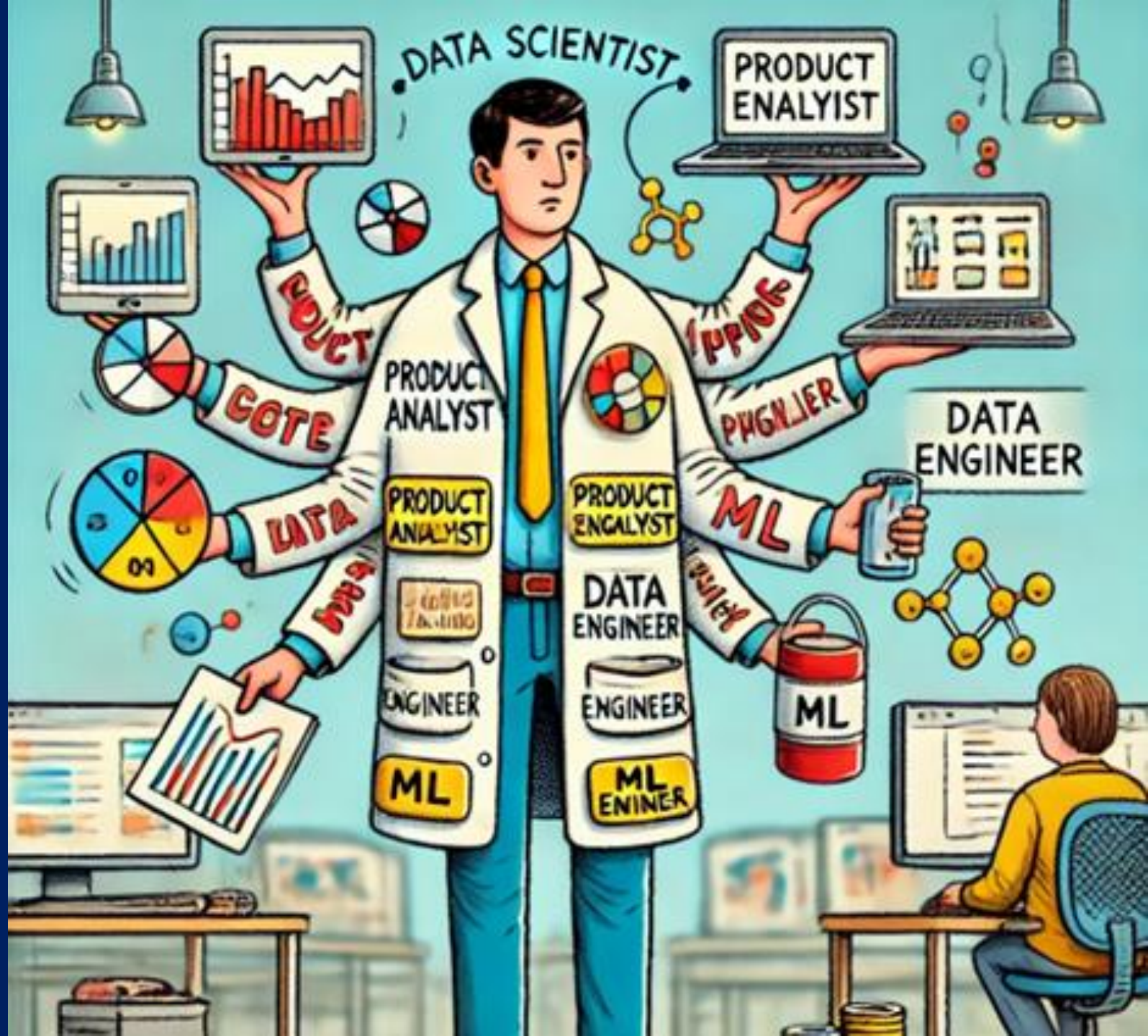
CS 51 Computer Science Principles

Module 3: Data, Internet, Computer and Programming

Unit 4: Hardware

LECTURE 9 COMPUTERS

DR. ERIC CHOU
IEEE SENIOR MEMBER



Objectives

- Computing Devices
- Computer Organization
- Instruction Set Architecture
- Parallelism
- Computing Models: Sequential Computing versus Parallel Computing
- Hardware and Software

Objectives

- Seven Layer Architecture (OSI Model)
- Layer 1: Physical Layer
- Layer 2: Data Link Layer
- Layer 3: Network Layer

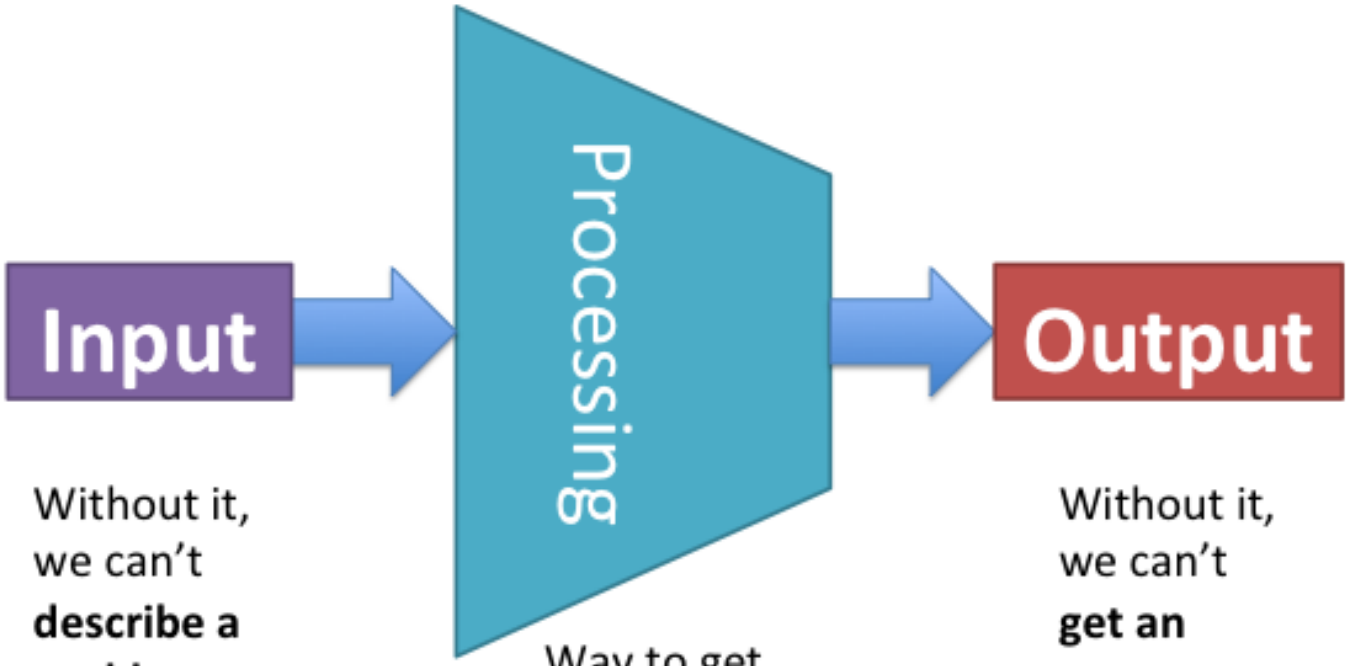
Computing Devices

LECTURE 1

Computing Devices

- A computing device is a physical artifact that can run a program. Some examples include computers, tables, servers, routers, and smart sensors. The device must be able to take inputs, process the inputs, and then calculate results based on those inputs.

What All Computers Need



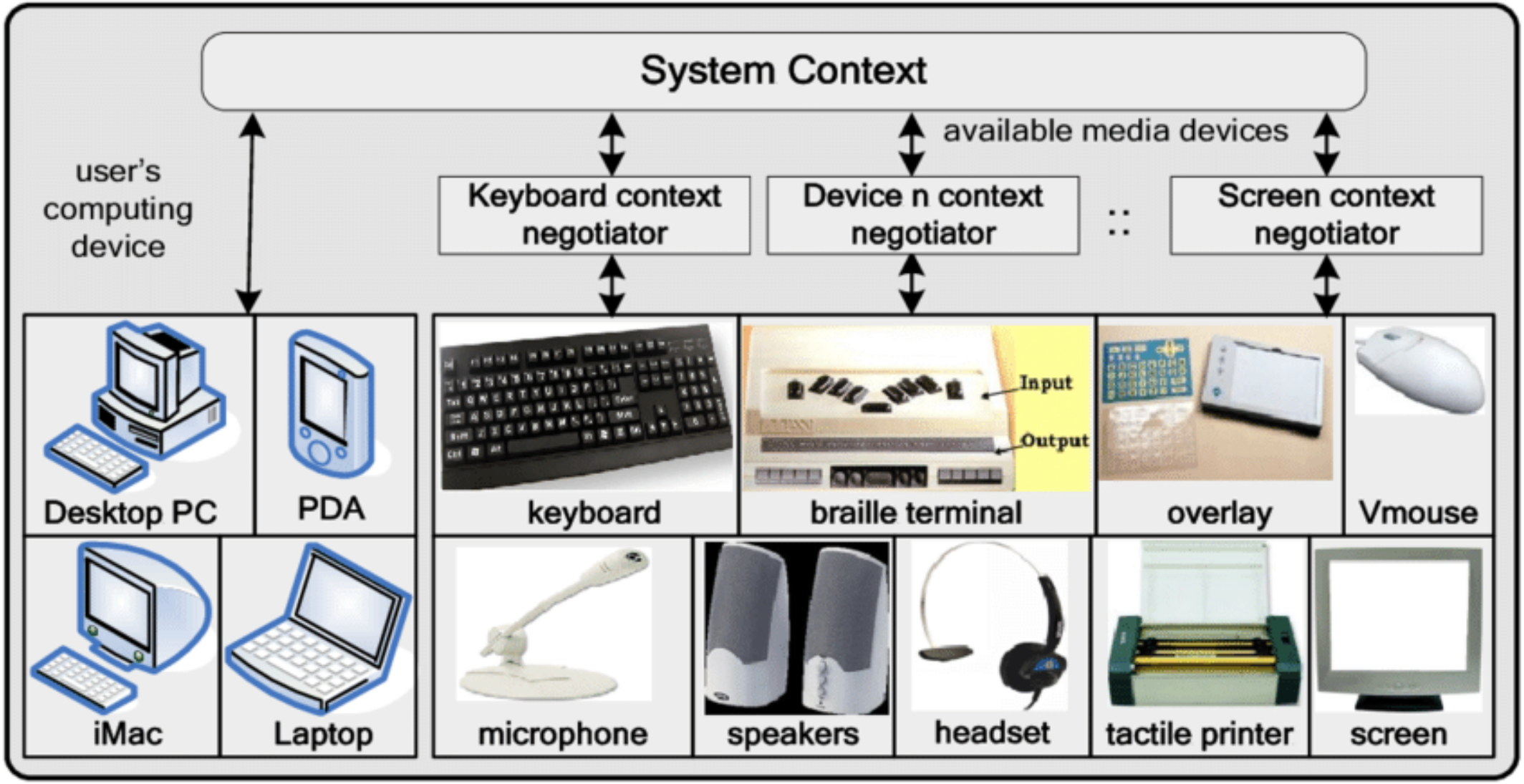
Without it, we can't **describe a problem**

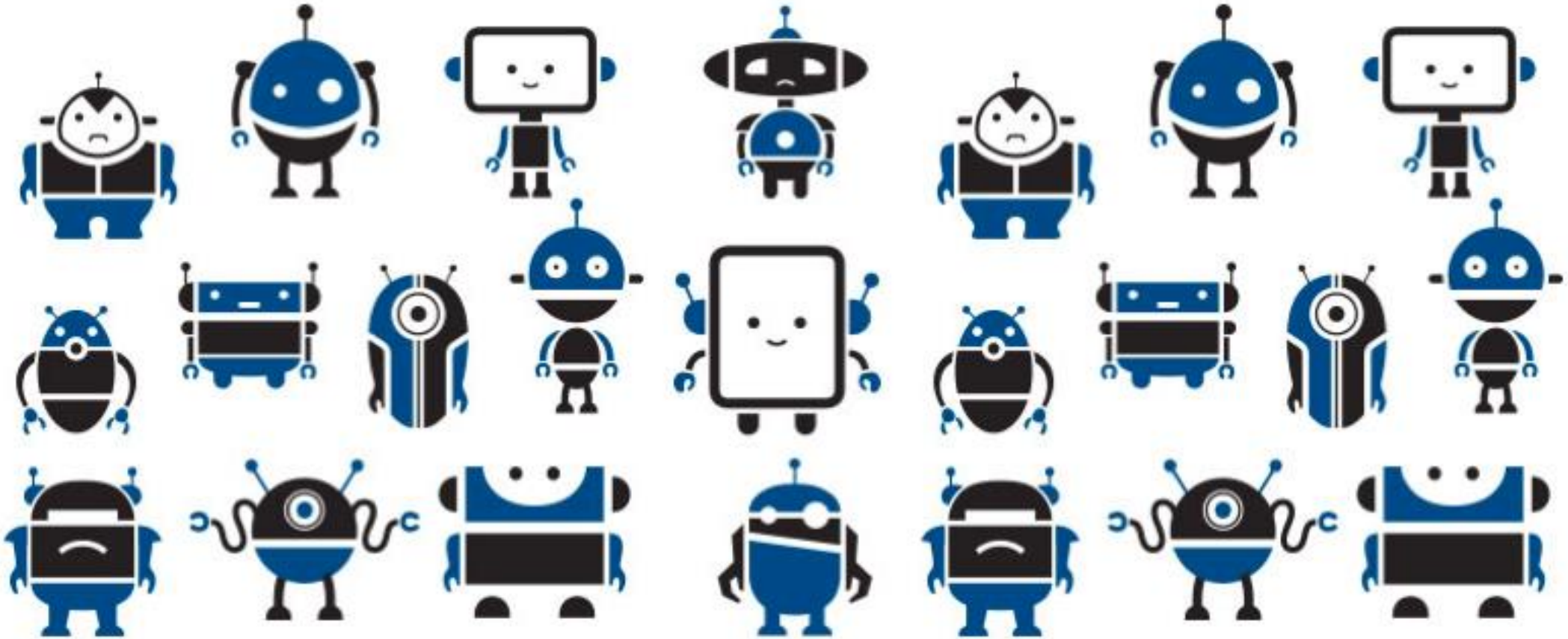
Way to get from input to output

Without it, we can't **get an answer**

Computing Devices

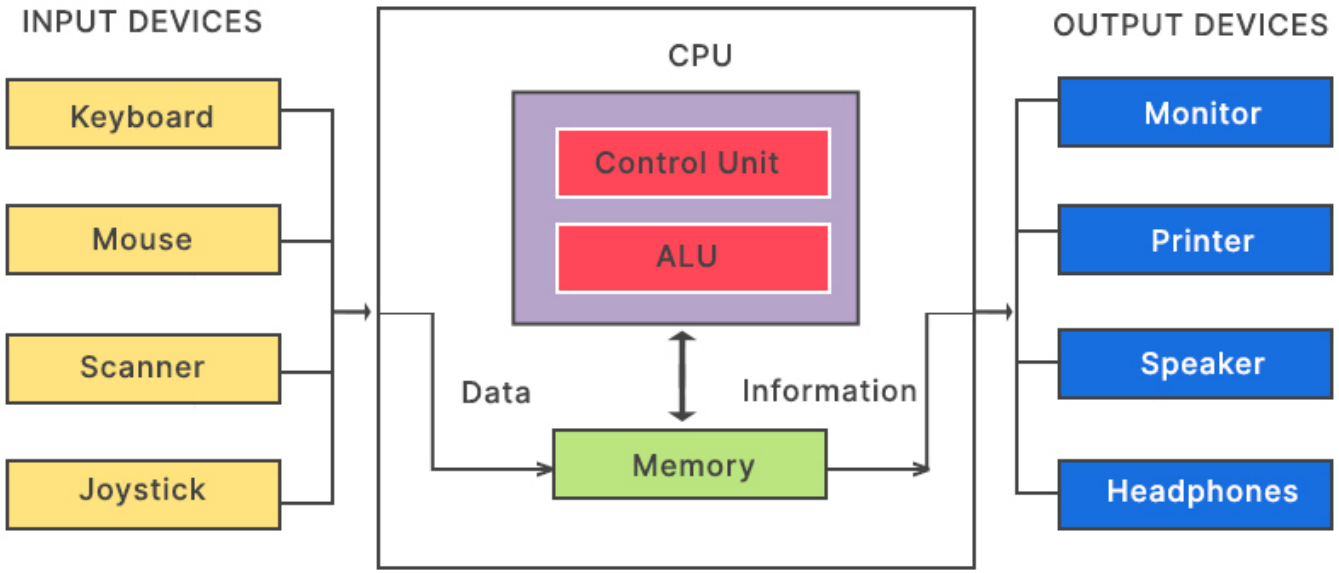
- Some examples of a computing input device include but not limited to a keyboard, mouse, scanner, microphone, and webcam. Some examples of a computing output device include a monitor, printer, and speakers.
- A computer is a computing device, but not all computing devices are computers. A computing device could be a calculator, printer, digital camera, smart phone, computer, or many more.
- A computing system is a group of computing devices and programs working together for a common purpose. A type of computing system Is a computer network. A computer network is a group of interconnected computing devices capable of sending or receiving data.


