounts of data that exceed the capacity of our RAM.

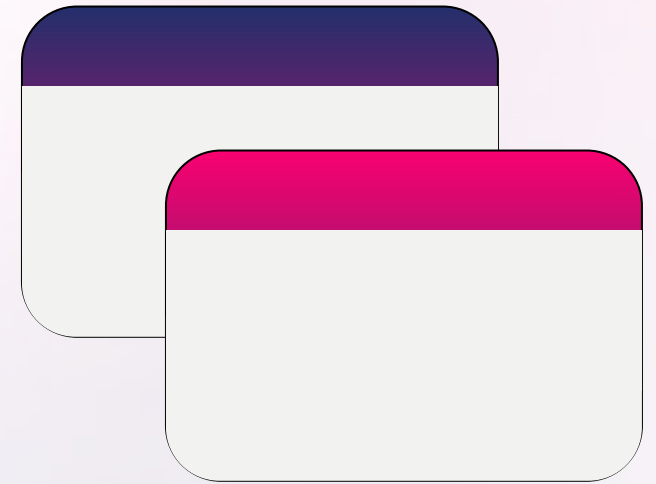
OS overcomes this limitation by storing RAM content in the HDD, and loading them back into the RAM when needed. All of this still “feels” like RAM from the program’s point of view. ([Paging](#))

Multitasking

Each CPU core can run a single task at any given time.

Contemporary operating systems implement scheduling schemes that ensure all processes (a running program is called a process) get their share of CPU time, and achieve an “illusion” of concurrency.

Example: Listening to music as you work, while your memory resident anti-virus software protects your computer from malware attacks.



Multitasking - Multithreading

Each process may have more than one **thread of execution** and run “simultaneous” tasks within themselves. This is referred to as **multithreading**.

Multithreaded programs may also utilize multiple CPU cores.

Example: User interfaces that respond to clicks and other interactions while an intensive task is being performed.



Multitasking - Multi-core CPU

Running independent jobs on multiple CPU cores enhances the performance significantly.

There are limits on how fast and small a CPU core can get, due to production costs and physical phenomena, such as quantum tunneling. ([Vertical scaling](#))

Adding CPU cores along with design solutions for division of labour, partially alleviates the situation. ([Horizontal scaling](#))

Virtual Memory

Resource management becomes even more important with multiple processes running simultaneously.

OS allocates resources, such as RAM, to processes, prevents them from interfering with one another.

Processes are under the impression that they are the only one running on the system, and are unaware of how actual physical memory is being distributed among them. ([Virtual memory](#))

Security

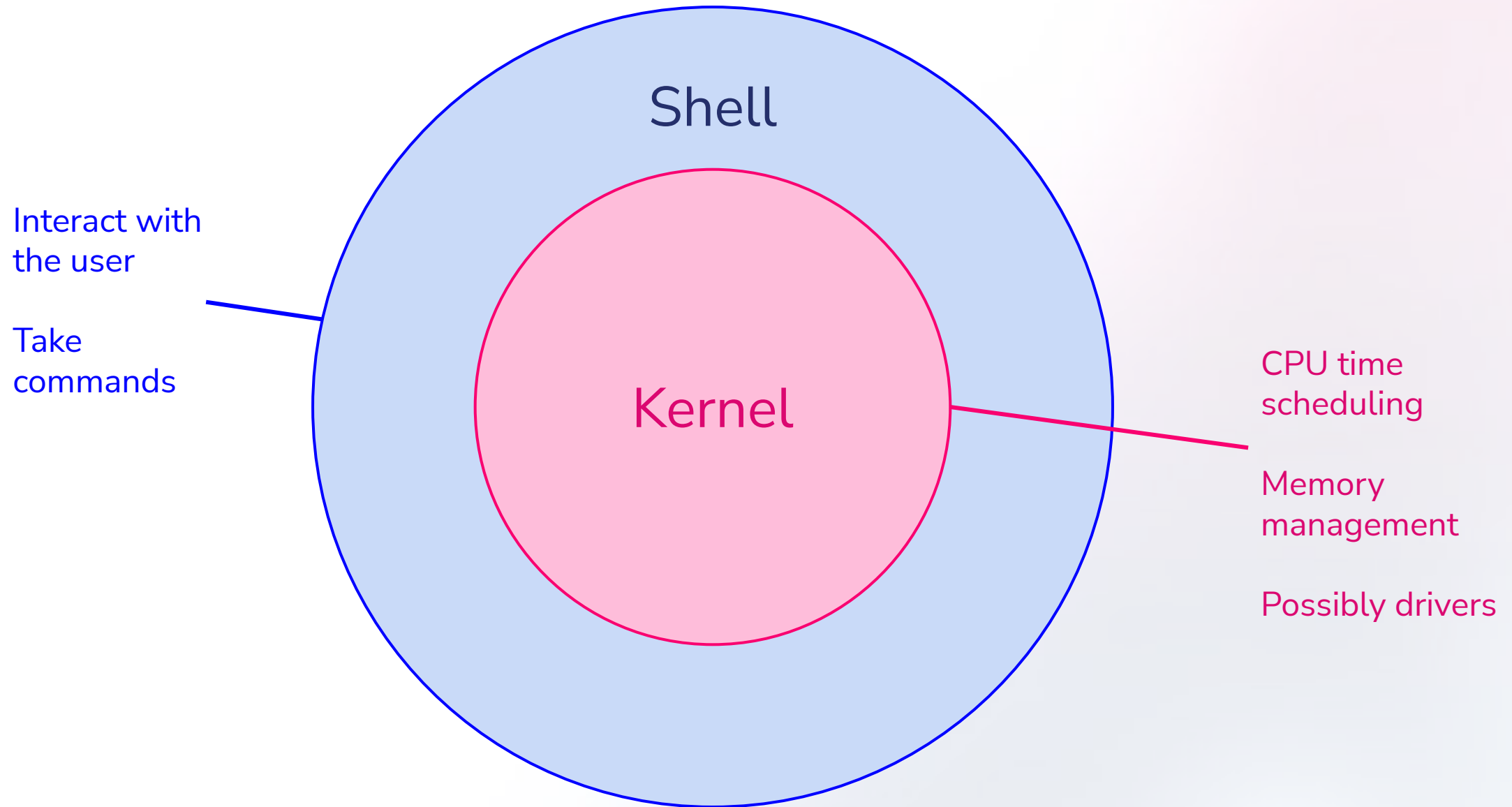
Especially important in multi-user environments.

OS enforces policies regarding their rights and privileges. Certain users may not run administrator utilities, change sensitive system settings, see or modify files that belong to others, etc.

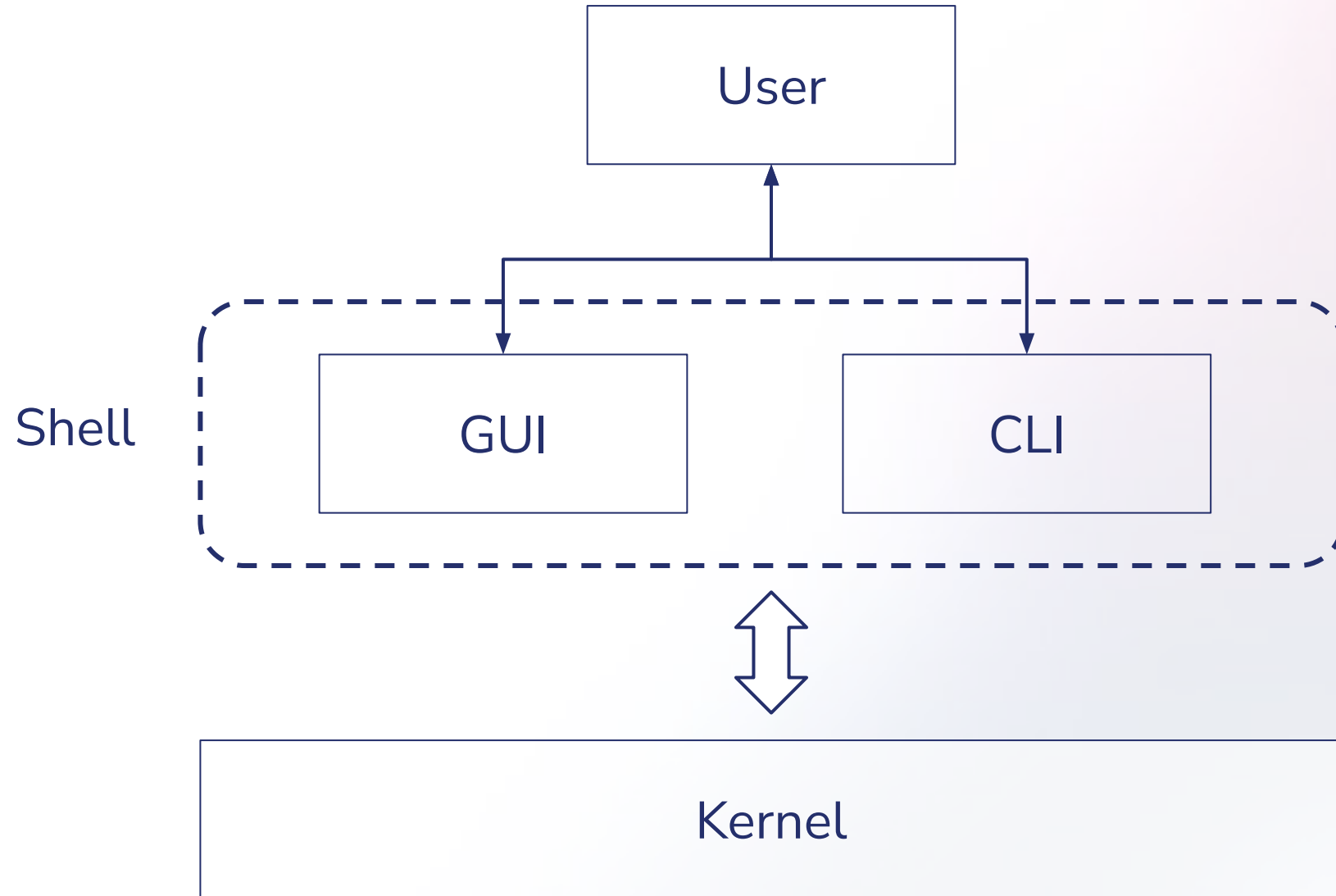
User Interface

- Command Line Interface (CLI) where users type commands. (MS-DOS)
- Graphical User Interface (GUI, also pronounced like gooey) where users are guided by visual cues interact with intuitive shapes using keyboards and later on, mice. More recent ones come with layers of “windows” stacked on top of one another. (Norton Commander, Windows)

Kernel and Shell



Kernel and Shell



Interrupts

Interrupts are signals regarding events that require attention. They **interrupt** the job that CPU was working on so, they can be **handled** (by kernel or firmware).

- Hardware interrupts are caused by hardware events, such as a key being pressed by the user.
- Software interrupts are caused by programs making system calls or certain errors.

In essence, it's a value that is set in a register.

File System

Is the way files are organized and stored on a disk. Affects the maximum size of files and partitions, file name lengths, etc.

Examples:

- FAT (File allocation table) MS-DOS default
- FAT32 Used by Win95
- NTFS (NT file system) Used by WindowsNT and later on by other versions
- ext4 Used by most Linux distros

https://en.wikipedia.org/wiki/File_system

Booting

1. Mainboard firmware starts running when the computer is powered on.
2. Power-on self test (POST) is performed. Success results in a motherboard beep.
3. Master Boot Record (MBR, UEFI on newer systems) is loaded, and starts to run, starts a secondary loader such as LILO(Linux Loader) or GRUB.
4. That bootloader then loads the kernel.
5. Kernel takes control and launches the shell.

<https://uefi.org/>

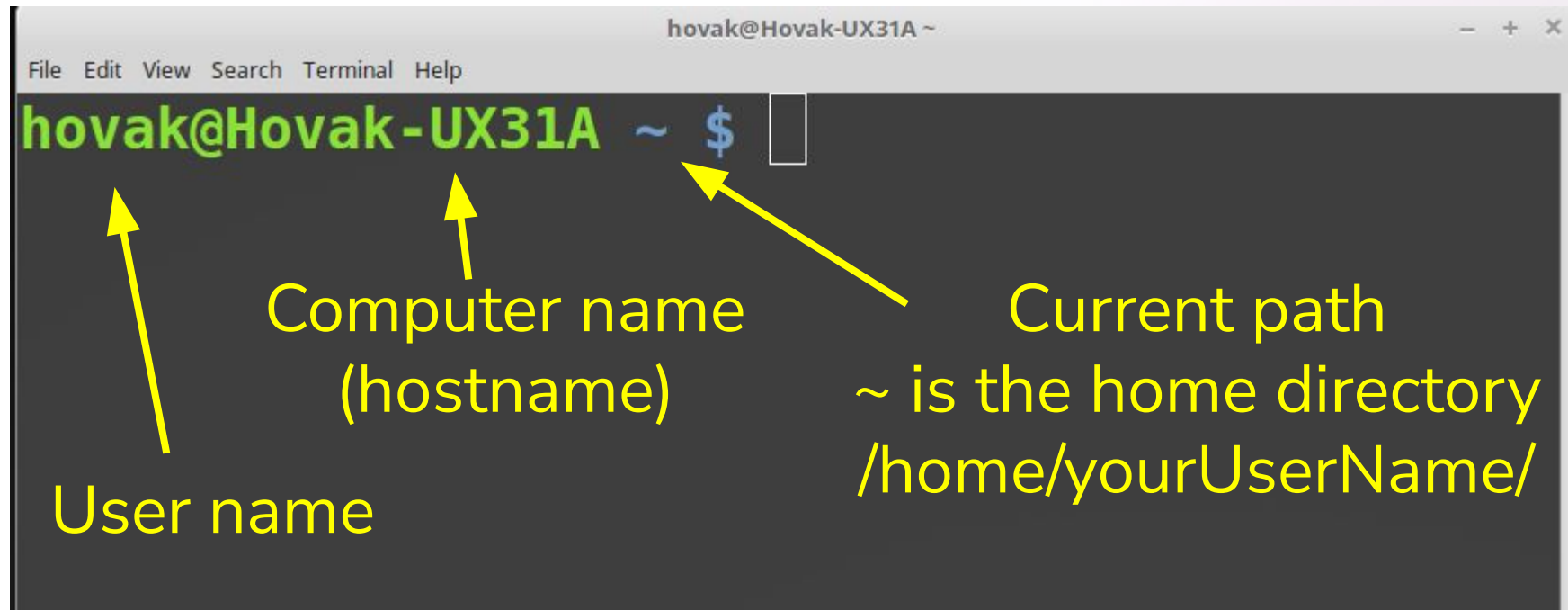
Terminology

- **ROM** (Read-only memory) is a chip where firmware is stored. Recent motherboards use flash memory and their firmware can be updated.
- **BIOS** (Basic input output system)
- **CMOS** (Complementary metal–oxide–semiconductor, pronounced "see-moss")

For storing date/time, boot priorities, various settings.
Powered by a battery.

Bash

Terminals (and terminal emulators) provide the user with a command line interface (CLI) to interact with the shell, namely Bash.



Bash

The general format for commands is as follows:

command --options input1 input2...

By convention, a command with a **--help** option displays a short guide that describes its usage and input parameters.

Similarly, **--version** options always display the version info of a command.

man somecommand on the other hand, displays a detailed manual for a command.

Bash

List of commands used during the class

apt-get

cd

cp

echo

find

gnome-system-monitor

grep

ls

man

mkdir

printenv

ps

pwd

rm

sudo

Bash

Streams

- > Write to file
- >> Append to file
- | Pipe output to another command

Wildcards in names

- ? Single character
- * Any number of characters
- [*list*] List of characters

Path

- / Root Directory
- .. Parent Directory
- . Current Directory

Bash

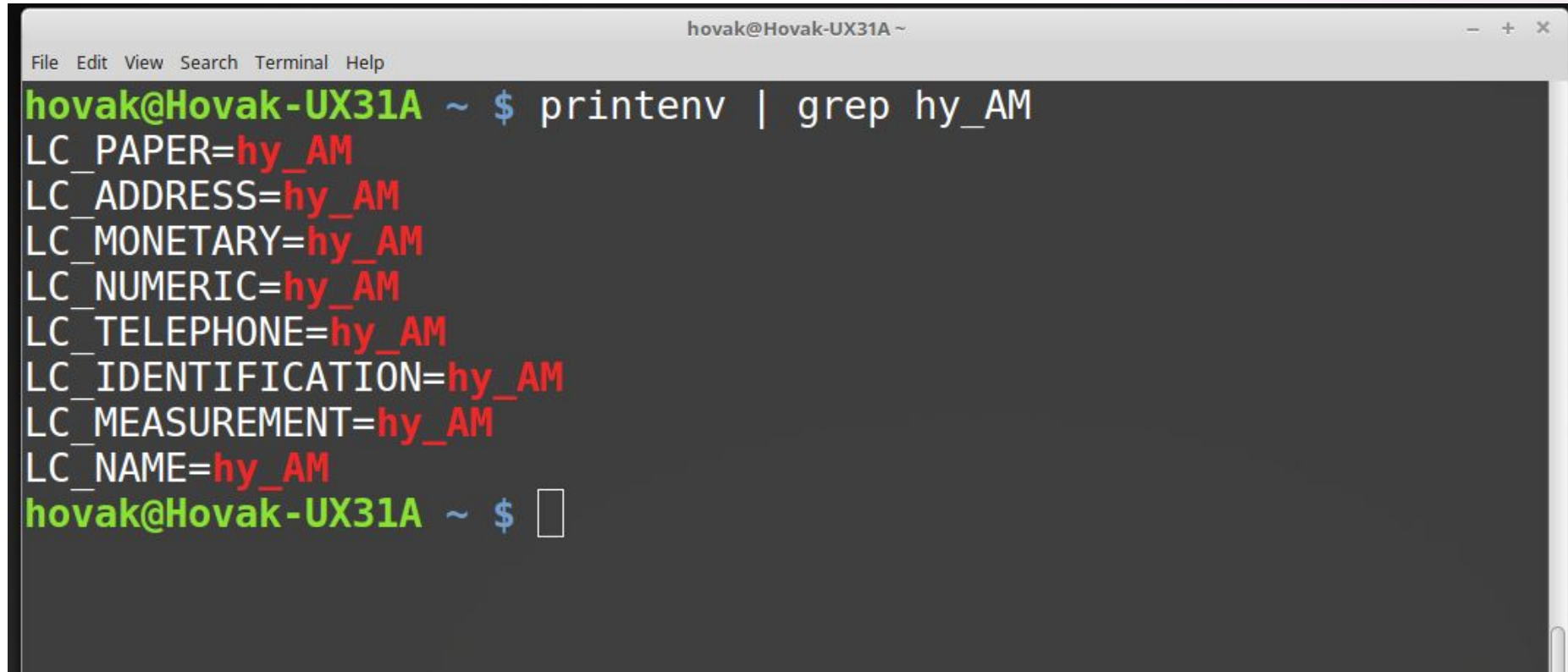
```
hovak@Hovak-UX31A ~  
File Edit View Search Terminal Help  
hovak@Hovak-UX31A ~ $ pwd  
/home/hovak  
hovak@Hovak-UX31A ~ $ cd Documents/  
hovak@Hovak-UX31A ~/Documents $ ls  
Copy of CS110w01-Number representations.pdf  
CS110w02 - Number Representations (cont).pdf  
CS110w03 - Hardware (cont) (1).pdf  
CS110w04 - Operating Systems.pdf  
Linux Installation Guide.pdf  
OS (1).pdf  
hovak@Hovak-UX31A ~/Documents $ pwd  
/home/hovak/Documents  
hovak@Hovak-UX31A ~/Documents $ cd ..  
hovak@Hovak-UX31A ~ $ cd ..  
hovak@Hovak-UX31A /home $ cd hovak/  
hovak@Hovak-UX31A ~ $
```

Bash

The | symbol redirects the output of the first command into the next.

grep searches its input for a given value.

printenv prints all environment variables.