System Context
available media devices
user's computing device
Keyboard context negotiator
Device n context negotiator
..
Screen context negotiator
Desktop PC
PDA
keyboard
braille terminal
Input
Output
overlay
Vmouse
iMac
Laptop
microphone
speakers
headset
tactile printer
screen



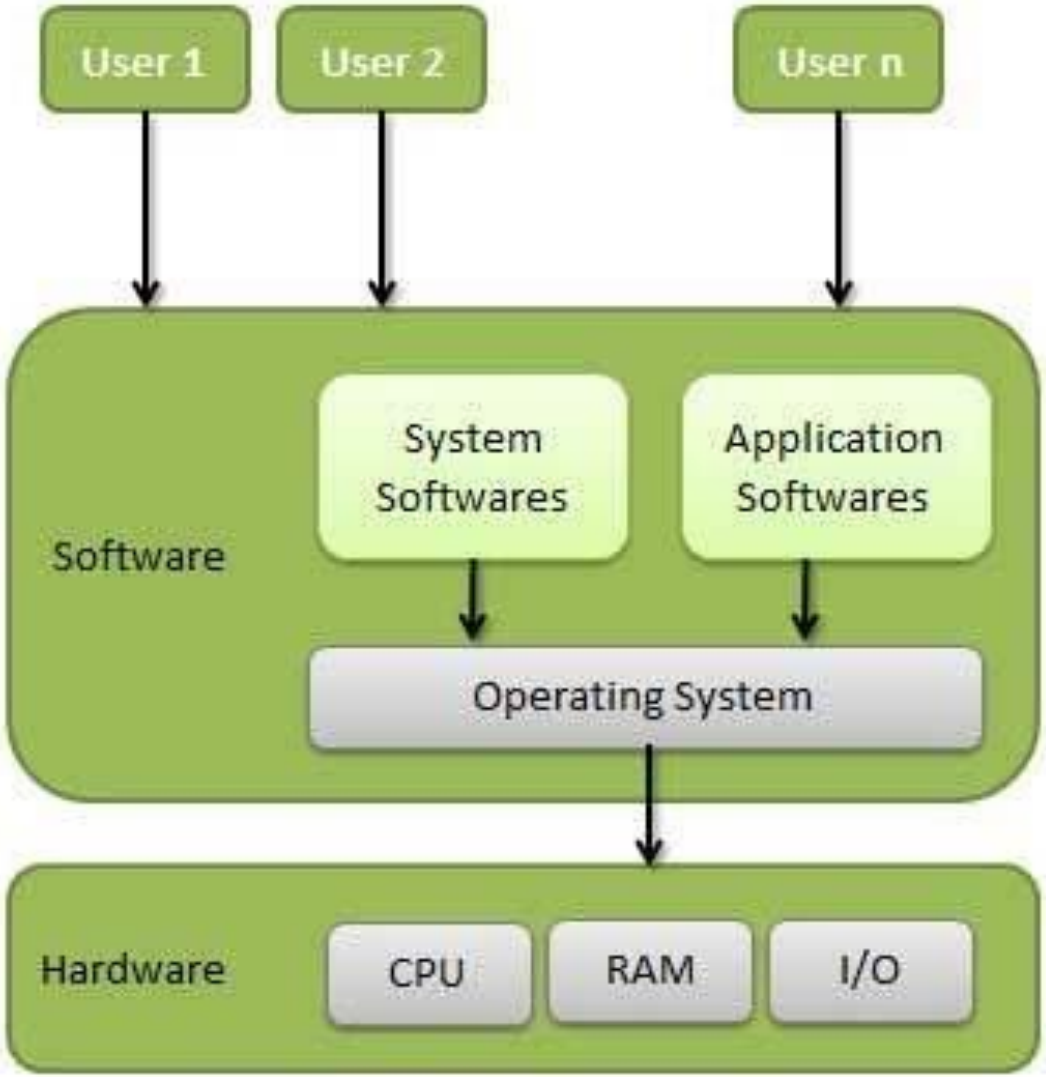
Computer Organization

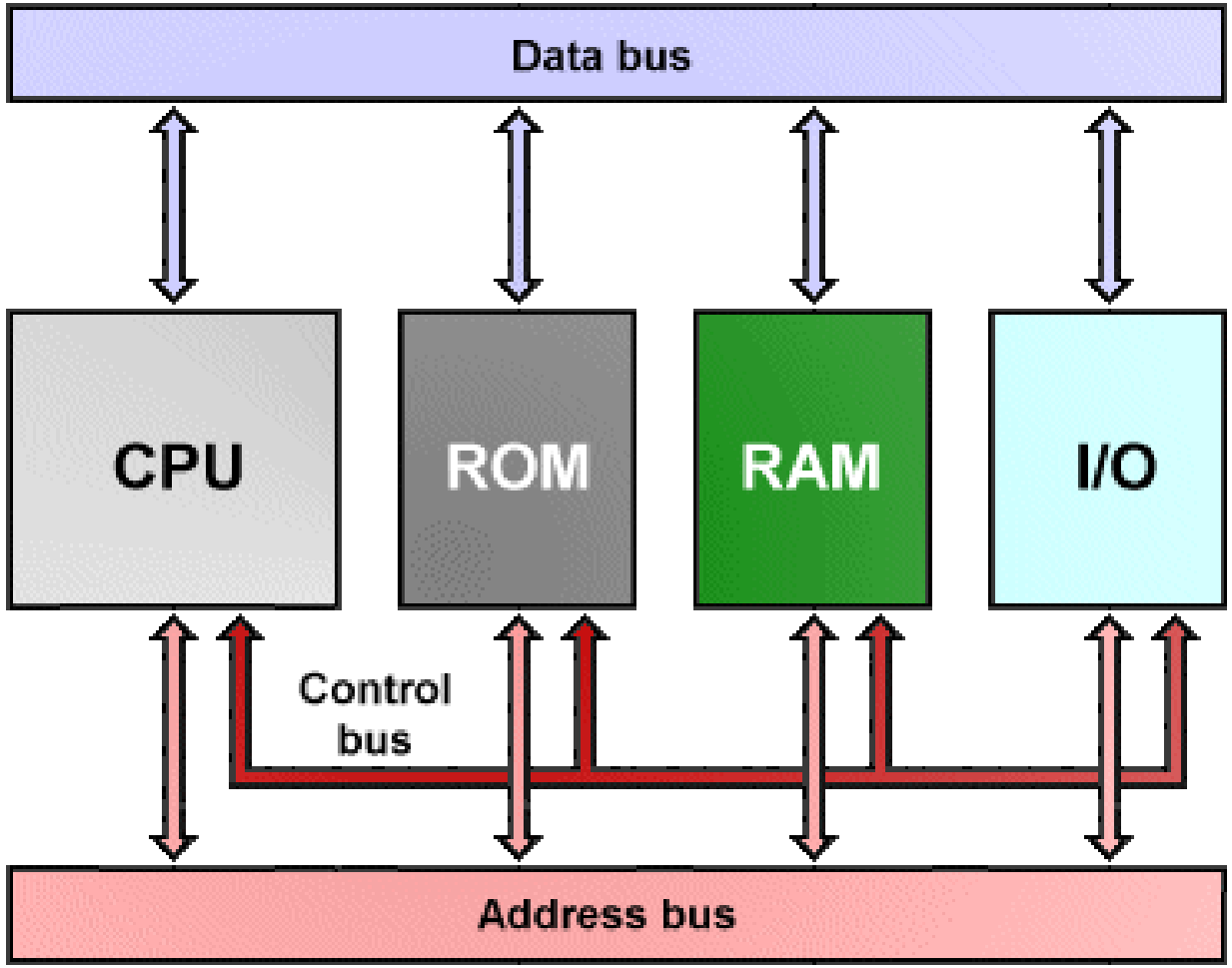
LECTURE 2



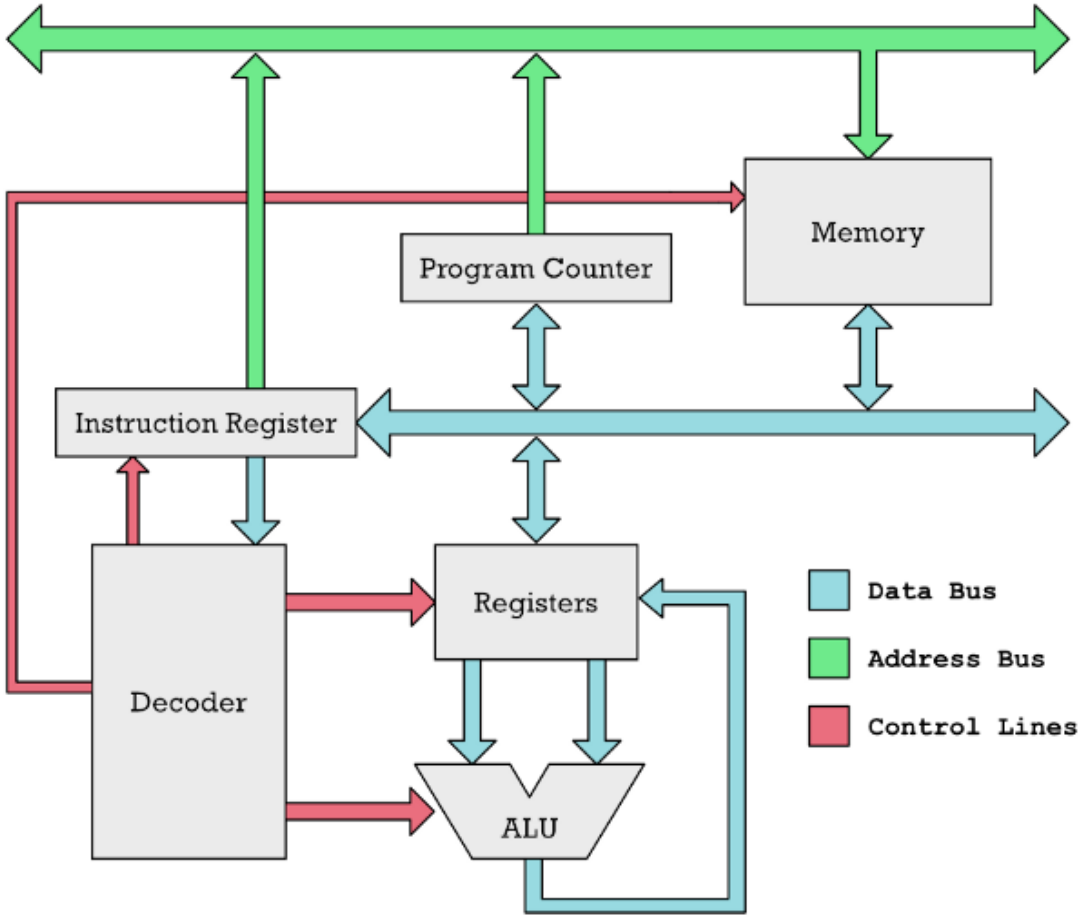
Basic Architecture Of a Computer

Computer Organization


User 1
User 2
User n
Software
System Softwares
Application Softwares
Operating System
Hardware
CPU
RAM
I/O



Computer System Architecture


Memory
Program Counter
Instruction Register
Registers
Decoder
ALU
Data Bus
Address Bus
Control Lines

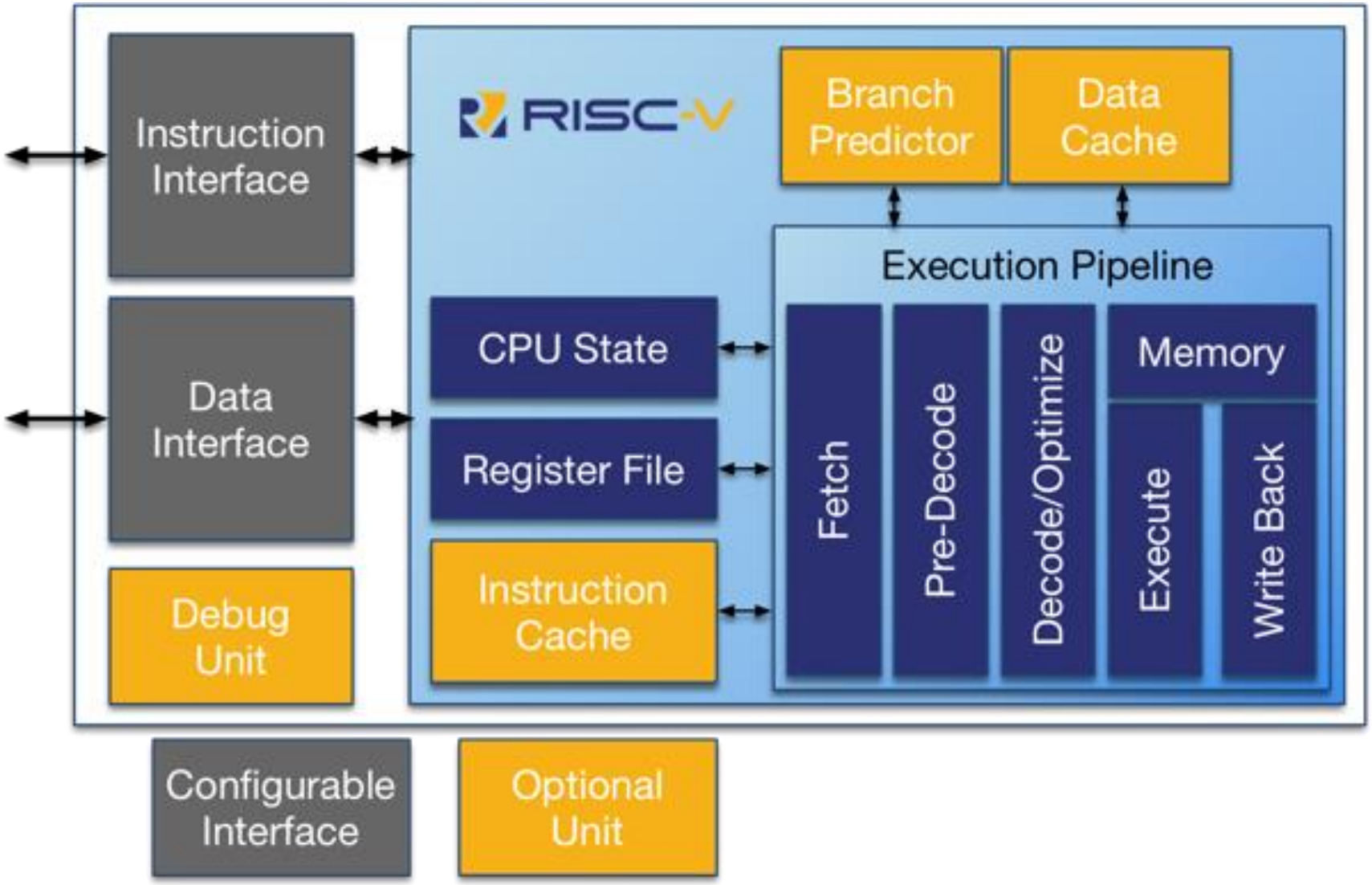
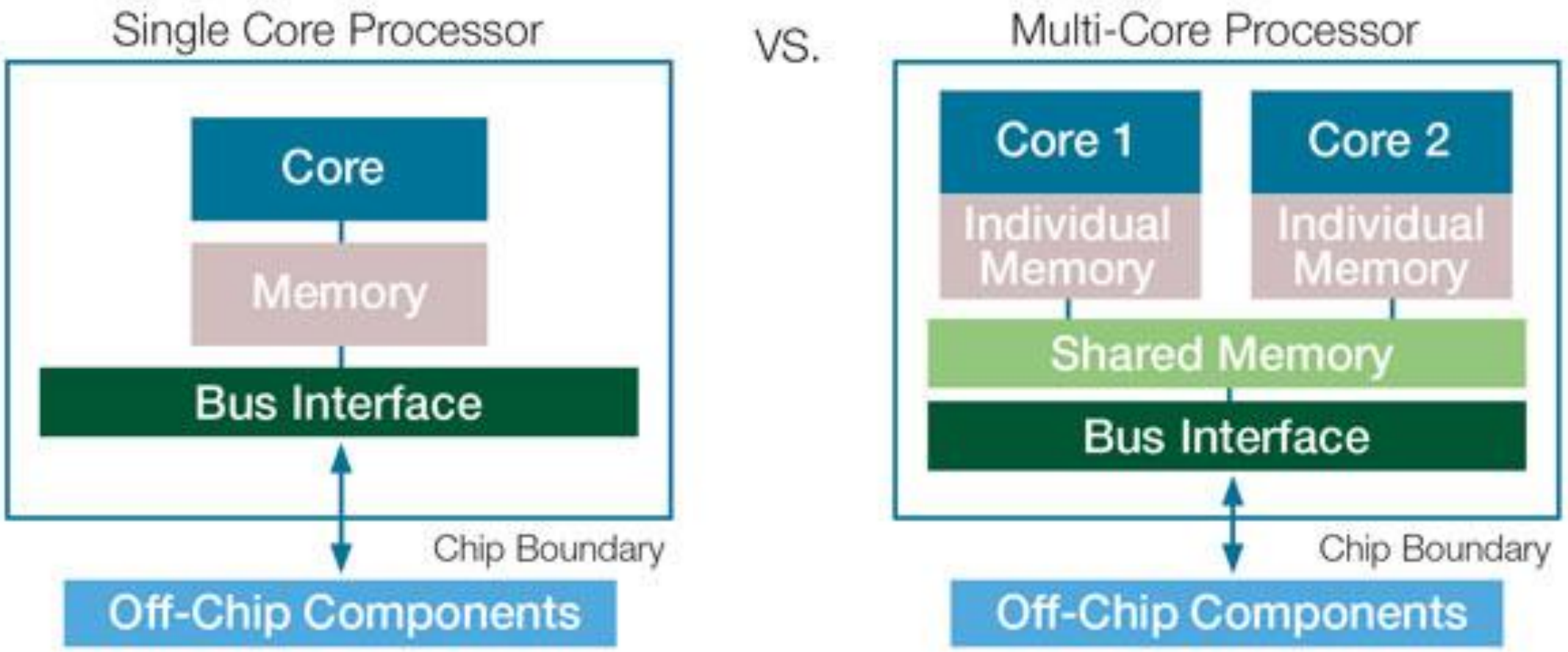
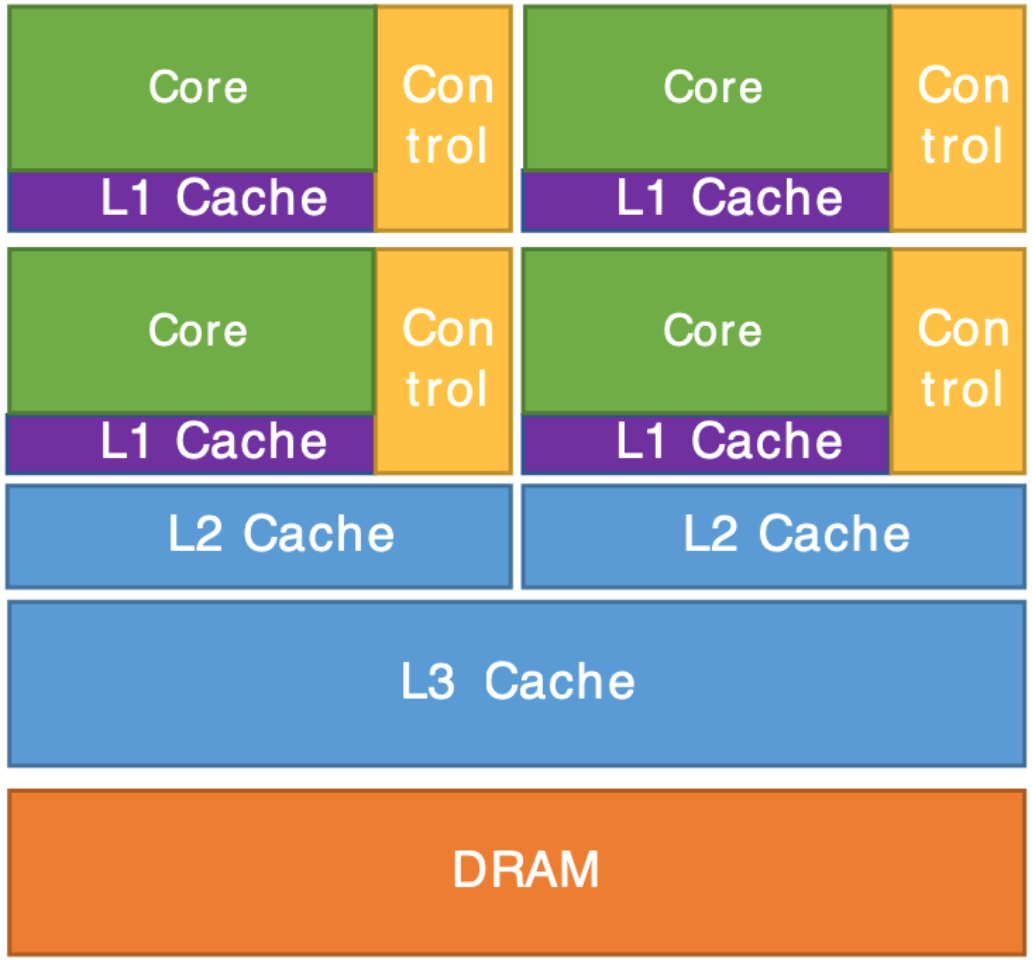