A terminal window titled 'hovak@Hovak-UX31A ~' with a menu bar (File, Edit, View, Search, Terminal, Help). The prompt is 'hovak@Hovak-UX31A ~ \$'. The command 'printenv | grep hy_AM' has been entered. The output shows several environment variables with 'hy_AM' as their value: LC_PAPER=hy_AM, LC_ADDRESS=hy_AM, LC_MONETARY=hy_AM, LC_NUMERIC=hy_AM, LC_TELEPHONE=hy_AM, LC_IDENTIFICATION=hy_AM, LC_MEASUREMENT=hy_AM, and LC_NAME=hy_AM. The prompt is now 'hovak@Hovak-UX31A ~ \$' with a cursor.

```
hovak@Hovak-UX31A ~ $ printenv | grep hy_AM
LC_PAPER=hy_AM
LC_ADDRESS=hy_AM
LC_MONETARY=hy_AM
LC_NUMERIC=hy_AM
LC_TELEPHONE=hy_AM
LC_IDENTIFICATION=hy_AM
LC_MEASUREMENT=hy_AM
LC_NAME=hy_AM
hovak@Hovak-UX31A ~ $
```

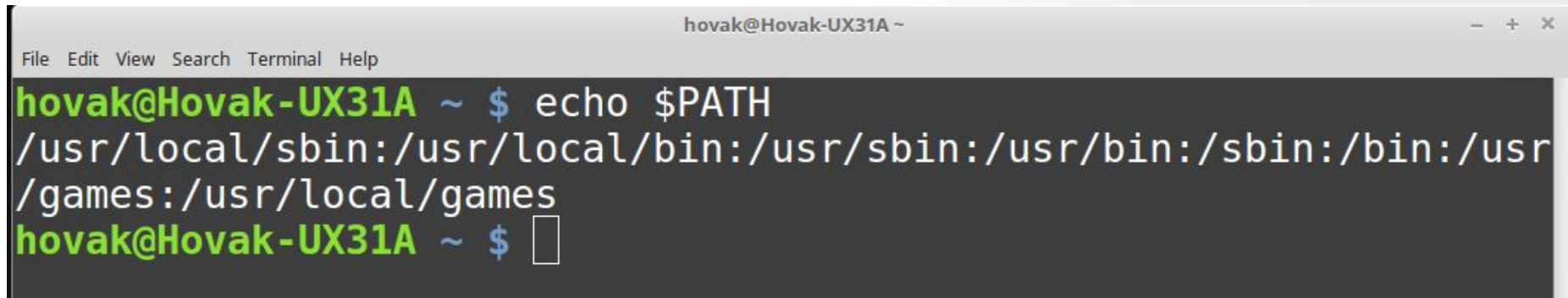

Bash

```
hovak@Hovak-UX31A ~  
File Edit View Search Terminal Help  
hovak@Hovak-UX31A ~ $ find . -name "*.pdf"  
./Documents/Copy of CS110w01-Number representations.pdf  
./Documents/CS110w02 - Number Representations (cont).pdf  
./Documents/CS110w03 - Hardware (cont) (1).pdf  
./Documents/Linux Installation Guide.pdf  
./Documents/CS110w04 - Operating Systems.pdf  
./Documents/OS (1).pdf  
./Music/S0.pdf  
./Music/OS.pdf  
./Downloads/CS110w01 - History.pdf  
./Downloads/CS110w03 - Hardware components (1).pdf  
./Downloads/Computer Hardware.pdf  
./Downloads/CS110w03 - Hardware (cont).pdf  
./Downloads/CS110w02 - Number Representations.pdf  
./Downloads/OS.pdf  
./Downloads/CS110-HW01 (1).pdf  
./Downloads/CS110-HW01.pdf  
./Downloads/CS110w03 - Hardware components.pdf  
./Downloads/CS110w01-Number representations.pdf  
hovak@Hovak-UX31A ~ $ find . -name "*.pdf" >> ListOfSlides.txt  
hovak@Hovak-UX31A ~ $
```

Bash

Printing the contents of the PATH environment variable. If a command is issued and is not present in the indicated directory, these locations are checked next.

Any command present in these directories can be invoked from anywhere.

A terminal window titled 'hovak@Hovak-UX31A ~' with a menu bar (File, Edit, View, Search, Terminal, Help). The prompt is 'hovak@Hovak-UX31A ~ \$' and the command 'echo \$PATH' has been entered. The output is displayed on three lines: '/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games'. The prompt is now 'hovak@Hovak-UX31A ~ \$' followed by a cursor.

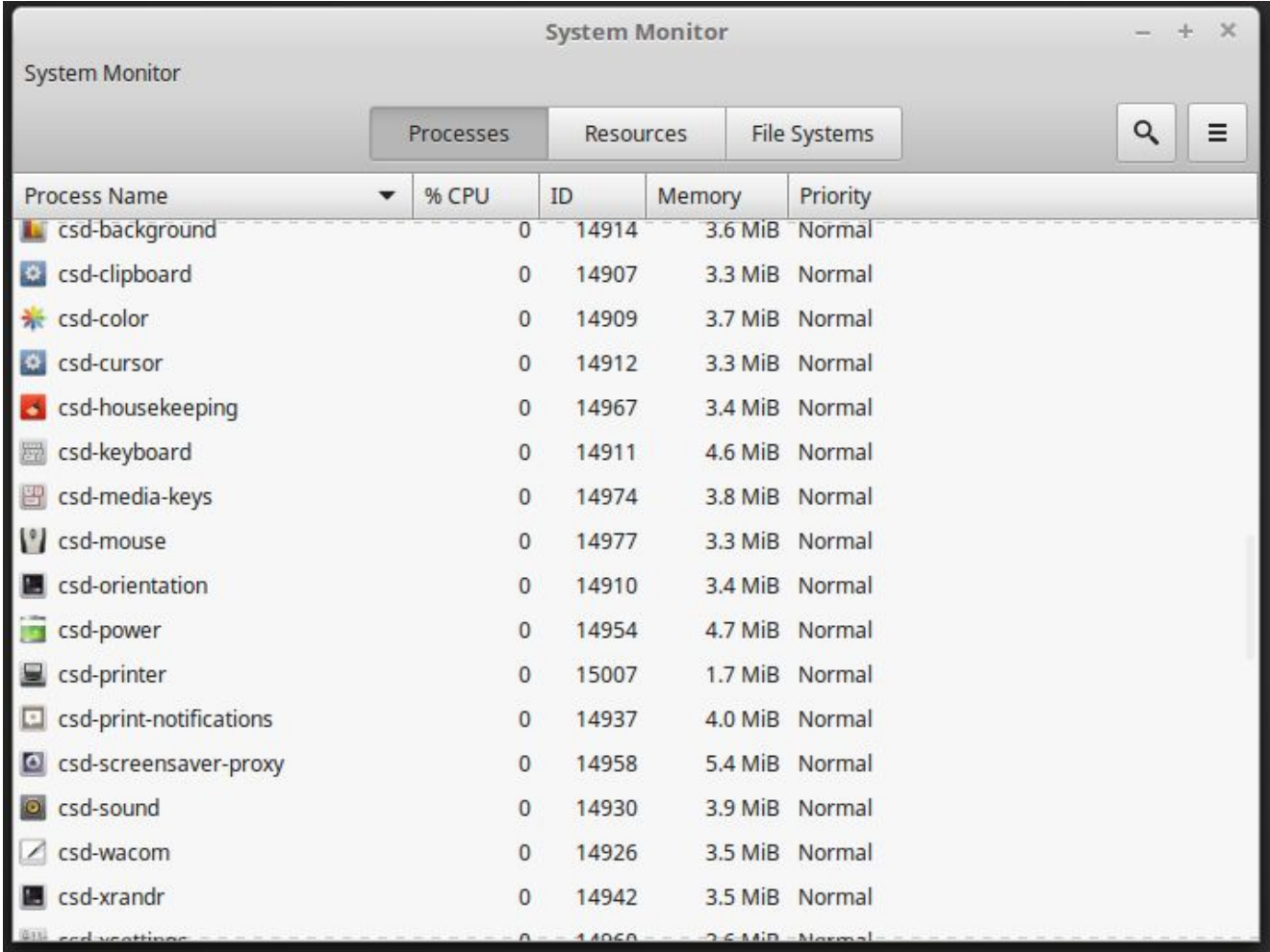
```
hovak@Hovak-UX31A ~ $ echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr
/games:/usr/local/games
hovak@Hovak-UX31A ~ $
```

Bash Scripts

Bash commands can be written in bulk in a text file, and run at once. Particularly useful for automating repetitive jobs, possibly for a lot of computers.

```
#!/bin/bash
# long list of software I install on a fresh
system
apt-get install python
apt-get install virtualbox
apt-get install geary
echo "All done!"
```

gnome-system-monitor



The screenshot shows the 'System Monitor' application window. The 'Processes' tab is selected, displaying a list of running processes. The table columns are Process Name, % CPU, ID, Memory, and Priority. The processes listed are all 'csd-*' (GNOME Shell components) and are all using 0% CPU. The memory usage varies, with 'csd-keyboard' using the most at 4.6 MiB. The priority for all processes is 'Normal'.