Instruction Interface
Data Interface
Debug Unit
RISC-V
Branch Predictor
Data Cache
Execution Pipeline
CPU State
Register File
Instruction Cache
Fetch
Pre-Decode
Decode/Optimize
Memory
Execute
Write Back
Configurable Interface
Optional Unit

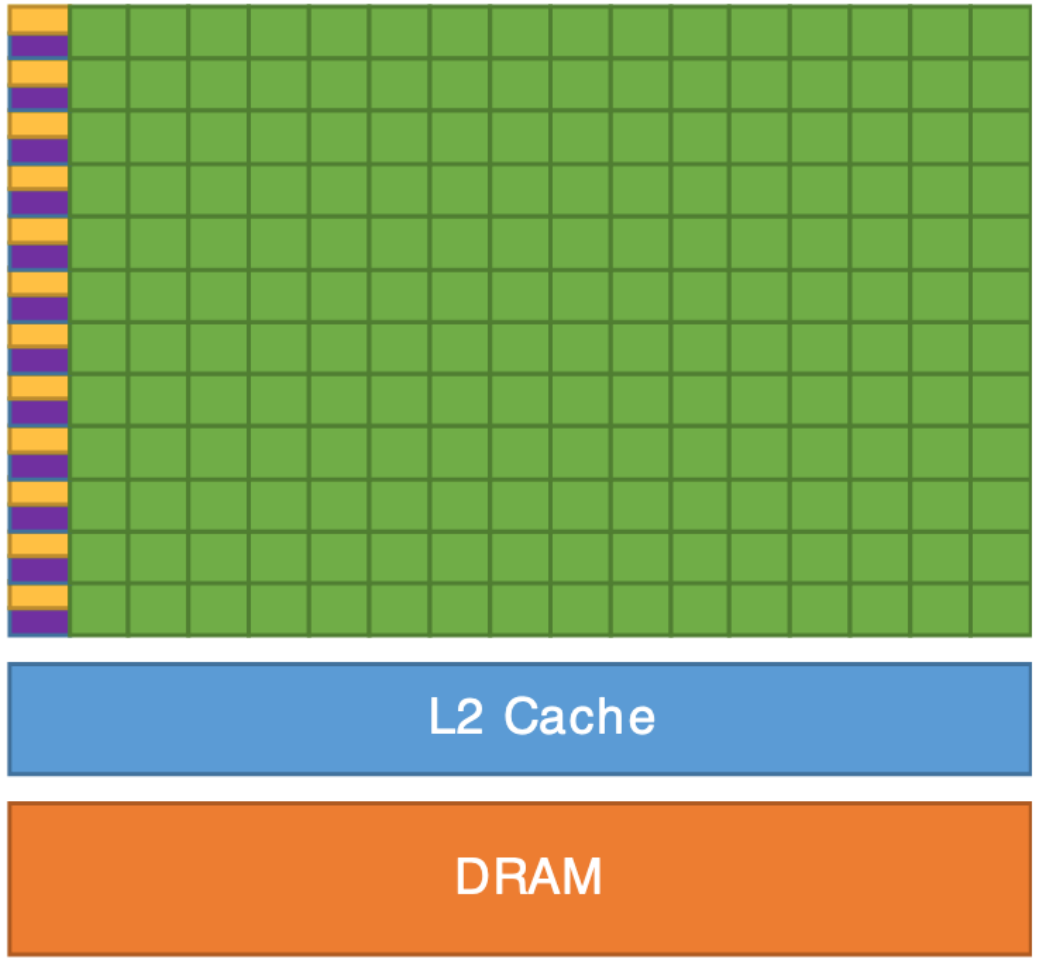
Table 1: Comparison of Superpipelining, Superscalar, and VLIW

	Superpipelining	**Superscalar**	**VLIW**
APPROACH & Effects	Dividing the long latency stages of a pipeline into several shorter stages Effects on *(clock cycle time)* of equation (1)	Dynamically issuing multiple instructions per cycle Effects on *(CPI)* of equation (1)	Utilizing the compiler to combine several instructions into one very long instruction and executing it Effects on (*IC*) of equation (1)
Instruction issue rate	1	N	M
Instruction issue style	Dynamic	Dynamic	Static
Difficulty of Design	Relatively easy compared to the other two	Complex design issues Run-time tracking	Complex design issues Compiler support
Keywords	Higher MHz, latch delay, clock skew	Dynamic scheduling, Out-of-Order execution, Register renaming, Score boarding	Compiler support, static scheduling, loop unrolling, trace scheduling

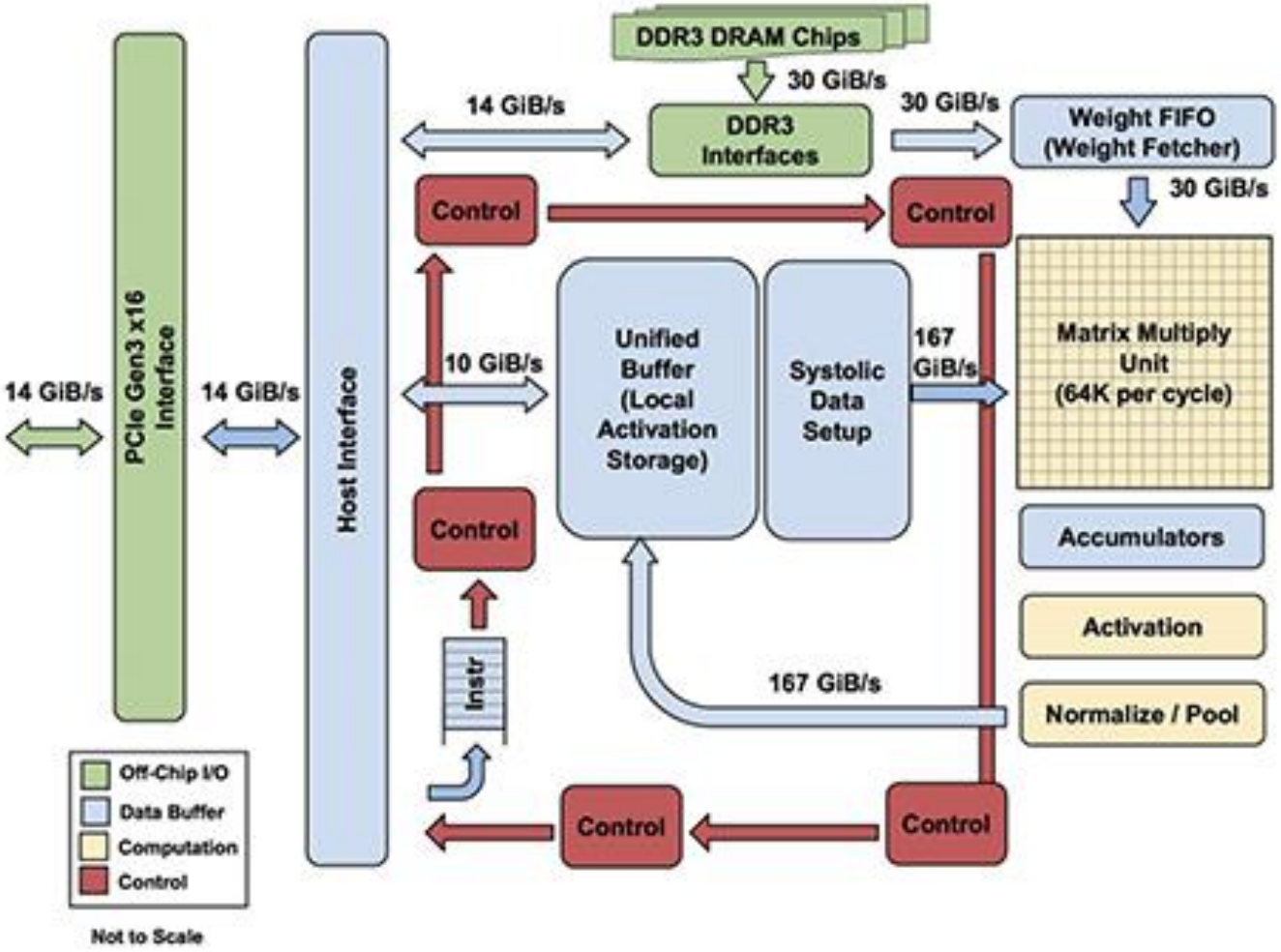

Single Core Processor
Core
Memory
Bus Interface
Chip Boundary
Off-Chip Components
VS.
Multi-Core Processor
Core 1
Core 2
Individual Memory
Individual Memory
Shared Memory
Bus Interface
Chip Boundary
Off-Chip Components

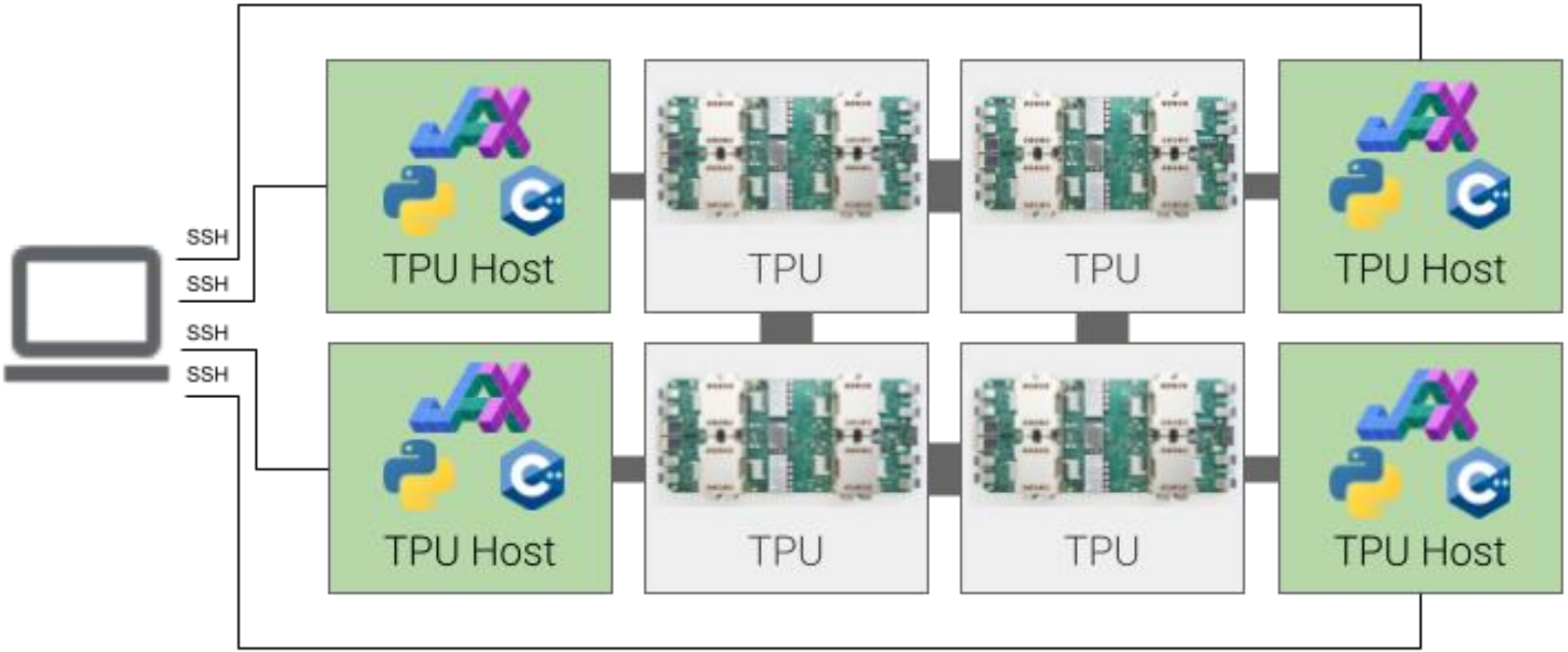

Core
Con trol
L1 Cache
Core
Con trol
L1 Cache
Core
Con trol
L1 Cache
Core
Con trol
L1 Cache
L2 Cache
L2 Cache
L3 Cache
DRAM

CPU


L2 Cache
DRAM

GPU


DDR3 DRAM Chips
30 GiB/s
14 GiB/s
DDR3 Interfaces
30 GiB/s
Weight FIFO (Weight Fetcher)
30 GiB/s
Control
Control
14 GiB/s
PCIe Gen3 x16 Interface
14 GiB/s
Host Interface
10 GiB/s
Unified Buffer (Local Activation Storage)
Systolic Data Setup
167 GiB/s
Matrix Multiply Unit (64K per cycle)
Control
Accumulators
Activation
Instr
167 GiB/s
Normalize / Pool
Off-Chip I/O
Data Buffer
Computation
Control
Control
Control
Not to Scale


SSH
SSH
SSH
SSH
TPU Host
TPU
TPU
TPU Host
TPU Host
TPU
TPU
TPU Host

Instruction Set Architecture

LECTURE 3

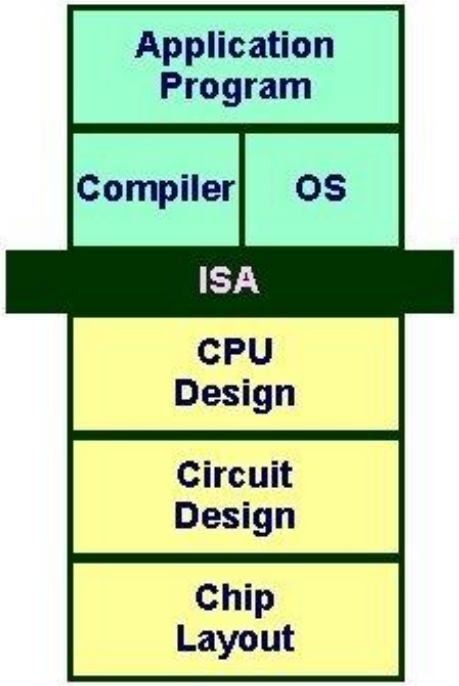
Instruction Set Architecture

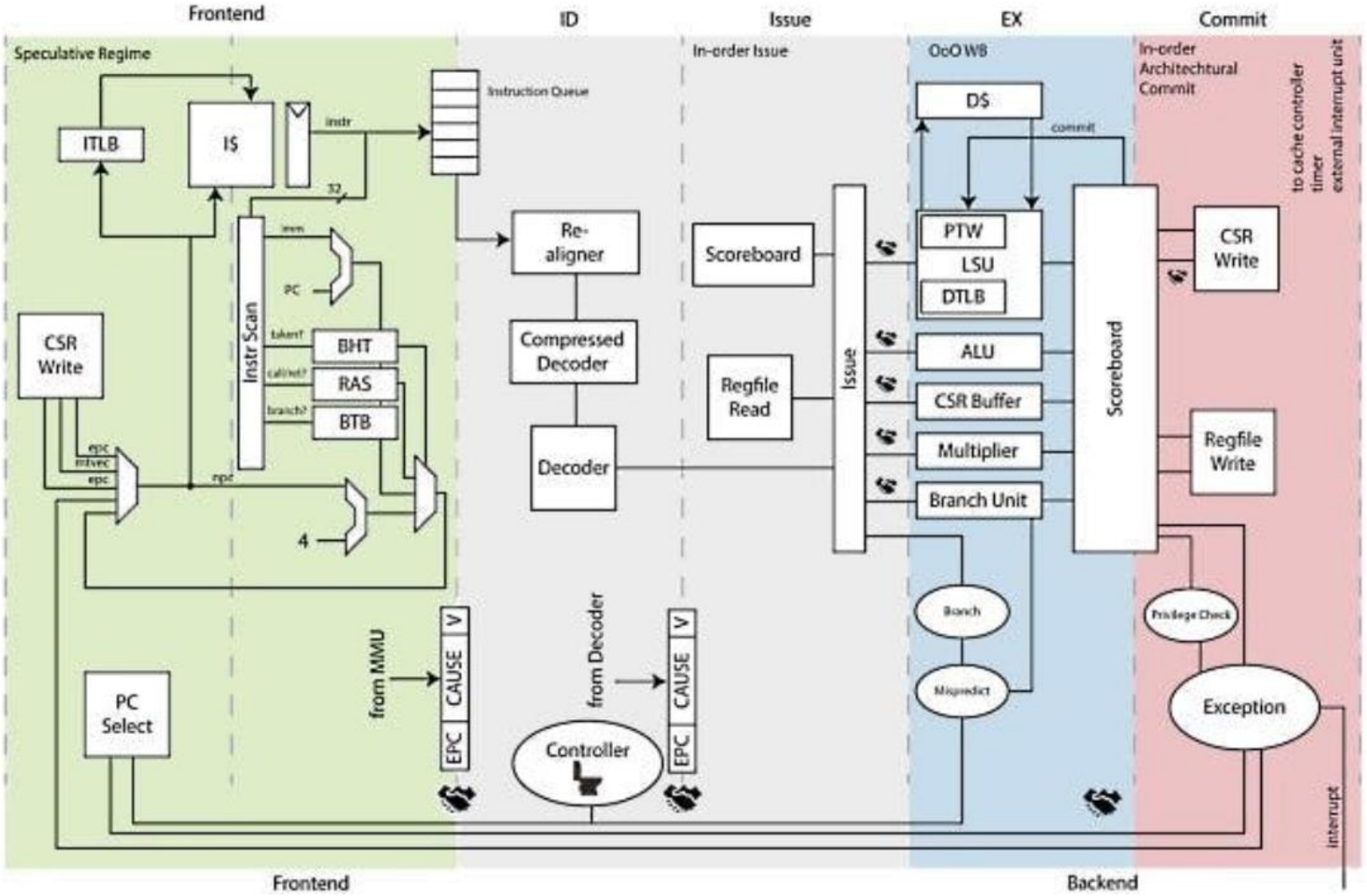
Assembly Language View

- Processor state
  - Registers, memory, …
- Instructions
  - `addq`, `pushq`, `ret`, …
  - How instructions are encoded as bytes

Layer of Abstraction

- Above: how to program machine
  - Processor executes instructions in a sequence
- Below: what needs to be built
  - Use variety of tricks to make it run fast
  - E.g., execute multiple instructions simultaneously





eC Learning Channel

Arithmetic Operation

Mnemonic	Instruction	Type	Description
ADD rd, rs1, rs2	Add	R	rd ← rs1 + rs2
SUB rd, rs1, rs2	Subtract	R	rd ← rs1 - rs2
ADDI rd, rs1, imm12	Add immediate	I	rd ← rs1 + imm12
SLT rd, rs1, rs2	Set less than	R	rd ← rs1 < rs2 ? 1 : 0
SLTI rd, rs1, imm12	Set less than immediate	I	rd ← rs1 < imm12 ? 1 : 0
SLTU rd, rs1, rs2	Set less than unsigned	R	rd ← rs1 < rs2 ? 1 : 0
SLTIU rd, rs1, imm12	Set less than immediate unsigned	I	rd ← rs1 < imm12 ? 1 : 0
LUI rd, imm20	Load upper immediate	U	rd ← imm20 << 12
AUIP rd, imm20	Add upper immediate to PC	U	rd ← PC + imm20 << 12

Logical Operations

Mnemonic	Instruction	Type	Description
AND rd, rs1, rs2	AND	R	rd ← rs1 & rs2
OR rd, rs1, rs2	OR	R	rd ← rs1 | rs2
XOR rd, rs1, rs2	XOR	R	rd ← rs1 ^ rs2
ANDI rd, rs1, imm12	AND immediate	I	rd ← rs1 & imm12
ORI rd, rs1, imm12	OR immediate	I	rd ← rs1 | imm12
XORI rd, rs1, imm12	XOR immediate	I	rd ← rs1 ^ imm12
SLL rd, rs1, rs2	Shift left logical	R	rd ← rs1 << rs2
SRL rd, rs1, rs2	Shift right logical	R	rd ← rs1 >> rs2
SRA rd, rs1, rs2	Shift right arithmetic	R	rd ← rs1 >> rs2
SLLI rd, rs1, shamt	Shift left logical immediate	I	rd ← rs1 << shamt
SRLI rd, rs1, shamt	Shift right logical imm.	I	rd ← rs1 >> shamt
SRAI rd, rs1, shamt	Shift right arithmetic immediate	I	rd ← rs1 >> shamt

32-bit instruction format

	31 30 29 28 27 26 25	24 23 22 21 20	19 18 17 16 15	14 13 12	11 10 9 8 7	6 5 4 3 2 1 0
R	func	rs2	rs1	func	rd	opcode
I	immediate		rs1	func	rd	opcode
SB	immediate	rs2	rs1	func	immediate	opcode
UJ	immediate				rd	opcode

Load / Store Operations

Mnemonic	Instruction	Type	Description
LD rd, imm12(rs1)	Load doubleword	I	rd ← mem[rs1 + imm12]
LW rd, imm12(rs1)	Load word	I	rd ← mem[rs1 + imm12]
LH rd, imm12(rs1)	Load halfword	I	rd ← mem[rs1 + imm12]
LB rd, imm12(rs1)	Load byte	I	rd ← mem[rs1 + imm12]
LWU rd, imm12(rs1)	Load word unsigned	I	rd ← mem[rs1 + imm12]
LHU rd, imm12(rs1)	Load halfword unsigned	I	rd ← mem[rs1 + imm12]
LBU rd, imm12(rs1)	Load byte unsigned	I	rd ← mem[rs1 + imm12]
SD rs2, imm12(rs1)	Store doubleword	S	rs2 → mem[rs1 + imm12]
SW rs2, imm12(rs1)	Store word	S	rs2(31:0) → mem[rs1 + imm12]
SH rs2, imm12(rs1)	Store halfword	S	rs2(15:0) → mem[rs1 + imm12]
SB rs2, imm12(rs1)	Store byte	S	rs2(7:0) → em[rs1 + imm12]

Branching

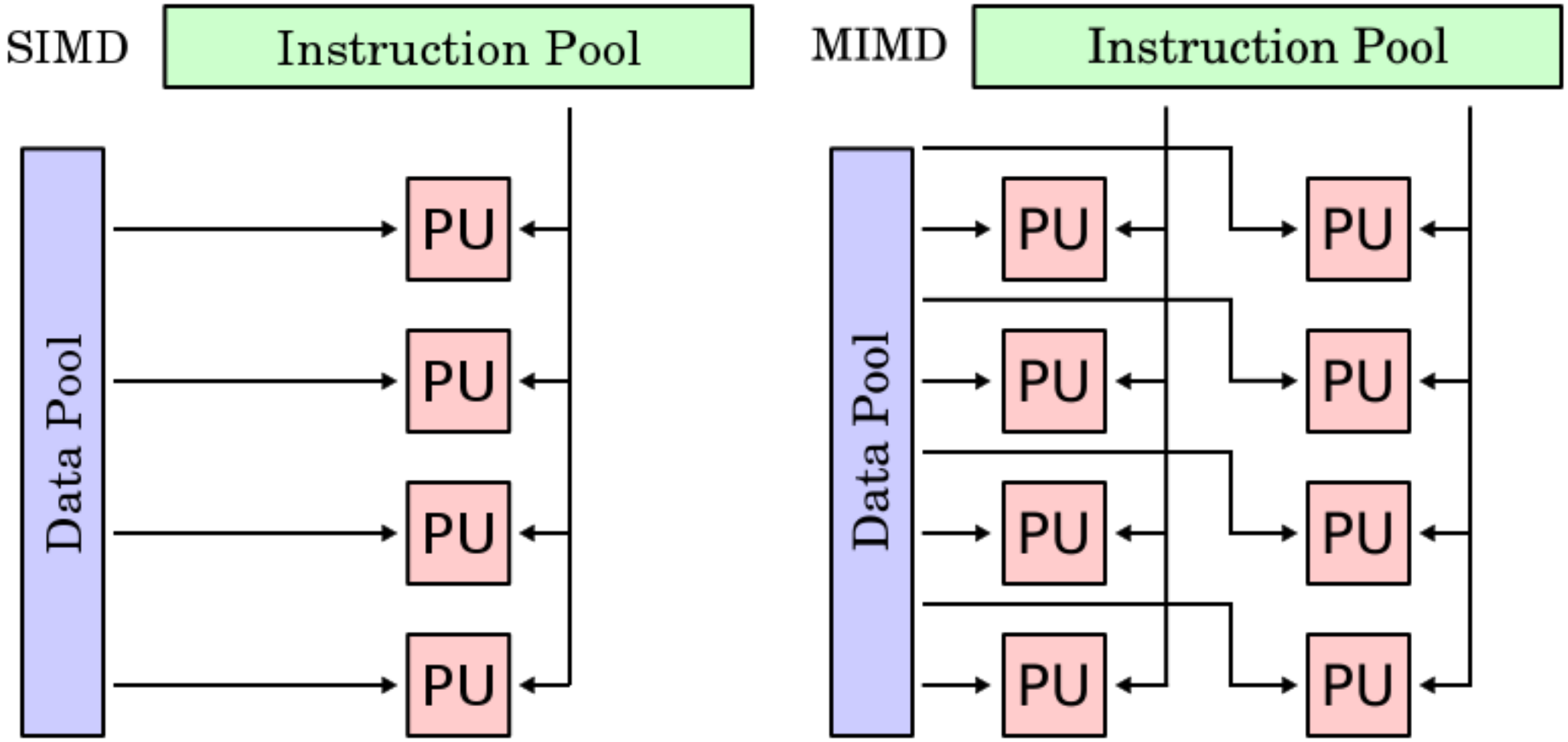
Mnemonic	Instruction	Type	Description
`BEQ rs1, rs2, imm12`	Branch equal	SB	`if rs1 == rs2` `pc ← pc + imm12`
`BNE rs1, rs2, imm12`	Branch not equal	SB	`if rs1 != rs2` `pc ← pc + imm12`
`BGE rs1, rs2, imm12`	Branch greater than or equal	SB	`if rs1 >= rs2` `pc ← pc + imm12`
`BGEU rs1, rs2, imm12`	Branch greater than or equal unsigned	SB	`if rs1 >= rs2` `pc ← pc + imm12`
`BLT rs1, rs2, imm12`	Branch less than	SB	`if rs1 < rs2` `pc ← pc + imm12`
`BLTU rs1, rs2, imm12`	Branch less than unsigned	SB	`if rs1 < rs2` `pc ← pc + imm12 << 1`
`JAL rd, imm20`	Jump and link	UJ	`rd ← pc + 4` `pc ← pc + imm20`
`JALR rd, imm12(rs1)`	Jump and link register	I	`rd ← pc + 4` `pc ← rs1 + imm12`

Parallelism

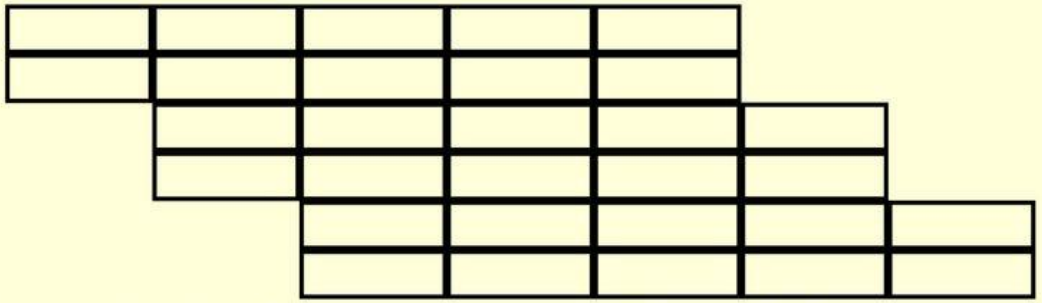
LECTURE 4

Parallelism

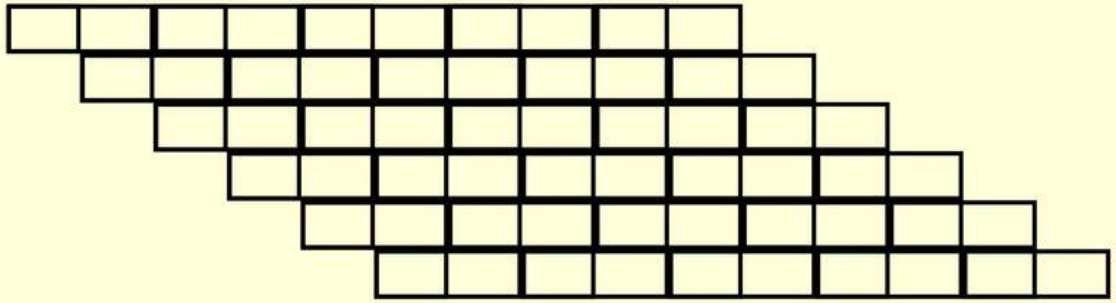
- Data parallelism
- Instruction Level Parallelism
- Task Level Parallelism – Multi-threading/Multi-processing
- Multi-programming


SIMD
Instruction Pool
Data Pool
PU
PU
PU
PU
MIMD
Instruction Pool
Data Pool
PU
PU
PU
PU
PU
PU
PU
PU