Process Name	% CPU	ID	Memory	Priority
csd-background	0	14914	3.6 MiB	Normal
csd-clipboard	0	14907	3.3 MiB	Normal
csd-color	0	14909	3.7 MiB	Normal
csd-cursor	0	14912	3.3 MiB	Normal
csd-housekeeping	0	14967	3.4 MiB	Normal
csd-keyboard	0	14911	4.6 MiB	Normal
csd-media-keys	0	14974	3.8 MiB	Normal
csd-mouse	0	14977	3.3 MiB	Normal
csd-orientation	0	14910	3.4 MiB	Normal
csd-power	0	14954	4.7 MiB	Normal
csd-printer	0	15007	1.7 MiB	Normal
csd-print-notifications	0	14937	4.0 MiB	Normal
csd-screensaver-proxy	0	14958	5.4 MiB	Normal
csd-sound	0	14930	3.9 MiB	Normal
csd-wacom	0	14926	3.5 MiB	Normal
csd-xrandr	0	14942	3.5 MiB	Normal
csd-xsettings	0	14960	2.6 MiB	Normal

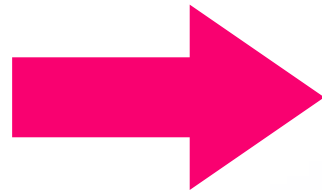
Graphical task manager, showing running processes.

Compiler

“**Compiler**, Computer software that translates (compiles) source code written in a high-level language (e.g., C++) into a set of machine-language instructions that can be understood by a digital computer’s CPU.”

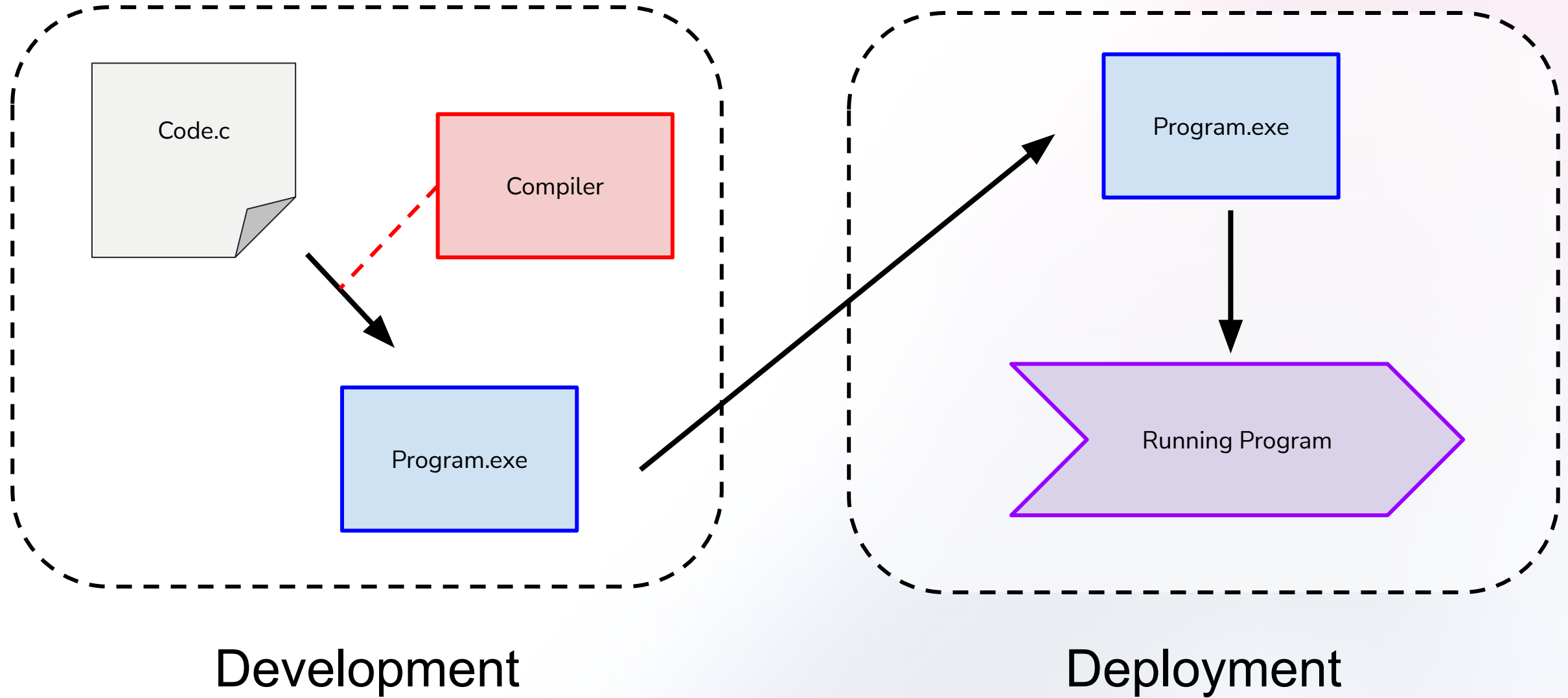
(Encyclopaedia Britannica, 2020)

```
Code code (code){  
    More code(7 + 6);  
    // comments  
    human readable code()  
}
```

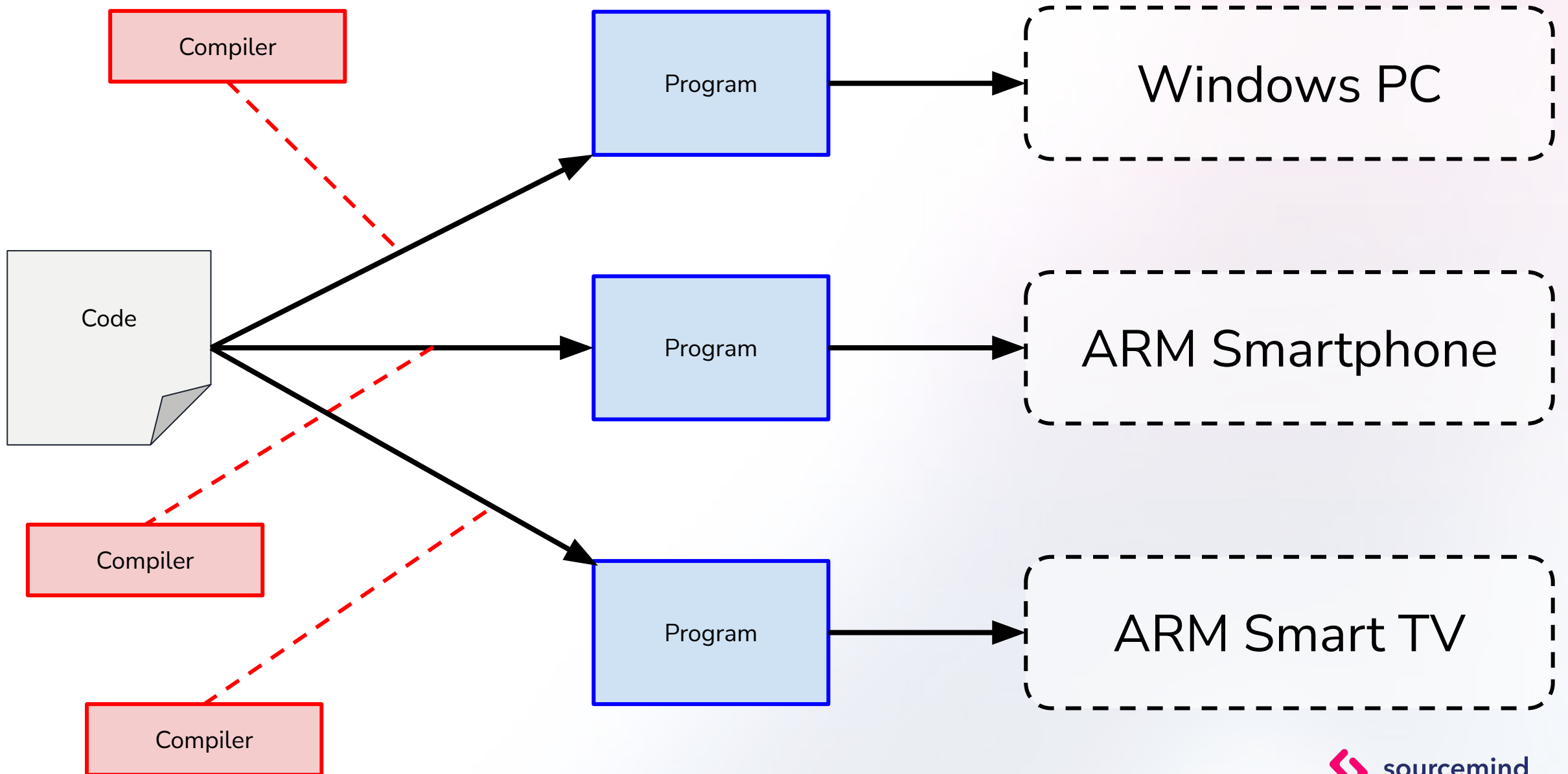


```
0000 0010 0000 0111  
0000 0001 ....
```