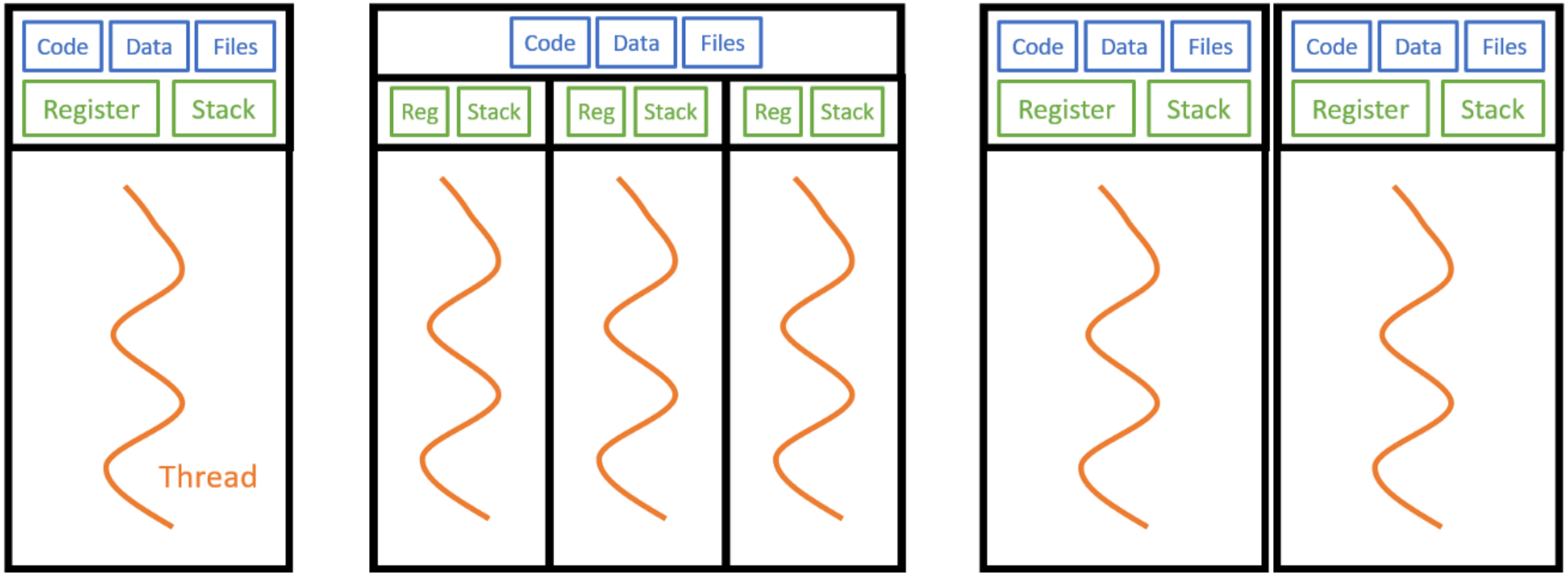
Superscalar vs. superpipelined



(multiple instructions in the same stage, same CR as scalar)



(more total stages, faster clock rate)

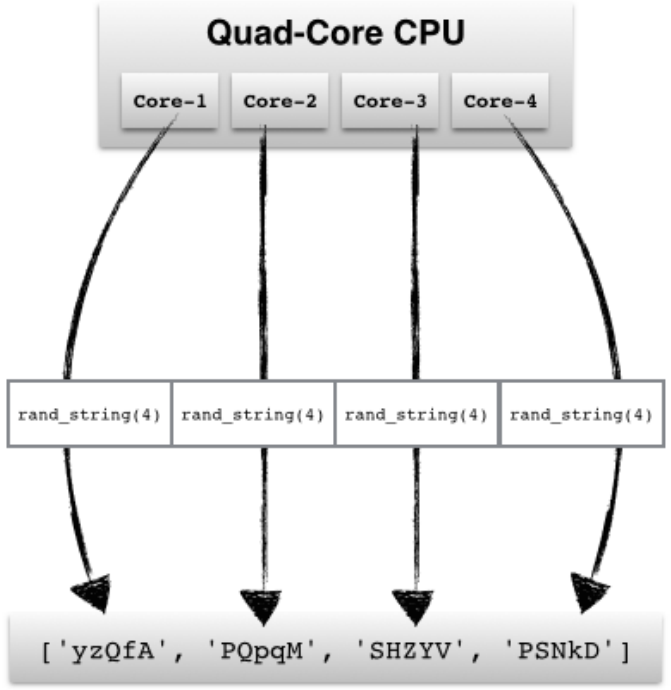


Single Processor Single Thread

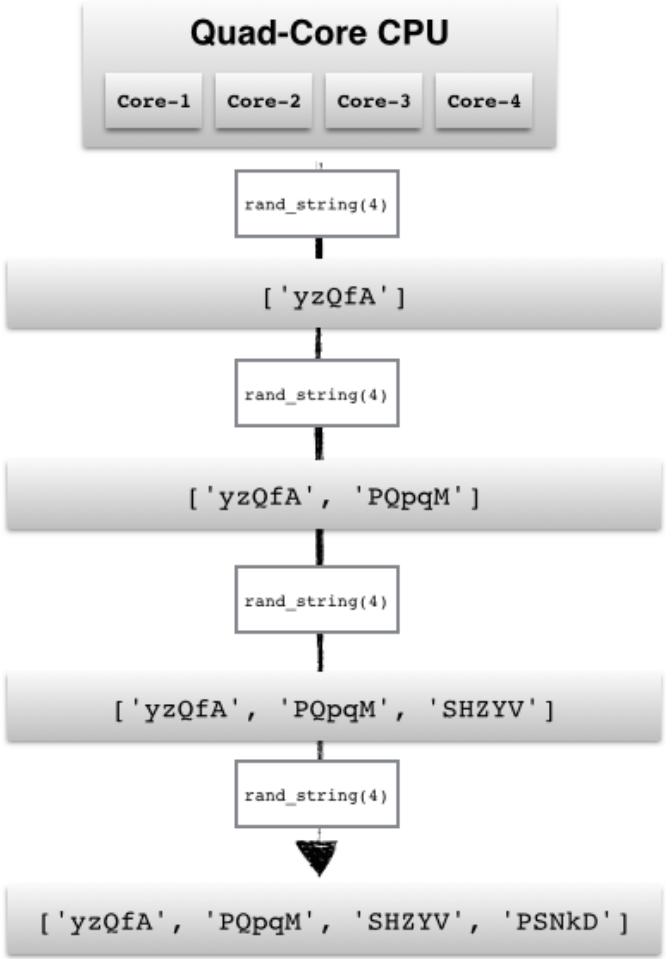
Single Processor Multithread

Multiprocessing

[parallel processing]



[serial processing]



Multi-Programming

Vs

Multi-Tasking

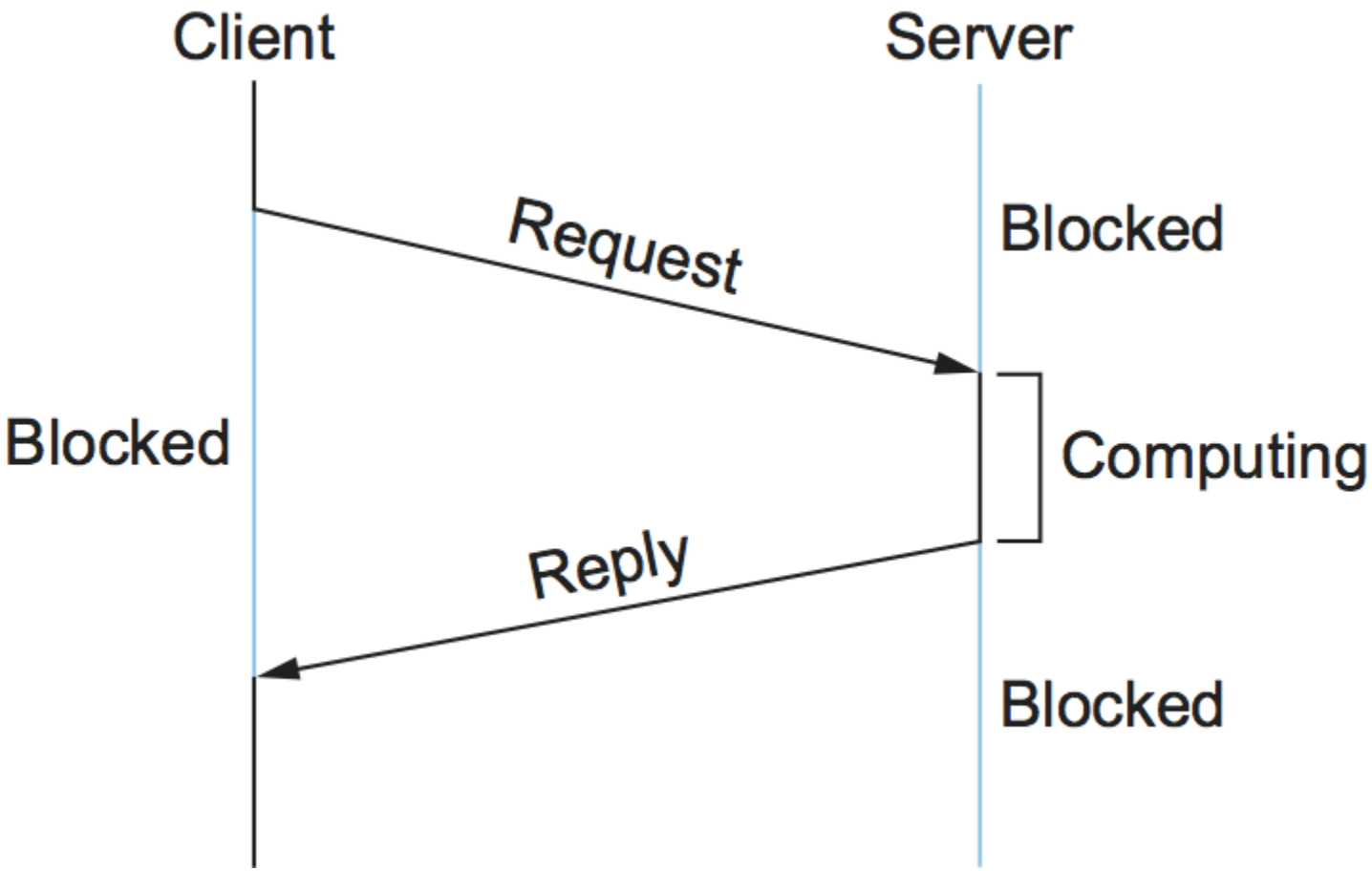
Vs

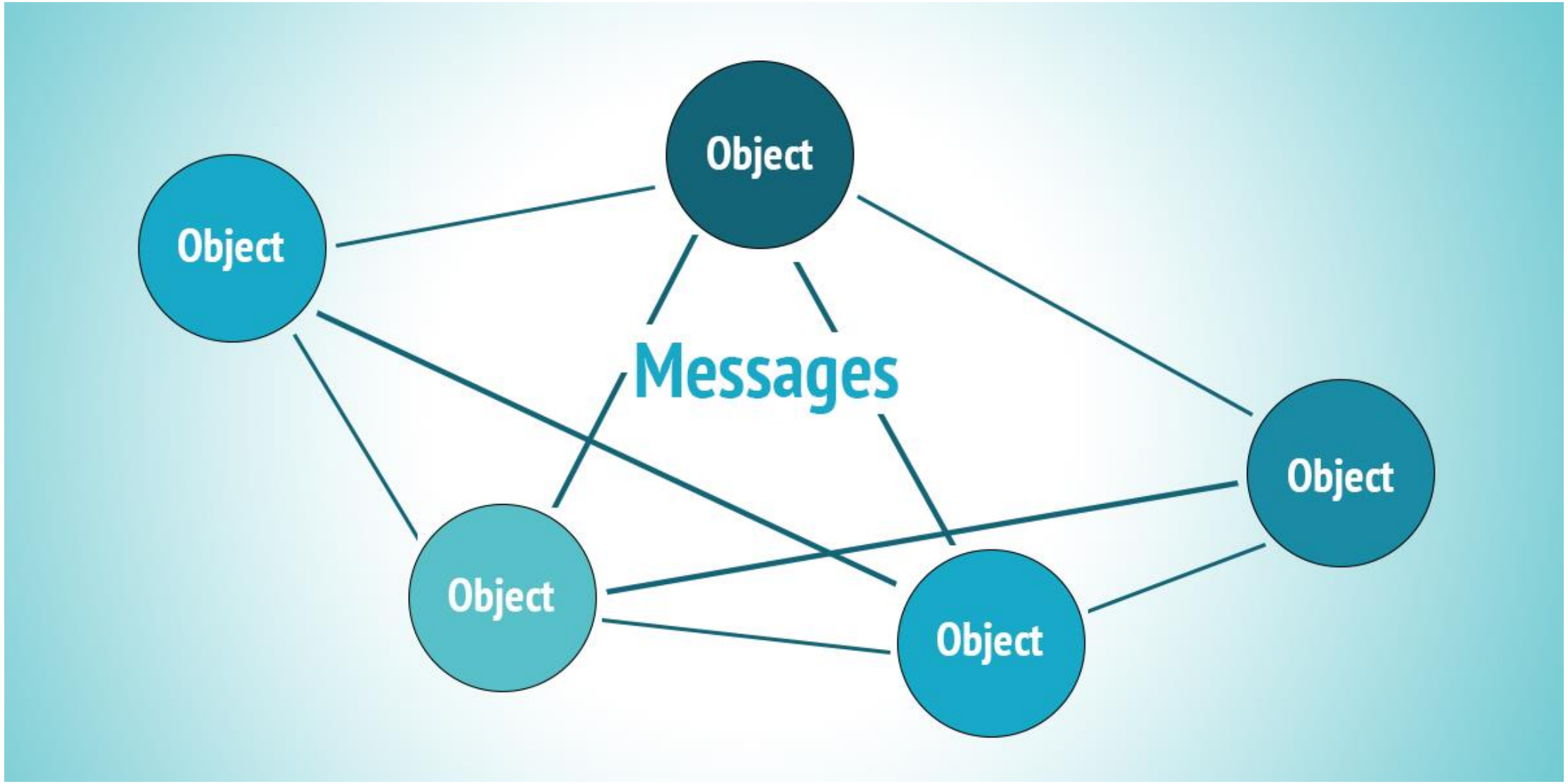
Multi-Processing

Multiprogramming v/s Multiprocessing v/s Multitasking

Multiprogramming	Multiprocessing	Multitasking
The concurrent residency of more than one program in the main memory is called as multiprogramming.	The availability of more than one processor per system, which can execute several set of instructions in parallel is called as multiprocessing.	The execution of more than one task simultaneously is called as multitasking.
Number of processor: one	Number of processor: more than one	Number of processor: one
One process is executed at a time.	More than one process can be executed at a time.	One by one job is being executed at a time.

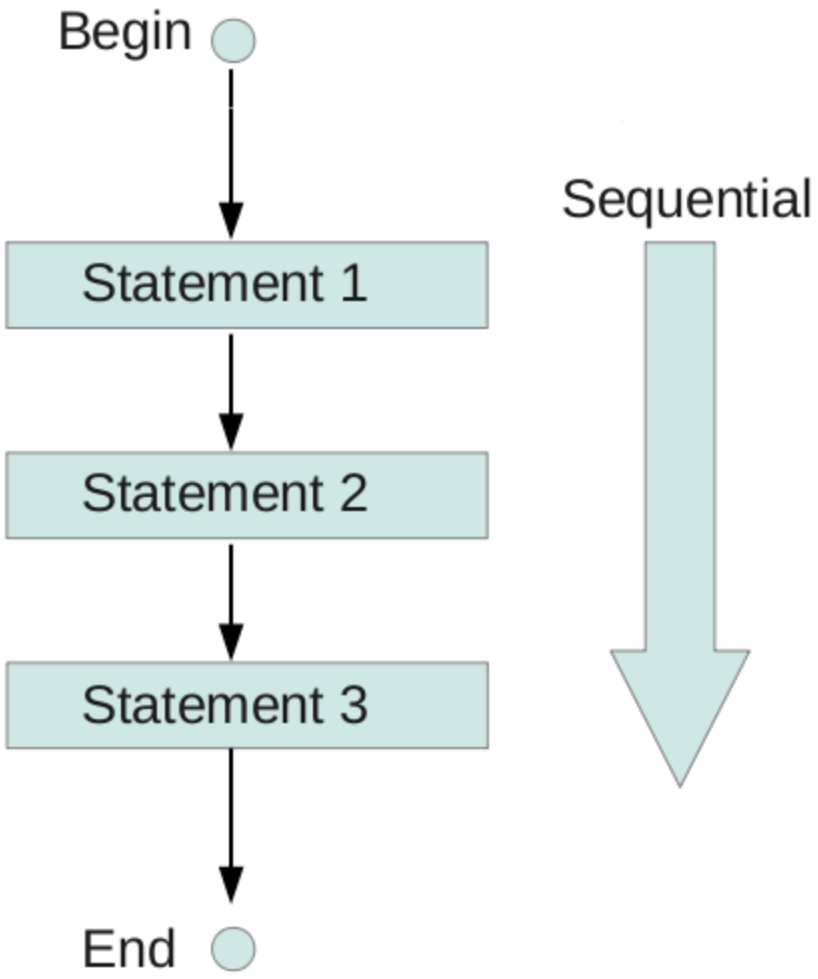
Remote Procedural Calls

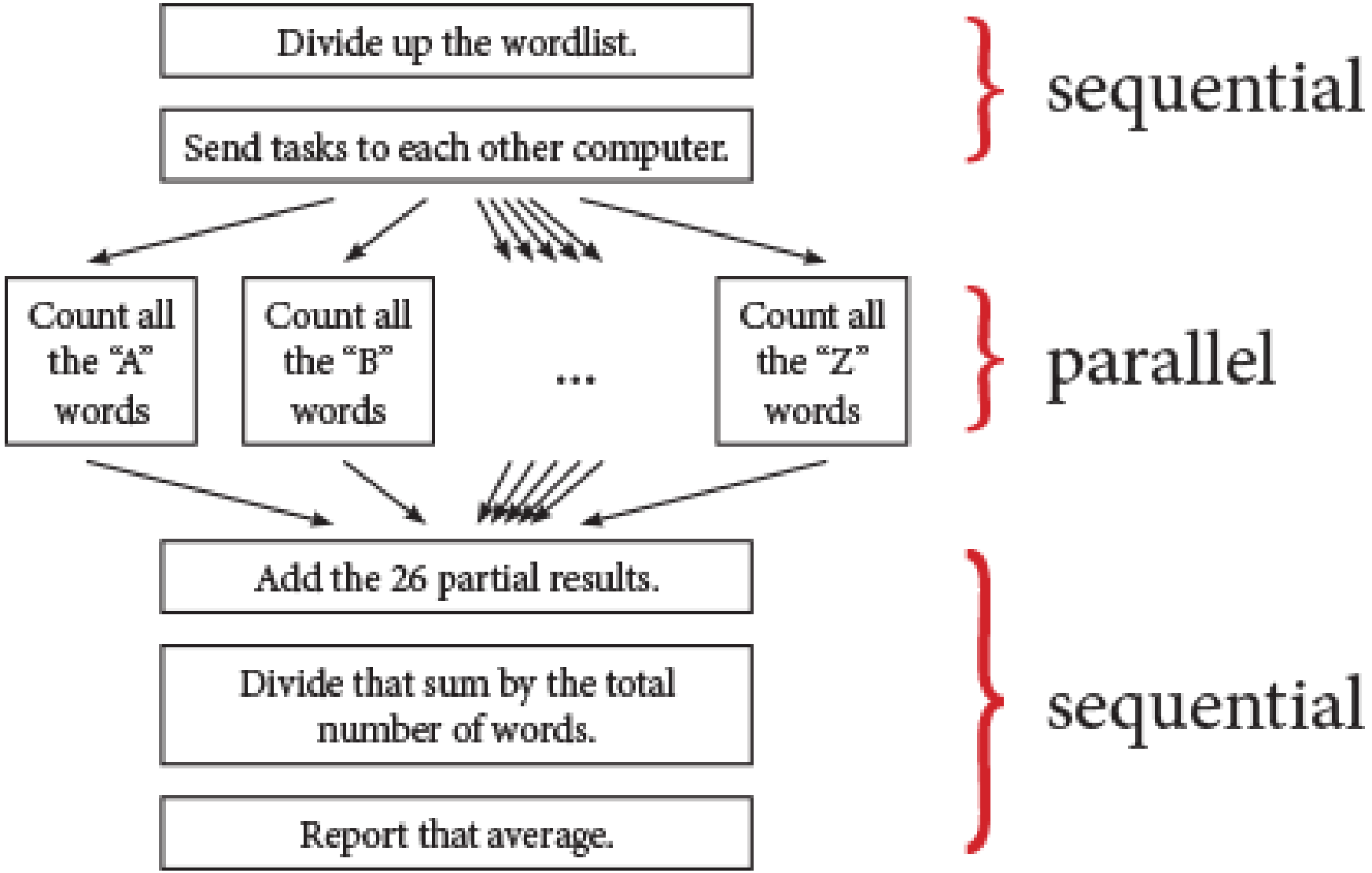



Object
Object
Messages
Object
Object
Object

Computing Model – Sequential Computing

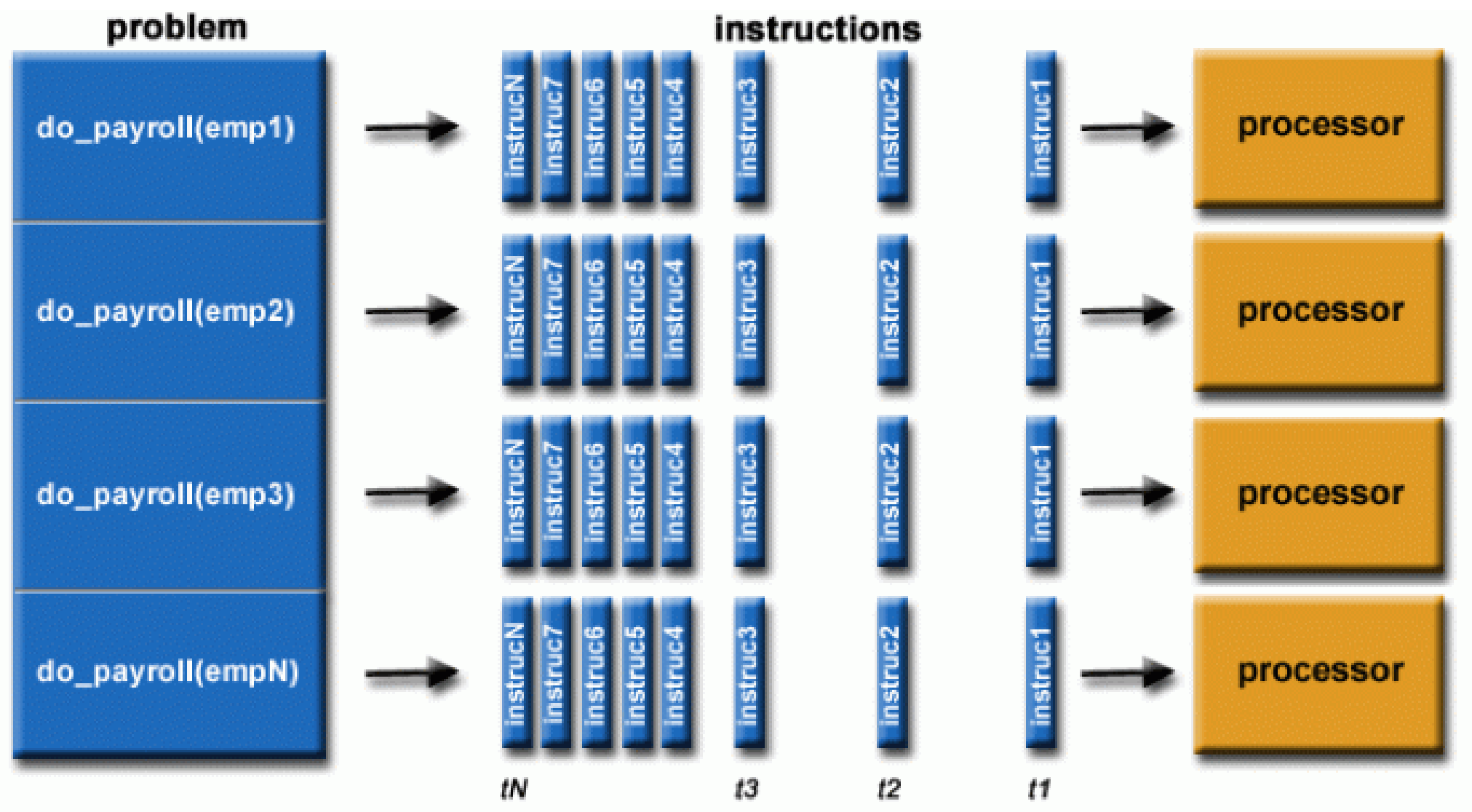
LECTURE 5

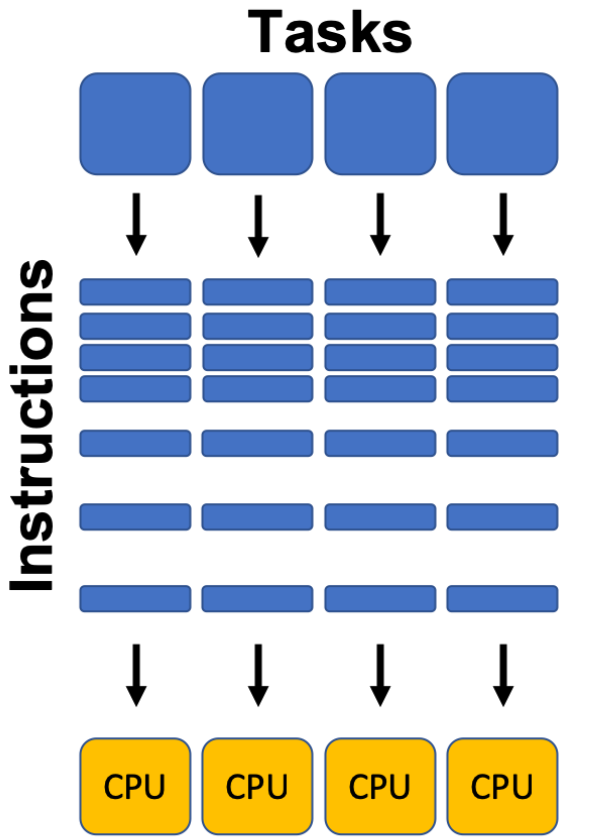

Begin
Statement 1
Statement 2
Statement 3
End
Sequential


Divide up the wordlist.
Send tasks to each other computer.
Count all the "A" words
Count all the "B" words
...
Count all the "Z" words
Add the 26 partial results.
Divide that sum by the total number of words.
Report that average.
sequential
parallel
sequential

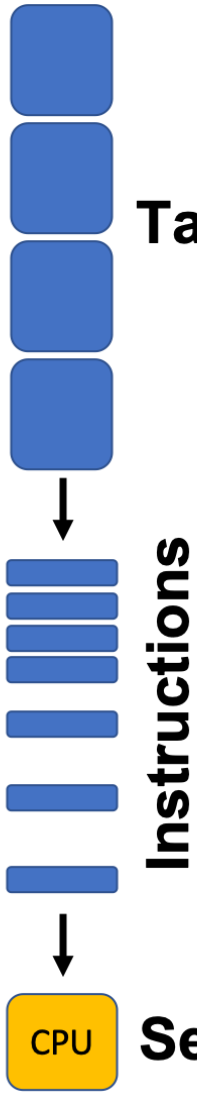
Computing Model – Parallel Computing

LECTURE 6


problem
instructions
do_payroll(emp1)
do_payroll(emp2)
do_payroll(emp3)
do_payroll(empN)
instrucN
instruc7
instruc6
instruc5
instruc4
instruc3
instruc2
instruc1
processor
tN
t3
t2
t1