Compiled Approach



Compiled Approach



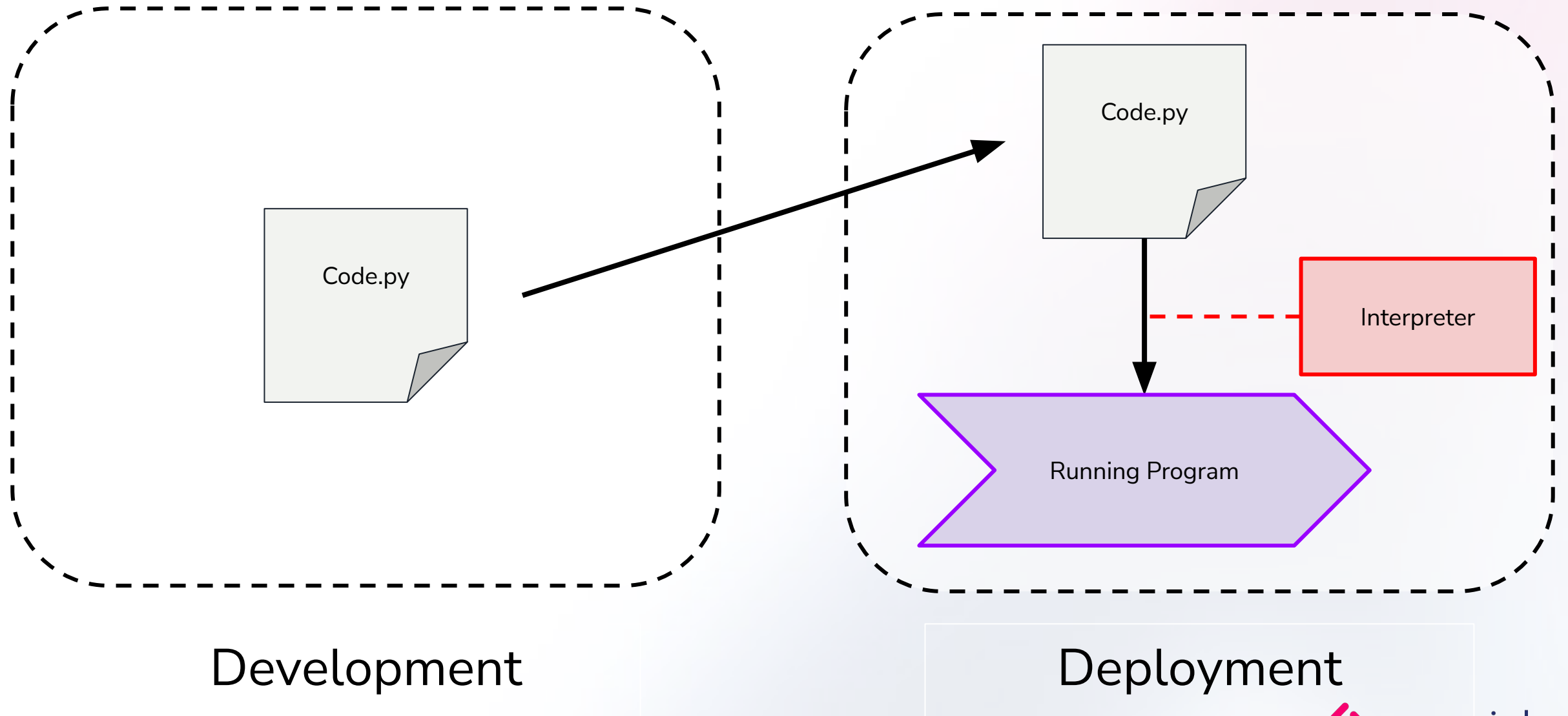
Interpreter

Interpreters translate given source code to machine code at runtime and therefore must be present on the target machine.

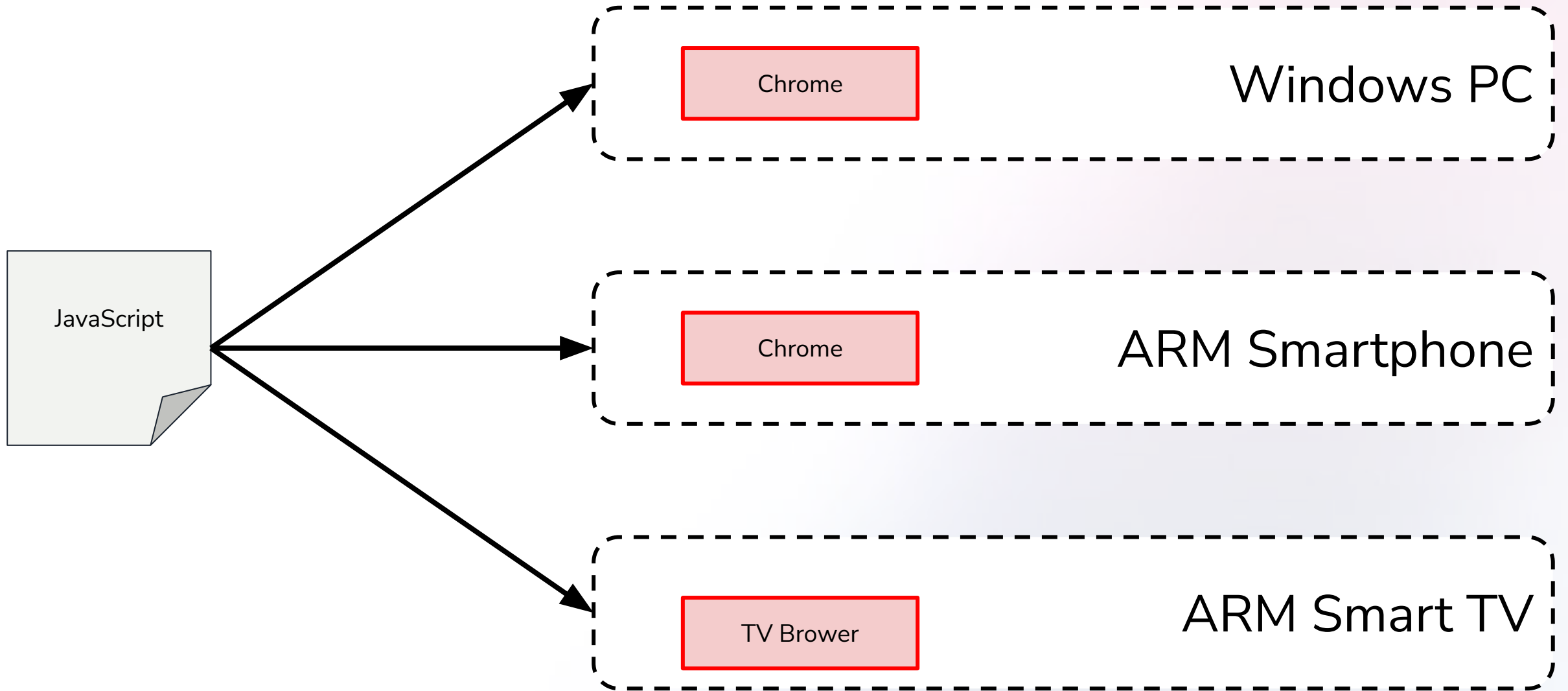
It is often said that languages like Python or Javascript are interpreted languages, but technically there is nothing inherent about the languages that make them interpreted or compiled.

e.g. nothing prevents anyone to create a Python compiler or a C++ interpreter.

Interpreted Approach



Interpreted Approach



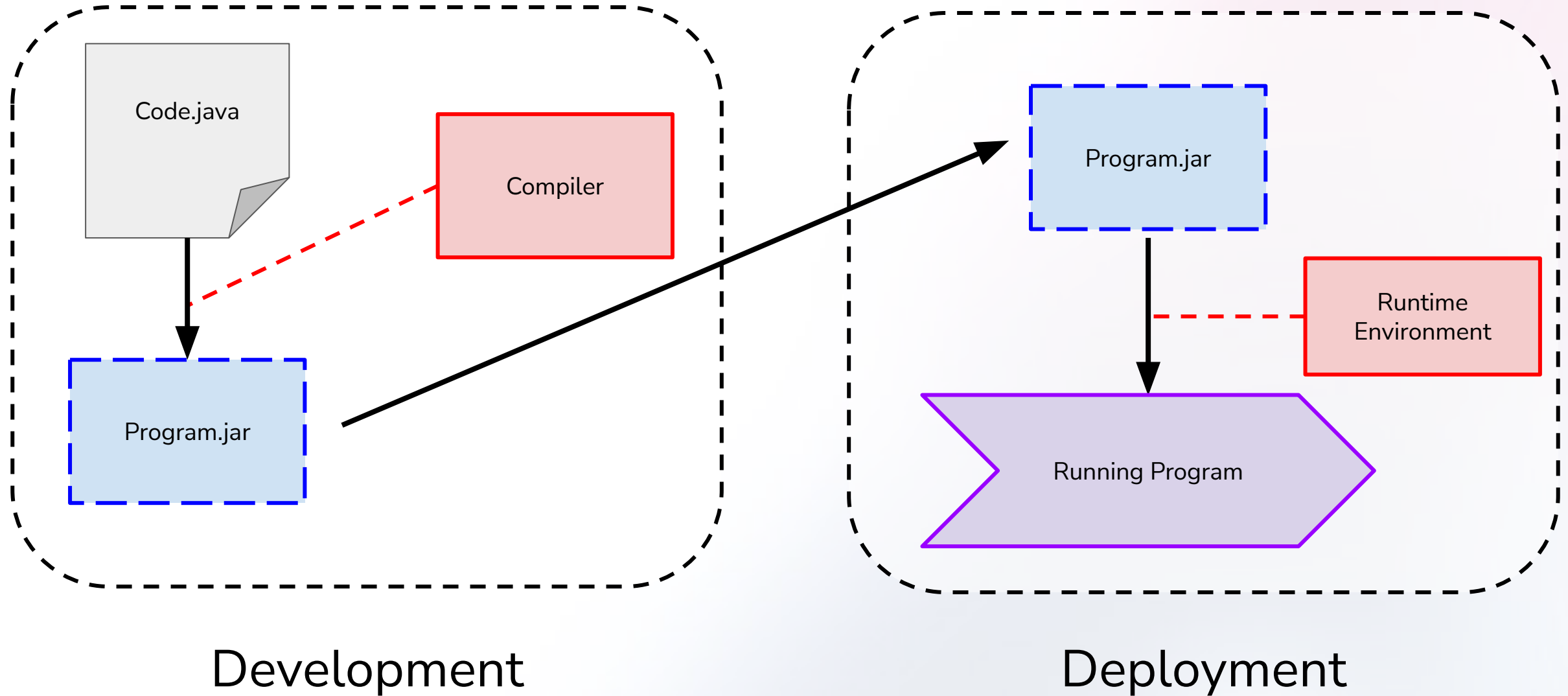
Compiled Code

- Difficult to reverse engineer
- Needs to be compiled for every platform
- Faster / Larger files

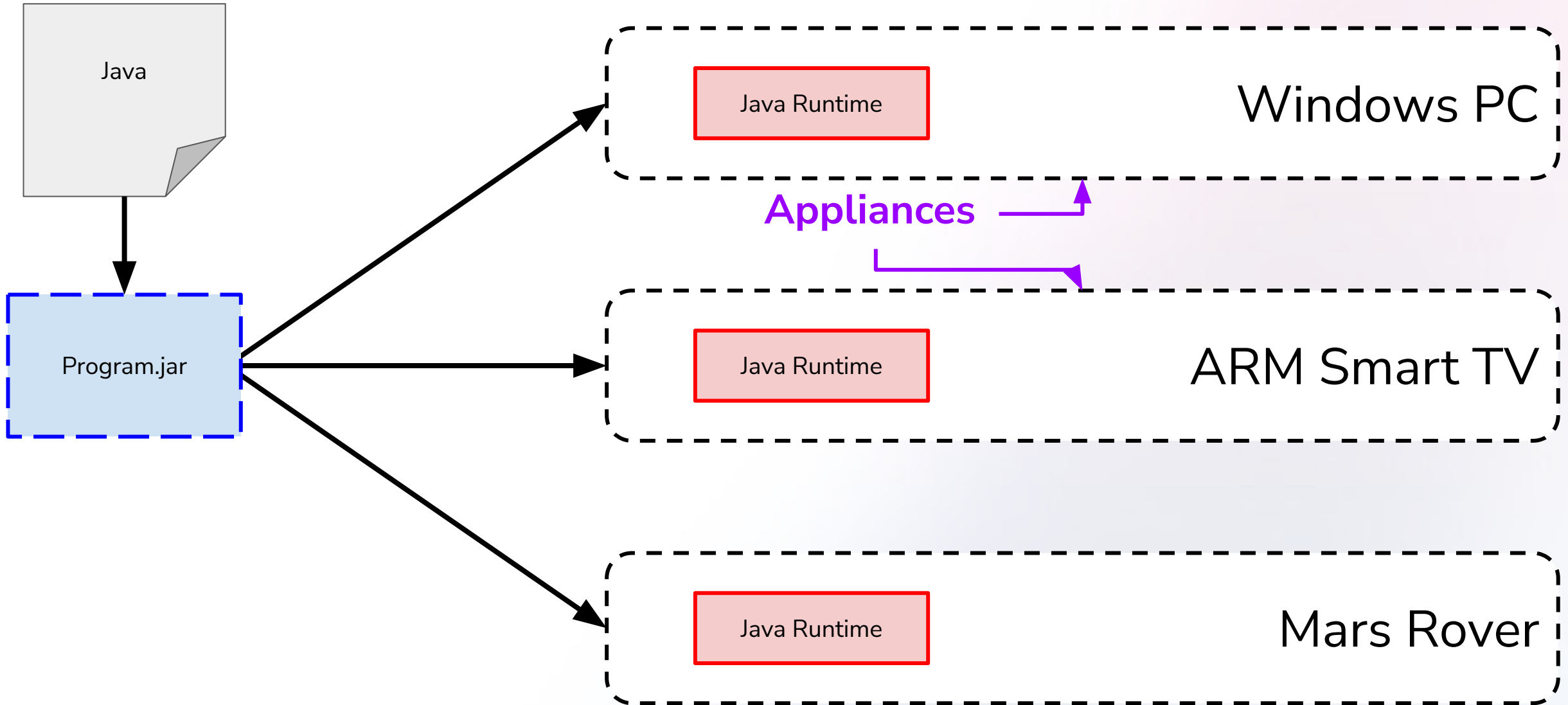
Interpreted Code

- Easy to modify
- Needs an interpreter during runtime
- Smaller file sizes / Slower

Mixed Approach



Java Runtime Environment



Virtual Machines

A virtual machine is a computer that is emulated by software. Their uses include:

- Running and testing operating systems
- Server virtualization and load distribution
- Malware analysis

It should be noted that Java Virtual Machine is a specification and actual JRE implementations might not contain an actual virtual machine. (JIT compilation)

Networks

“Computer networking refers to connected computing devices (such as laptops, desktops, servers, smartphones, and tablets) and an ever-expanding array of IoT devices (such as cameras, door locks, doorbells, refrigerators, audio/visual systems, thermostats, and various sensors) that communicate with one another.” (Cisco, 2021)

These communications are facilitated by various protocols.

TCP/IP is the protocol suite on which the Internet relies.

TCP-IP

Applications

TCP

IP

Data Link

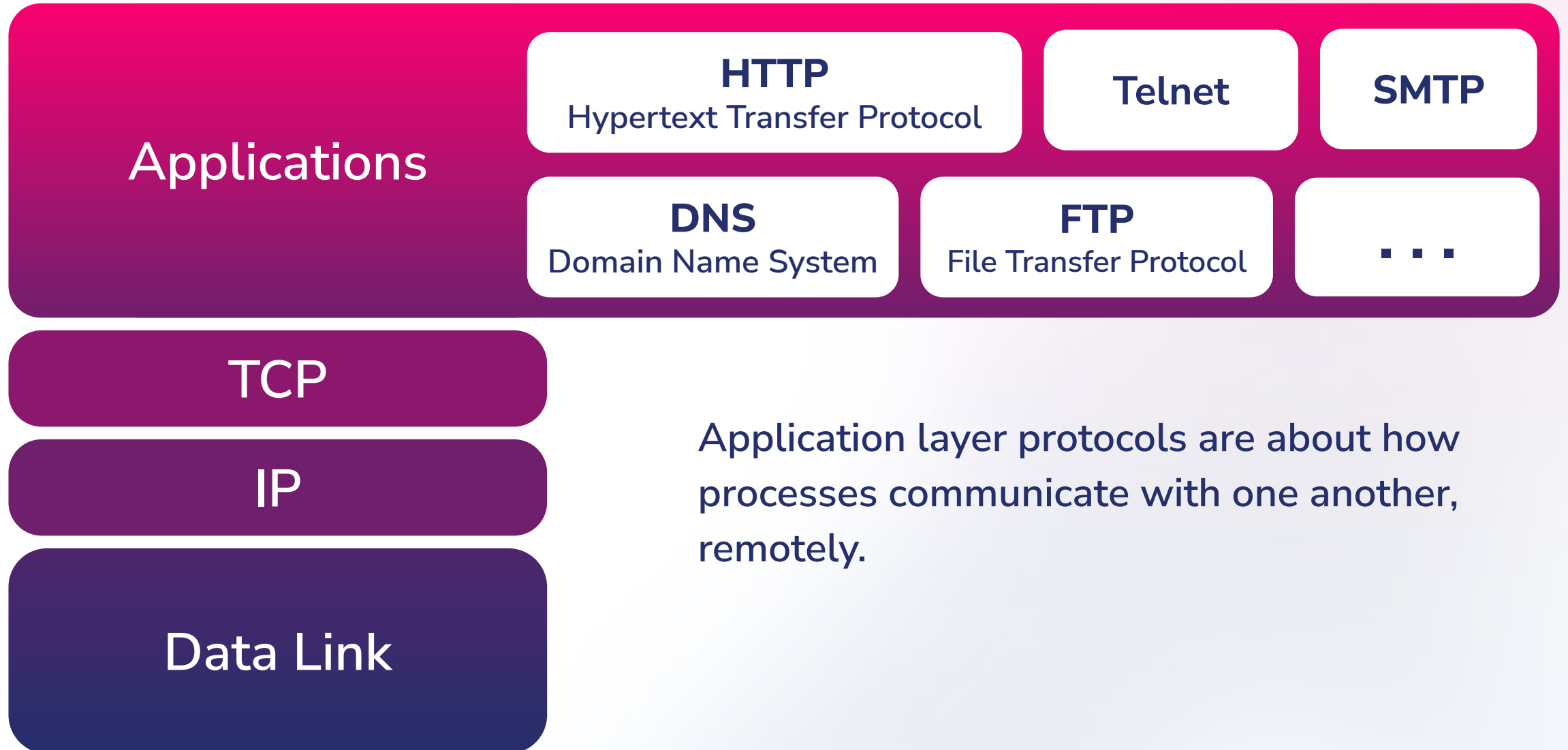
An email application does not concern itself with:

- Whether it is connected to the internet via WiFi, cellular data or satellite
- What routes the communication passes through

Ethernet
IEEE 802.3

Wifi
IEEE 802.11

TCP-IP



TCP-IP

Applications

TCP (Transmission Control Protocol)

Rearranges sent packets in the correct order at the destination.