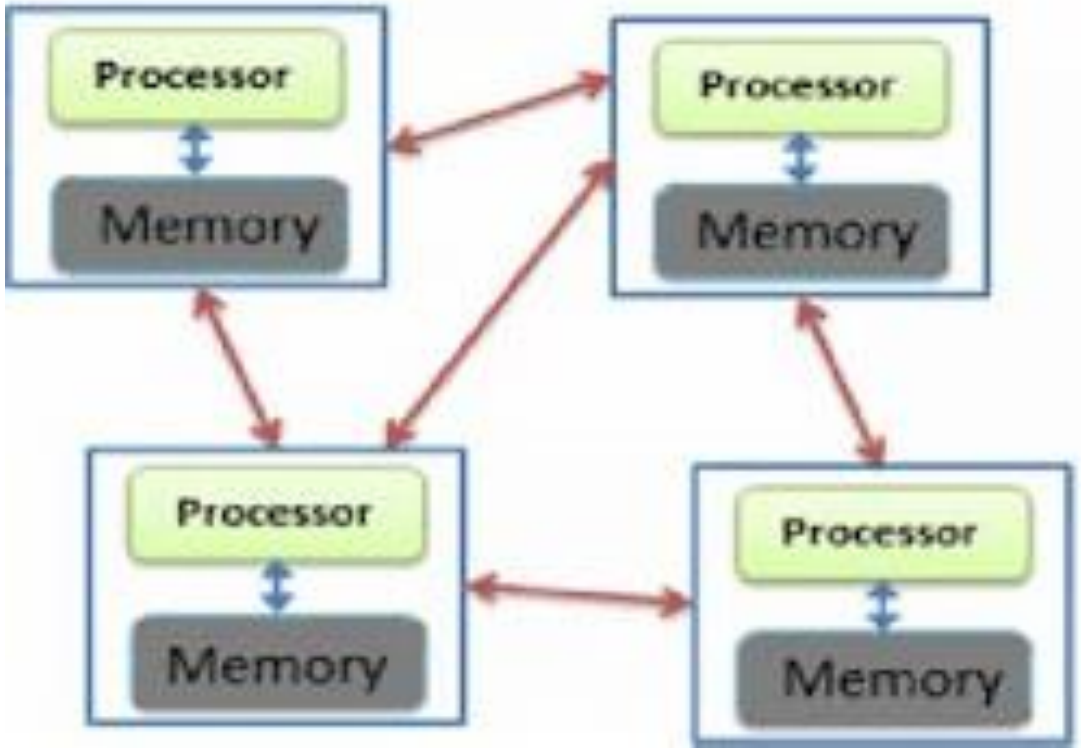


Parallel computing

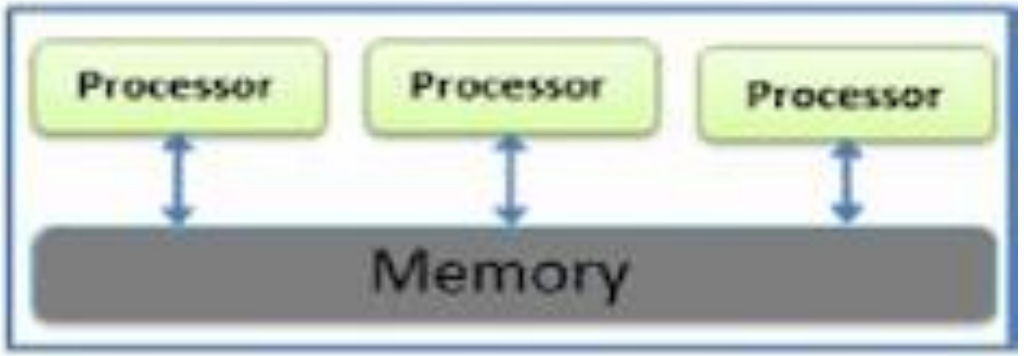


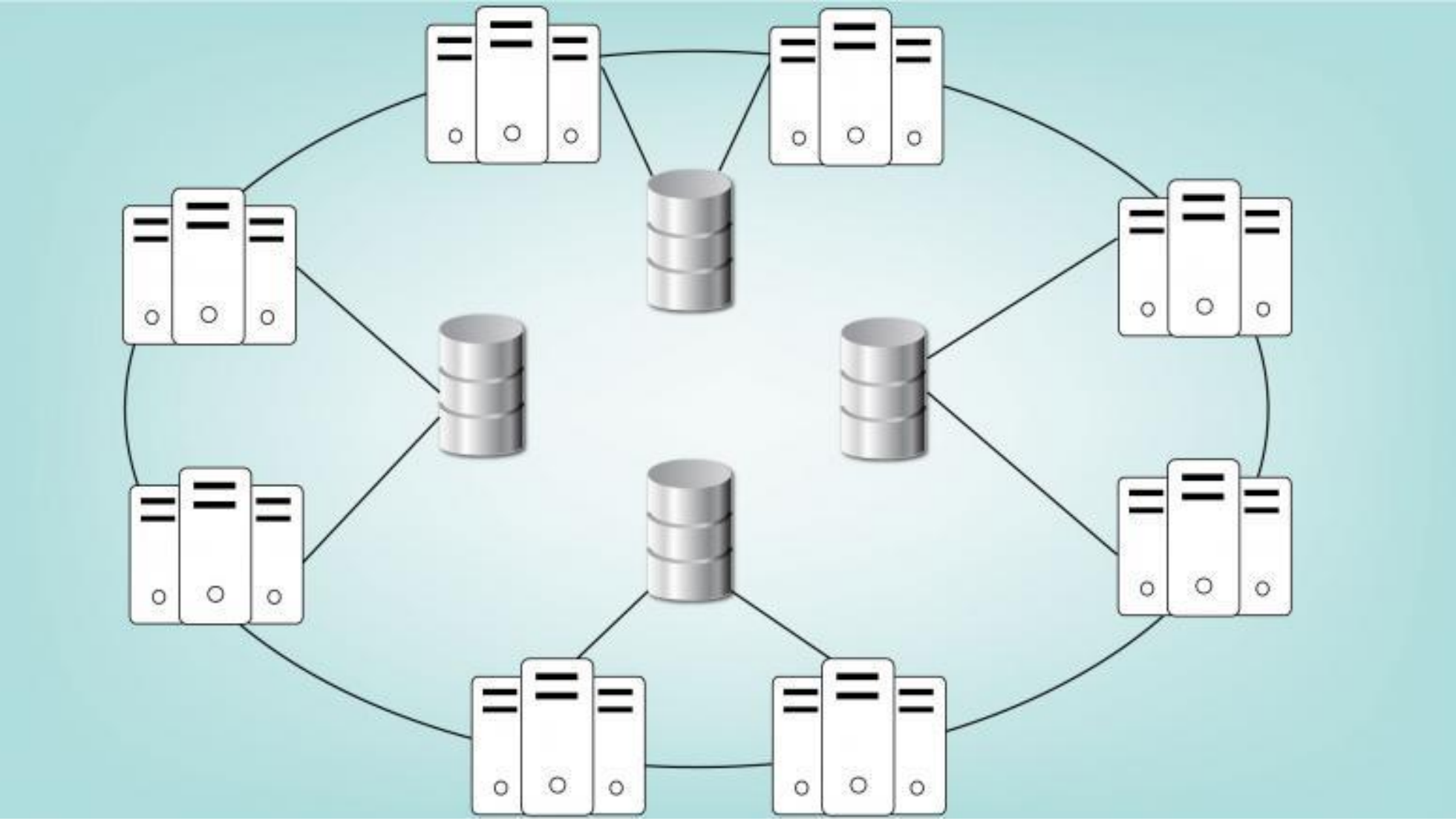
Serial computing

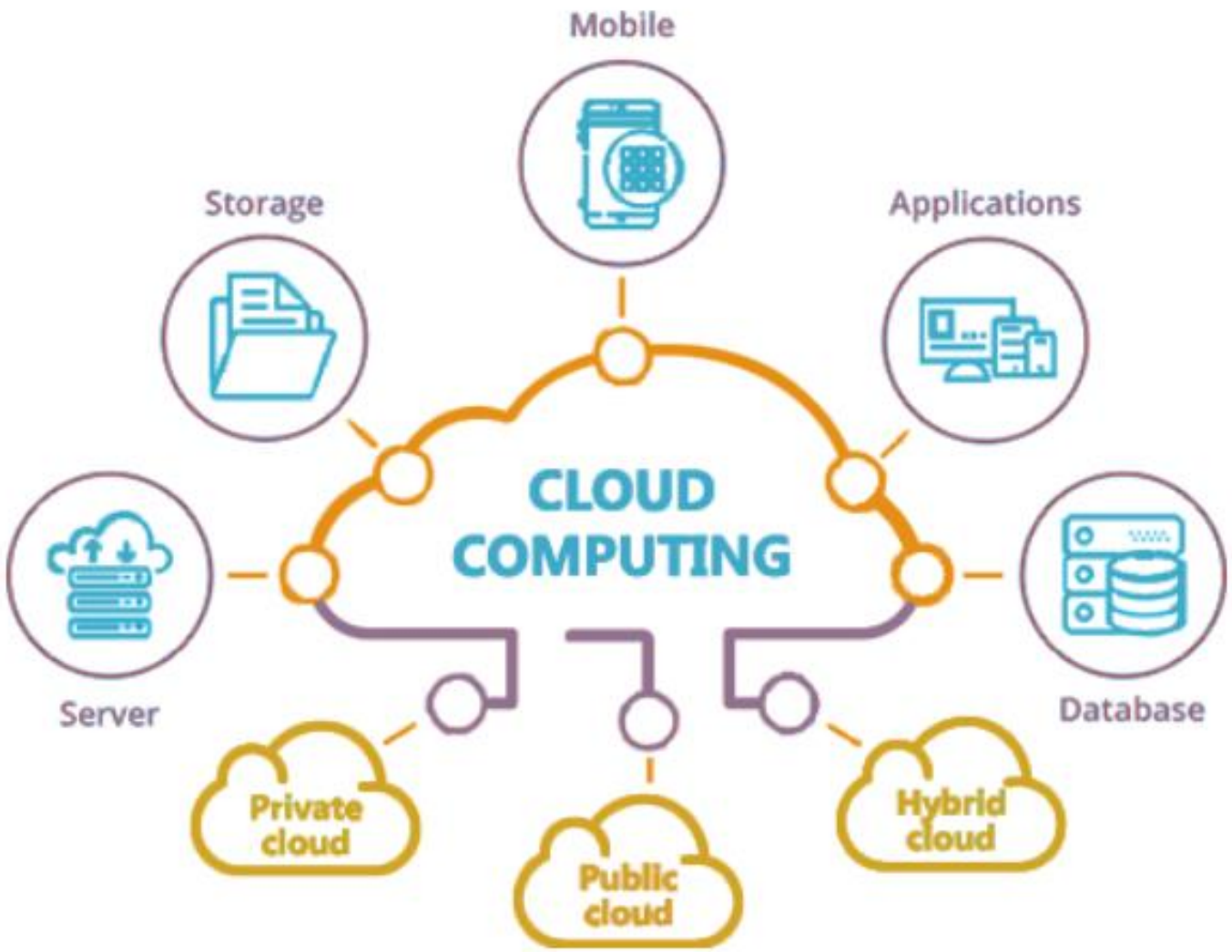
Distributed Computing



Parallel Computing

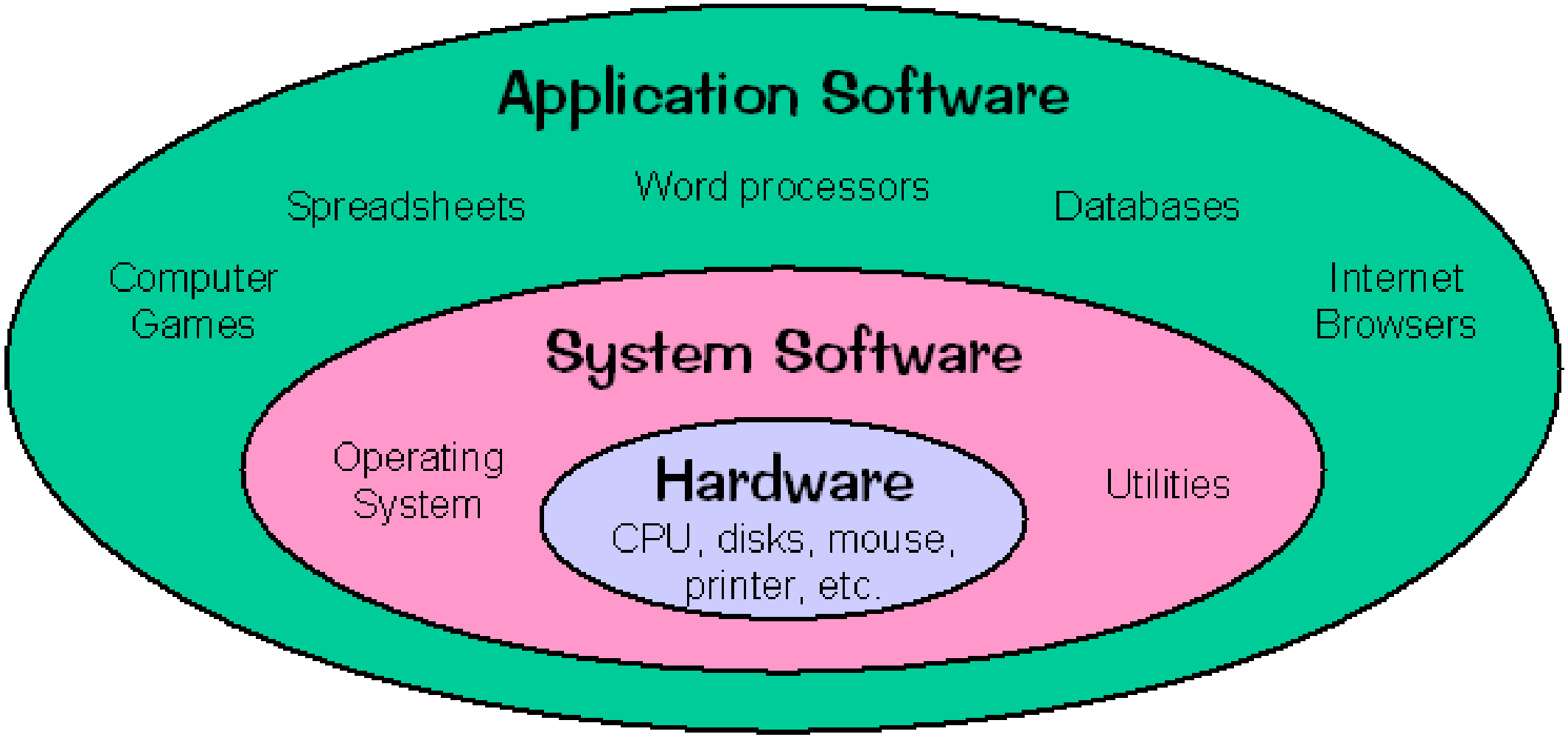


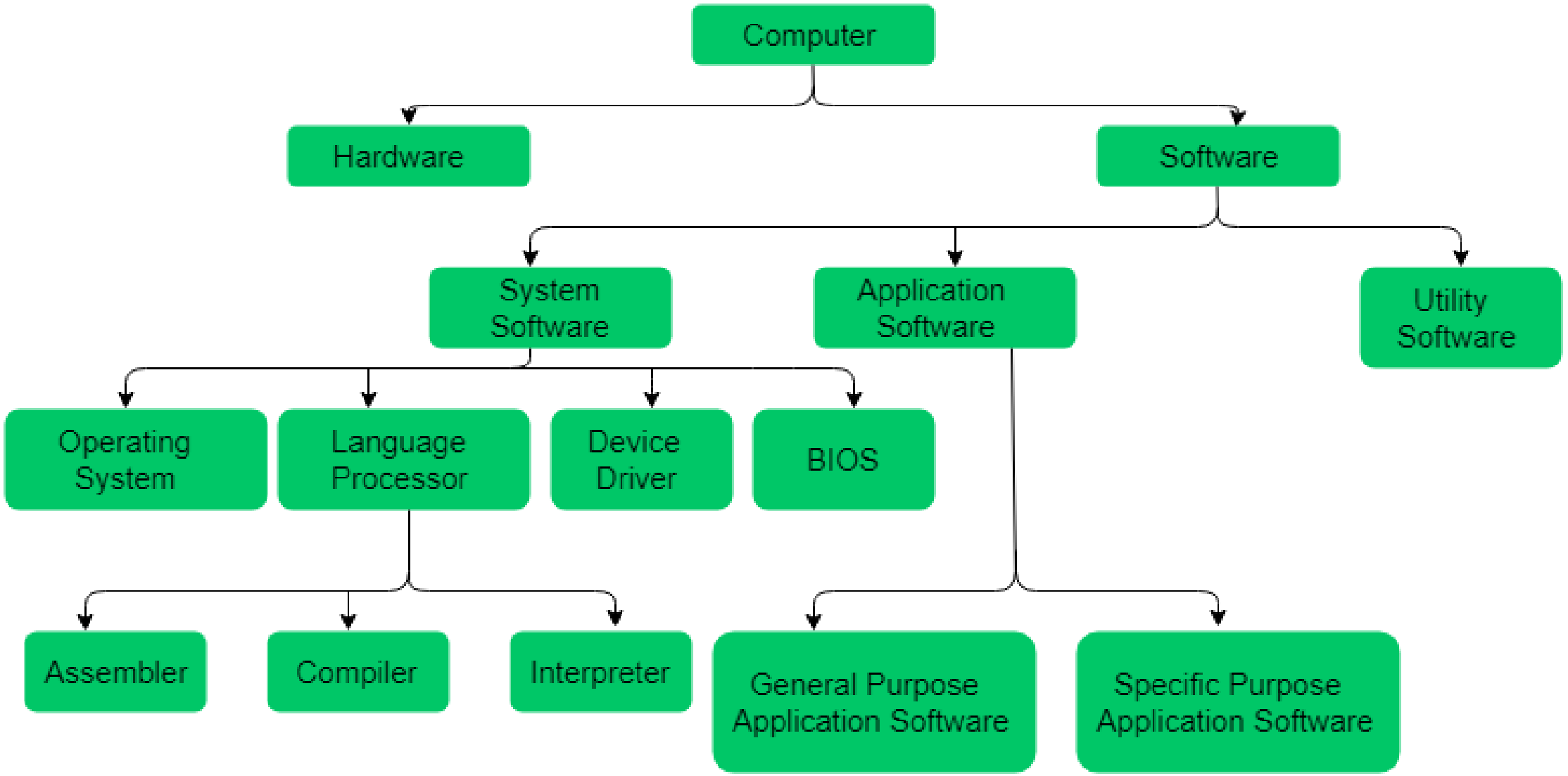



Mobile
Storage
Applications
CLOUD
COMPUTING
Server
Database
Private
cloud
Public
cloud
Hybrid
cloud