Sends confirmation to the sender.

TCP

TCP

UDP

...

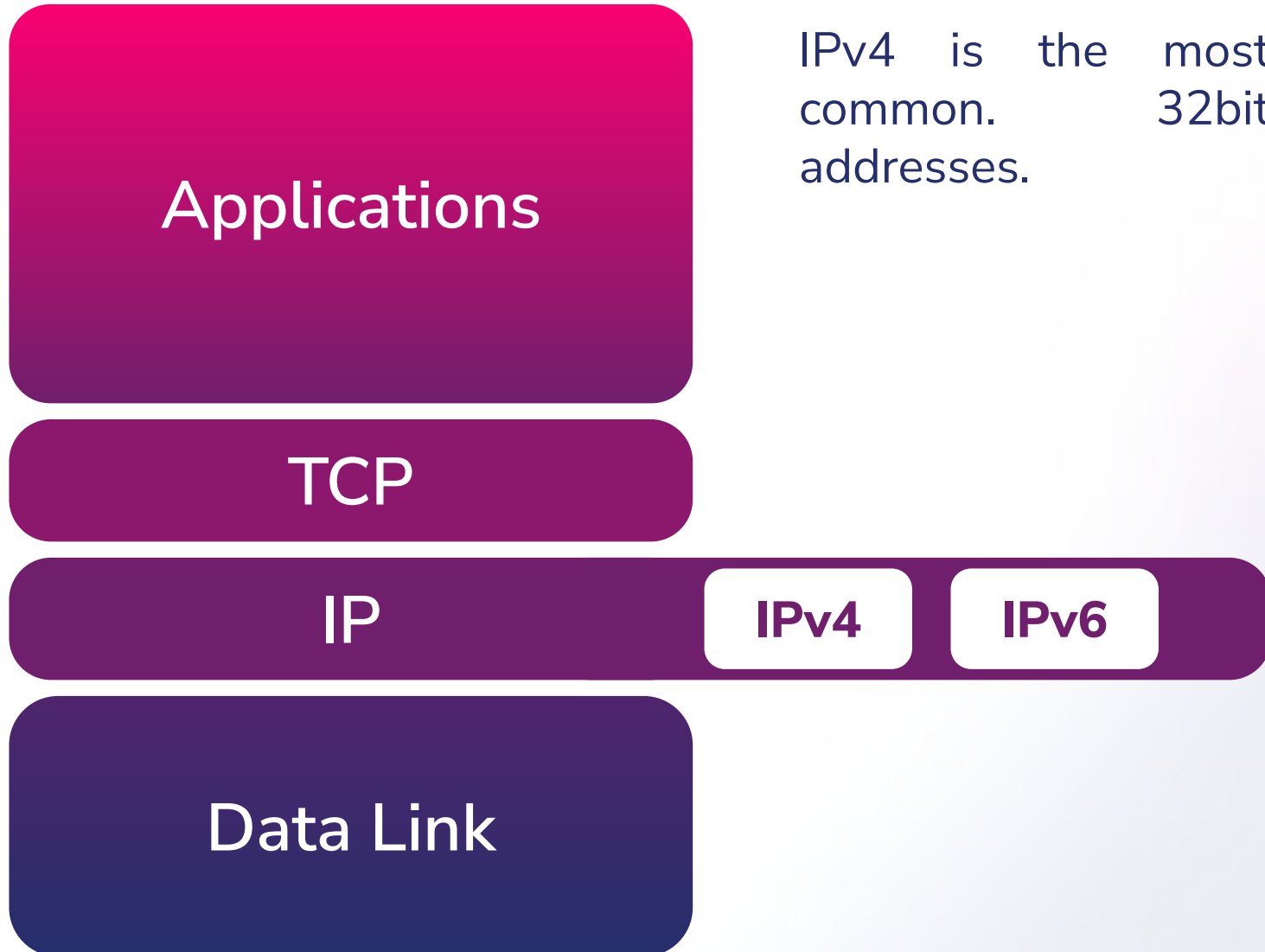
IP

UDP (User Datagram Protocol)

Is less “pedantic” than TCP. Used for loss-tolerant connections where speed is more important than reliability.

Data Link

TCP-IP



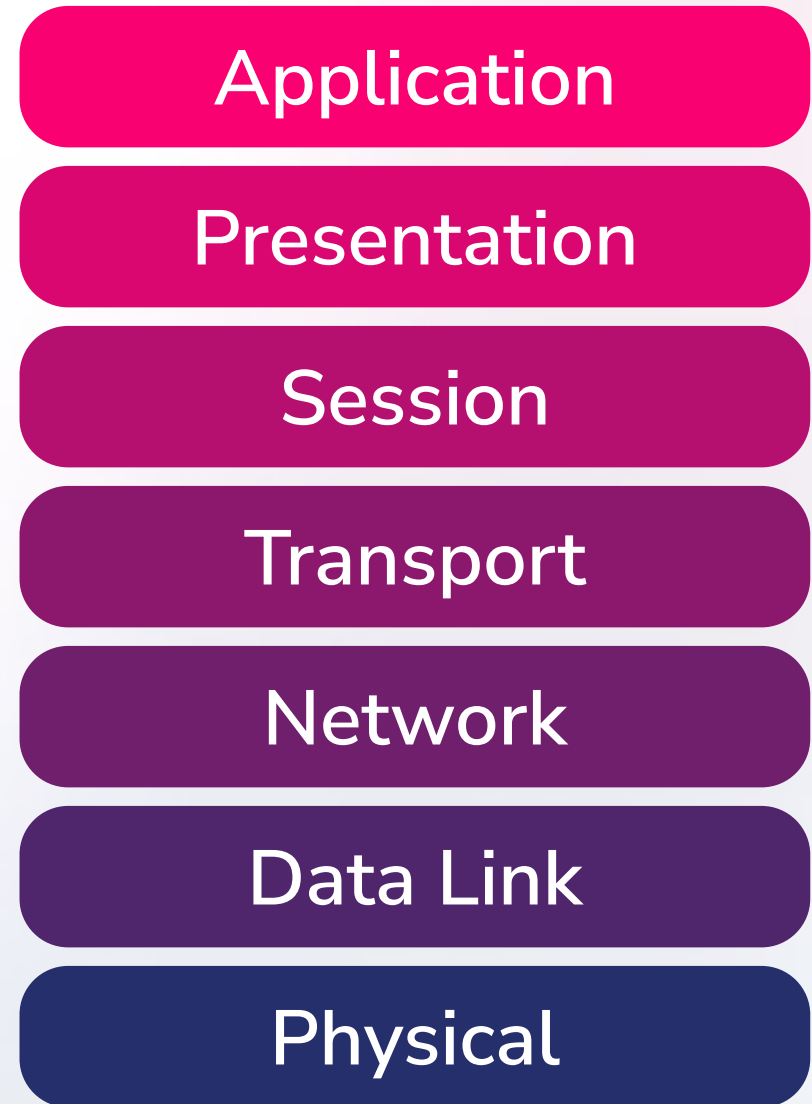
IPv4 is the most common. 32bit addresses.

IPv6 is 128 bits.

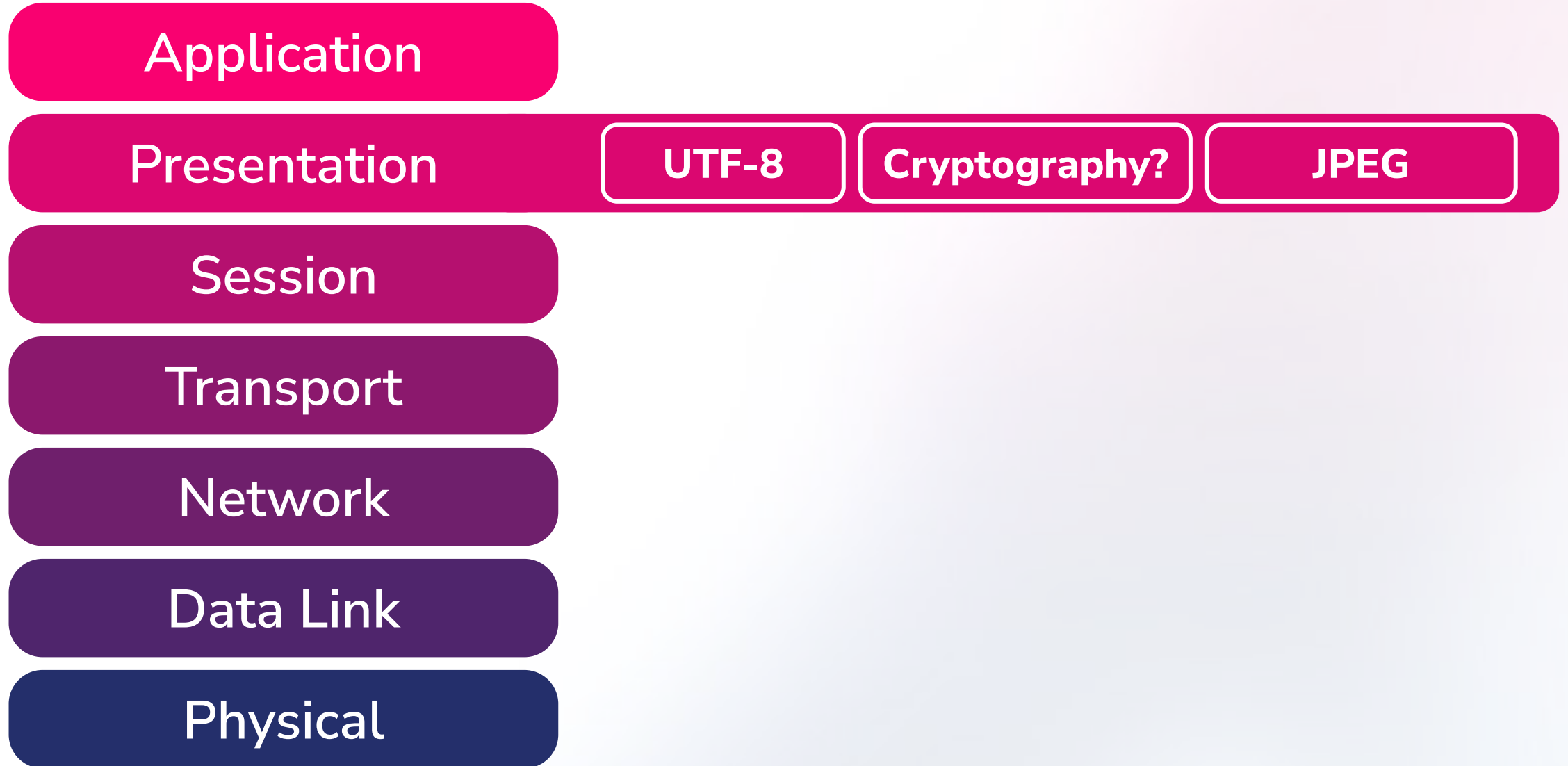
Open Systems Interconnection Model (OSI)

The OSI model is a conceptual model that classifies protocols.

Lower levels are closer to the hardware.



Open Systems Interconnection Model (OSI)



Open Systems Interconnection Model (OSI)

Application

Presentation

Session

PPTP

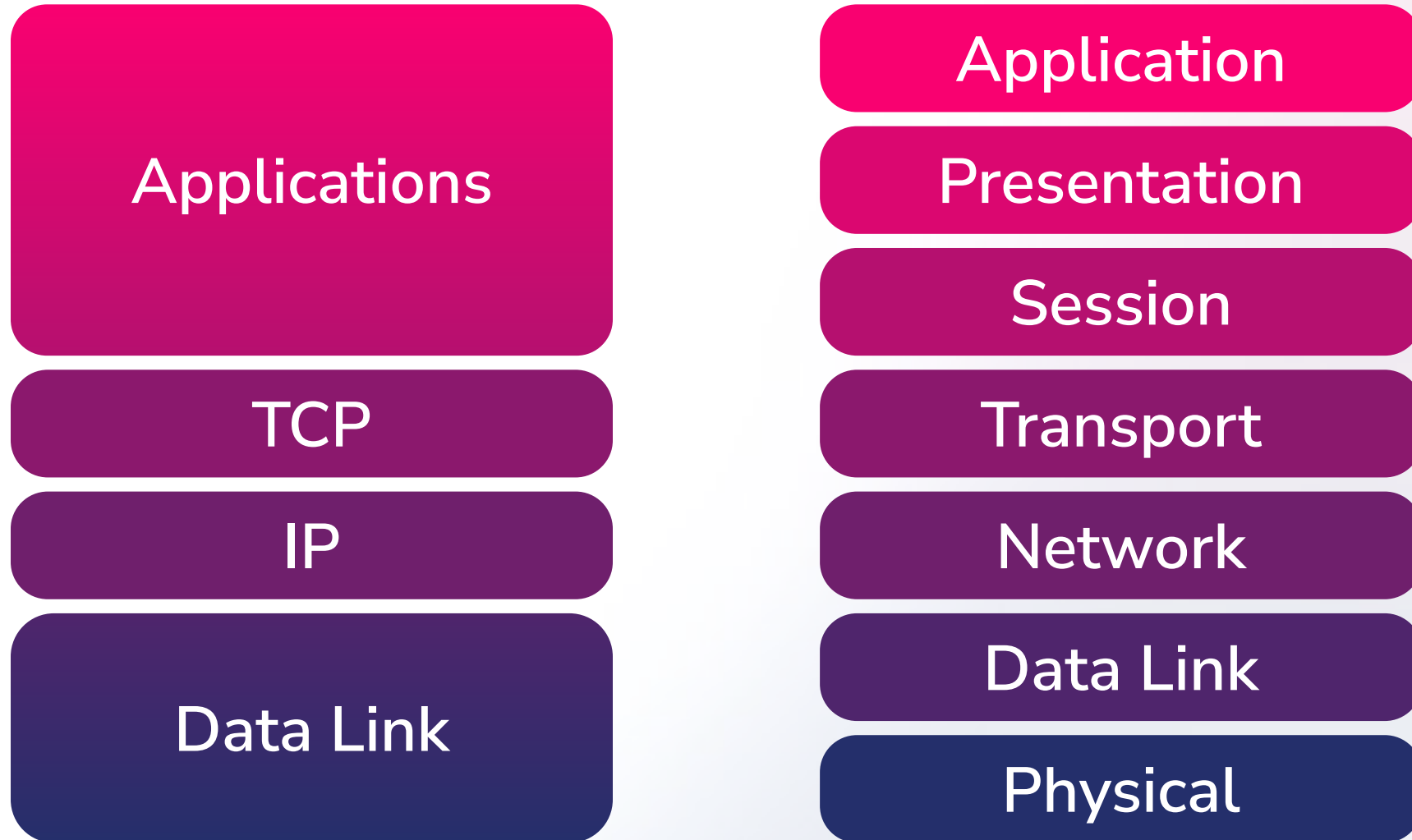
Transport

Network

Data Link

Physical

OSI/TCP-IP Equivalency



Domain Name Service (DNS)

“**DNS**, in full **domain name system**, network service that converts between **World Wide Web** “name” addresses and numeric **Internet** addresses.”

(Encyclopedia Britannica, 2021)

“The basic form of this structure is the name of a machine, followed by a top level domain (TLD), separated by dots (periods).”

(Encyclopedia Britannica, 2021)

Hypertext Transfer Protocol (http)

“HTTP, in full **HyperText Transfer Protocol**, standard application-level protocol used for exchanging files on the World Wide Web. HTTP runs on top of the TCP/IP protocol. Web browsers are HTTP clients that send file requests to Web servers, which in turn handle the requests via an HTTP service.”

(Encyclopedia Britannica, 2021)

<https://developer.mozilla.org/en-US/docs/Web/HTTP/Overview>

Uniform Resource Locator (URL)

“**URL**, in full **Uniform Resource Locator**, address of a resource on the Internet, or of a file stored locally. The resource can be any type of file stored on a server, such as a Web page, a text file, a graphics file, or an application program. The address contains three elements: the type of protocol used to access the file (e.g., HTTP for a Web page, ftp for an FTP site); the domain name or IP address of the server where the file resides; and, optionally, the pathname to the file (i.e., description of the file’s location). “

(Encyclopedia Britannica, 2021)

Relative and Absolute Paths

- **Absolute paths** are in relation to the host system or the network.
`C:\Windows\system32.dll`

- **Relative paths** are given with relation to the active directory.

A program can access a photo located in the same folder as itself.
`./background.jpg`

- Dot (.) indicates “here”, two dots (..) indicate parent director.
- Windows paths denote directories with backslash \ whereas in UNIX based systems and the Internet, forward slash / is used.

Internet Topology

- There are programs that map the topology of networks. One such program is `traceroute` and displays every node between two hosts.
- Physical map of underwater cables
<https://www.submarinecablemap.com/>

Useful Links

Linux directory structure

<https://www.tecmint.com/linux-directory-structure-and-important-files-paths-explained/>

Bash terminal

<https://www.youtube.com/watch?v=oxuRxtrO2Ag>

List of important commands

<https://www.howtogeek.com/412055/37-important-linux-commands-you-should-know/>

List of commands your shell knows

<https://unix.stackexchange.com/questions/94775/list-all-commands-that-a-shell-knows>

Comprehensive bash reference

<https://devdocs.io/bash/>

References

1. Britannica, The Editors of Encyclopaedia. "Compiler". Encyclopedia Britannica, 4 May. 2021, <https://www.britannica.com/technology/compiler>. Accessed 15 June 2021.
2. Cisco. "Netwroking". Cisco Systems, <https://www.cisco.com/c/en/us/solutions/enterprise-networks/what-is-computer-networking.html>. Accessed 15 June 2021.
3. Britannica, The Editors of Encyclopaedia. "DNS". Encyclopedia Britannica, 22 Oct. 2020, <https://www.britannica.com/topic/DNS>. Accessed 15 June 2021.
4. Britannica, The Editors of Encyclopaedia. "HTTP". Encyclopedia Britannica, 10 Oct. 2017, <https://www.britannica.com/technology/HTTP>. Accessed 15 June 2021.
5. Britannica, The Editors of Encyclopaedia. "URL". Encyclopedia Britannica, 10 Aug. 2020, <https://www.britannica.com/technology/URL>. Accessed 15 June 2021.