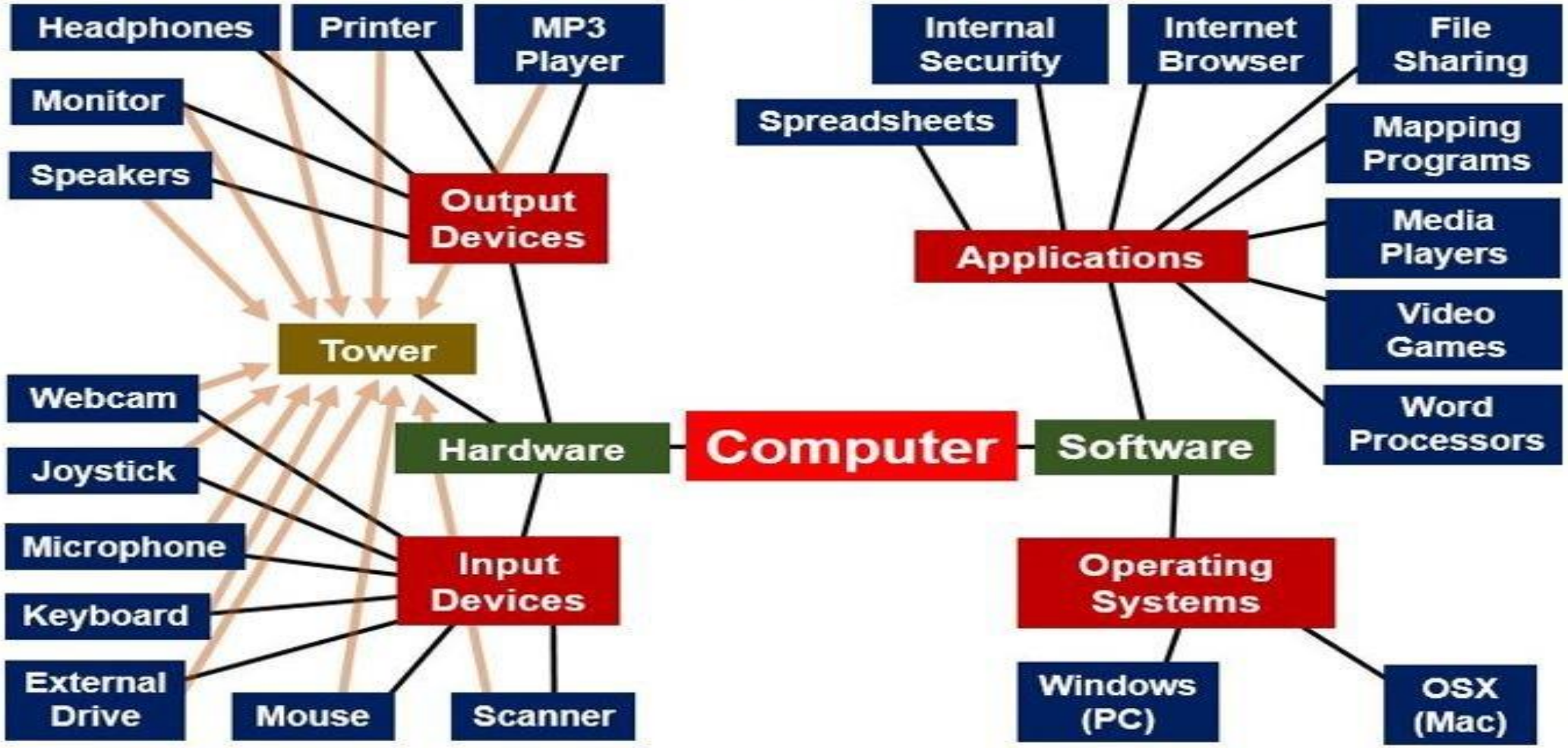
Hardware and Software

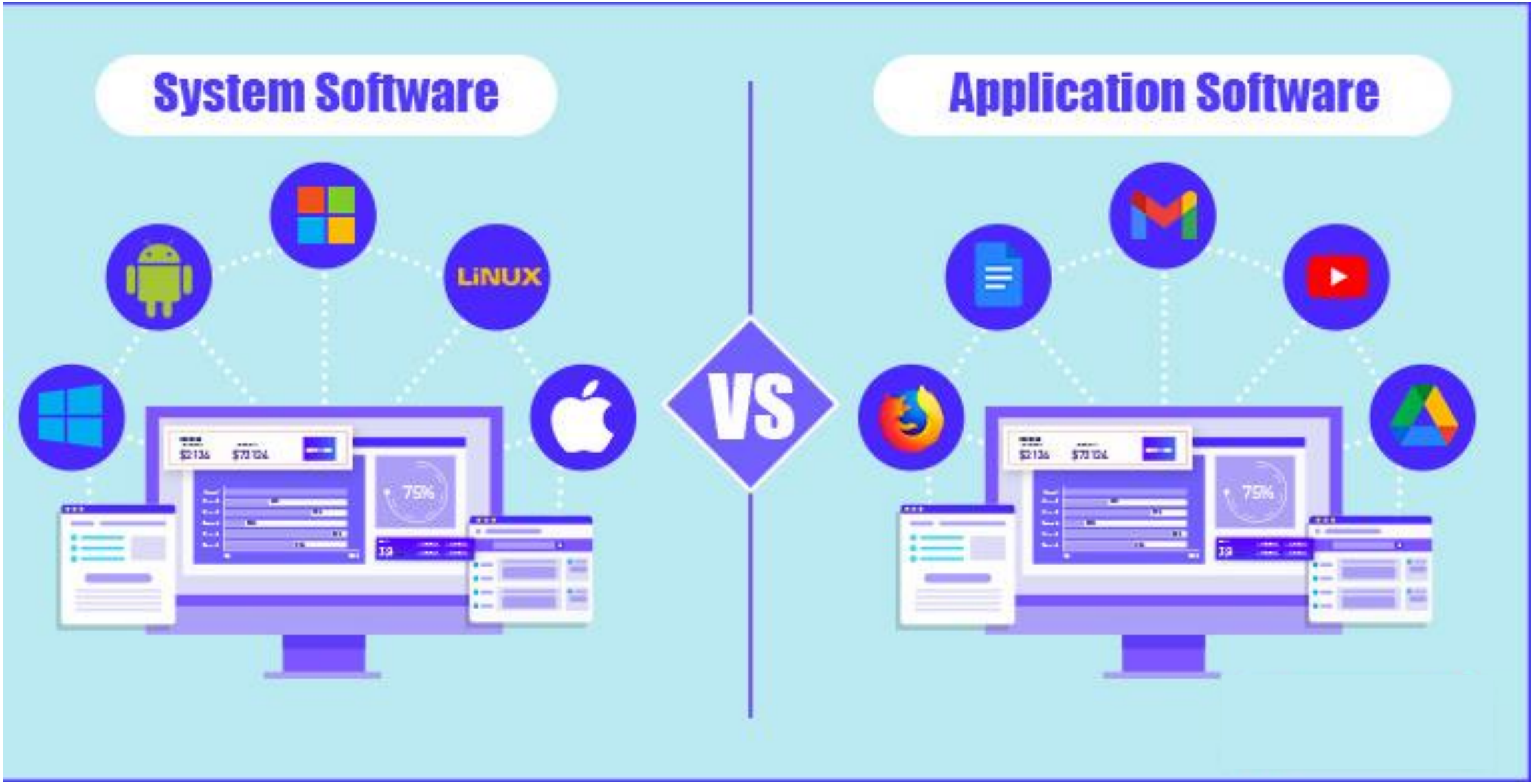
LECTURE 7


Application Software
Spreadsheets
Word processors
Databases
Computer Games
Internet Browsers
System Software
Operating System
Utilities
Hardware
CPU, disks, mouse, printer, etc.


Computer
Hardware
Software
System Software
Application Software
Utility Software
Operating System
Language Processor
Device Driver
BIOS
Assembler
Compiler
Interpreter
General Purpose Application Software
Specific Purpose Application Software

Hardware vs. Software

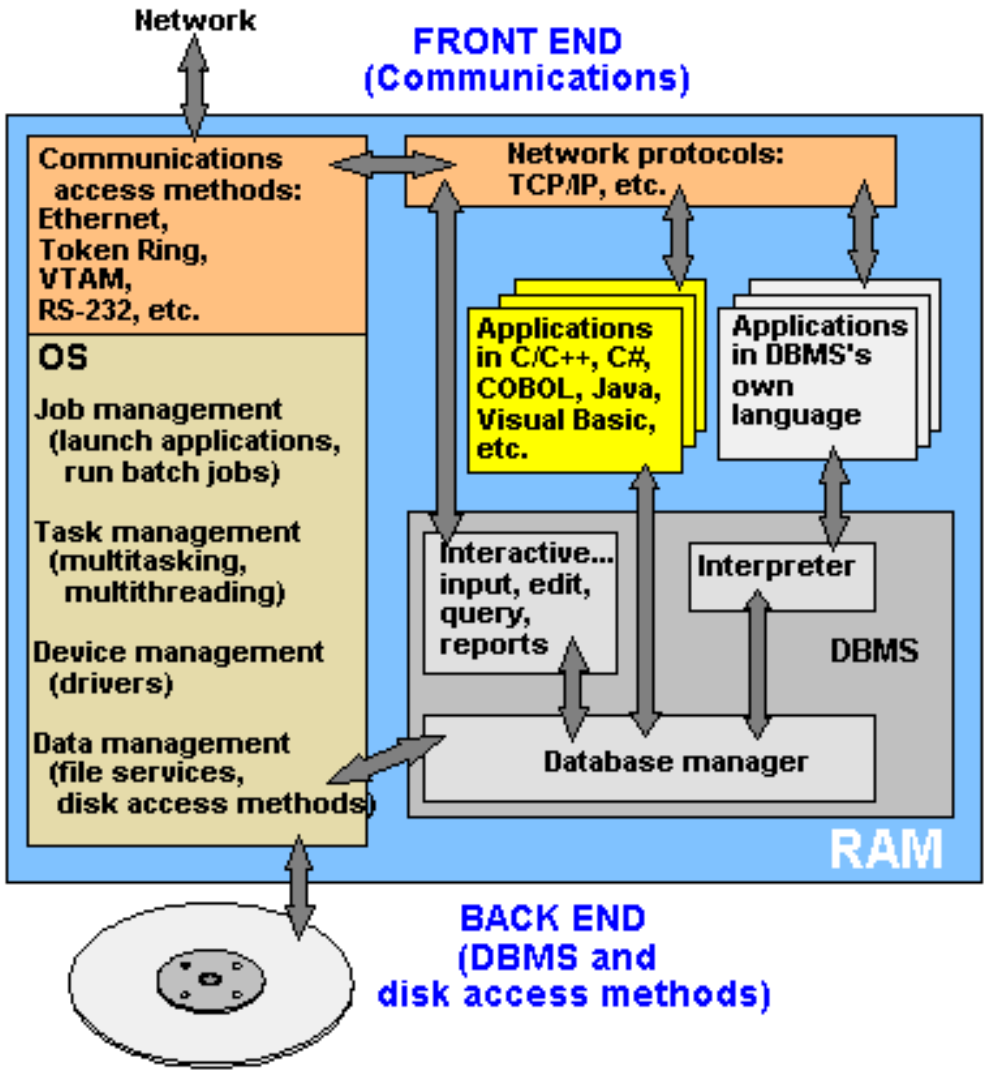



System Software
LINUX
VS
Application Software

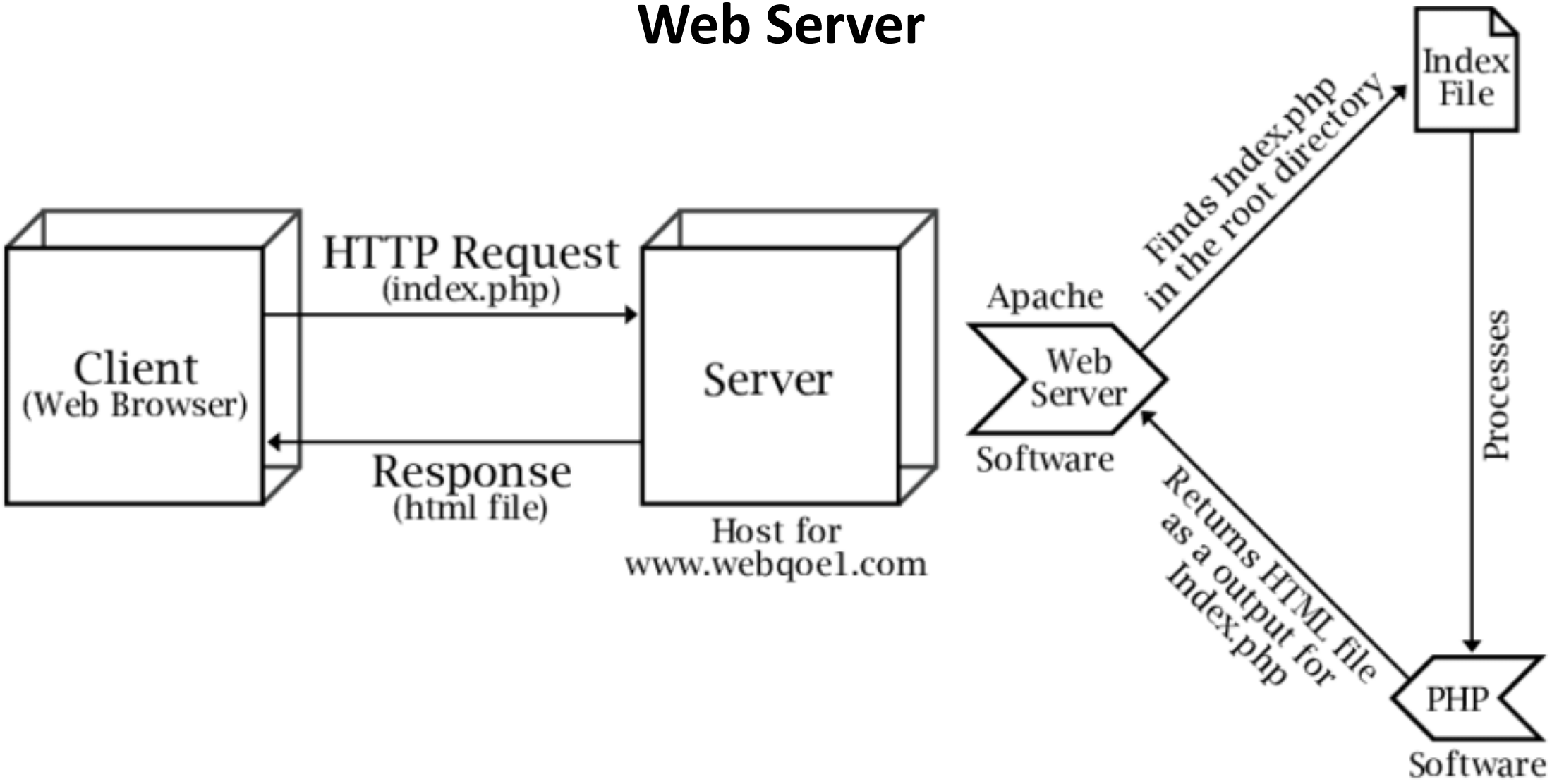
Operation System

- Memory Management
- Processor Management
- File Management
- Device Management
- I/O management
- Secondary Storage management
- Security
- Command Interpretation
- Networking
- Communication Management
- Job accounting

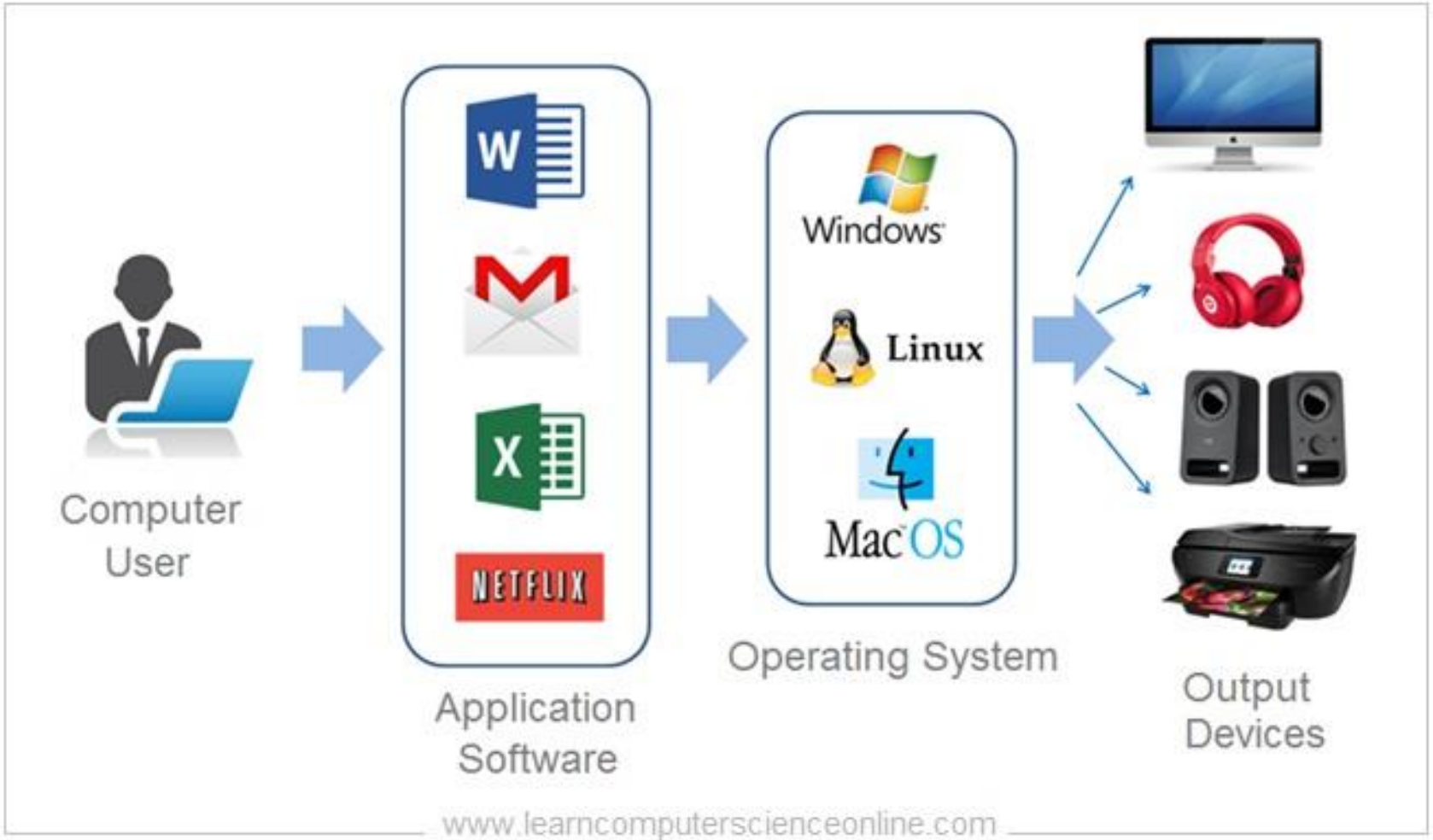
Software Architecture



Web Server


Client
(Web Browser)
HTTP Request
(index.php)
Response
(html file)
Server
Host for
www.webqoe1.com
Apache
Web
Server
Software
Finds Index.php
in the root directory
Index
File
Processes
Returns HTML file
as a output for
Index.php
PHP
Software

Computer System Architecture - Application Software



Application Software

Word Processors

Examples:

- MS Word
- OpenOffice Writer
- GoogleDocs
- Corel WordPerfect
- AbiWord

Database Software

Examples:

- MS Access
- MySQL
- dBase
- Clipper
- File Maker

Multimedia Software

Examples:

- VLC
- Picasa
- Adobe Photoshop
- Windows Media Player
- Inkscape

Graphics Software

Examples:

- GIMP
- MS Paint
- CorlDRAW
- Blender
- Autodesk Maya

Web Browsers

Examples:

- Google Chrome
- Mozilla Firefox
- Microsoft Edge
- Internet Explorer
- Safari

Other Types of Application Software include "Educational & Reference Software" and "Customized & Specific Purpose Software".

Types of Application Software

7-Layer Architecture

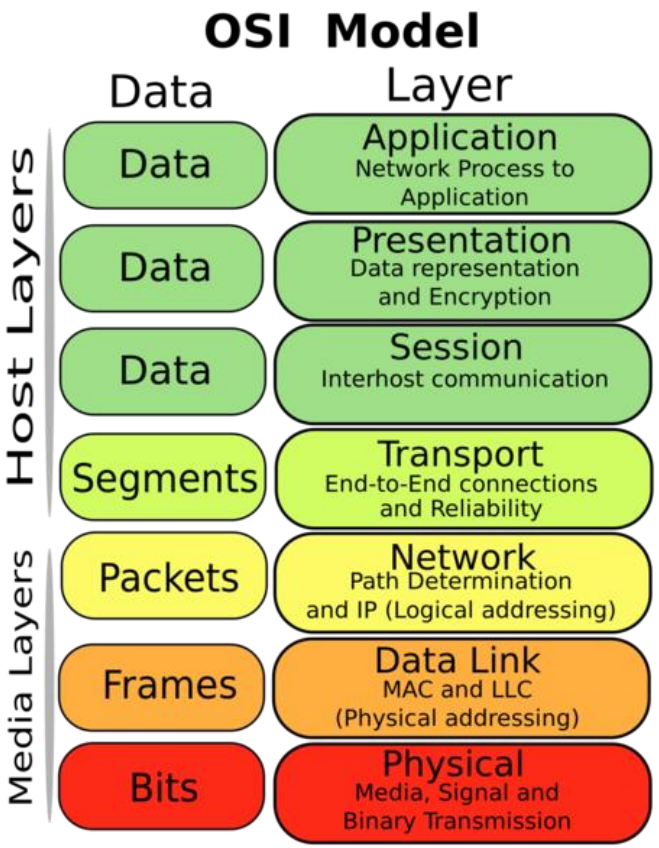
LECTURE 1

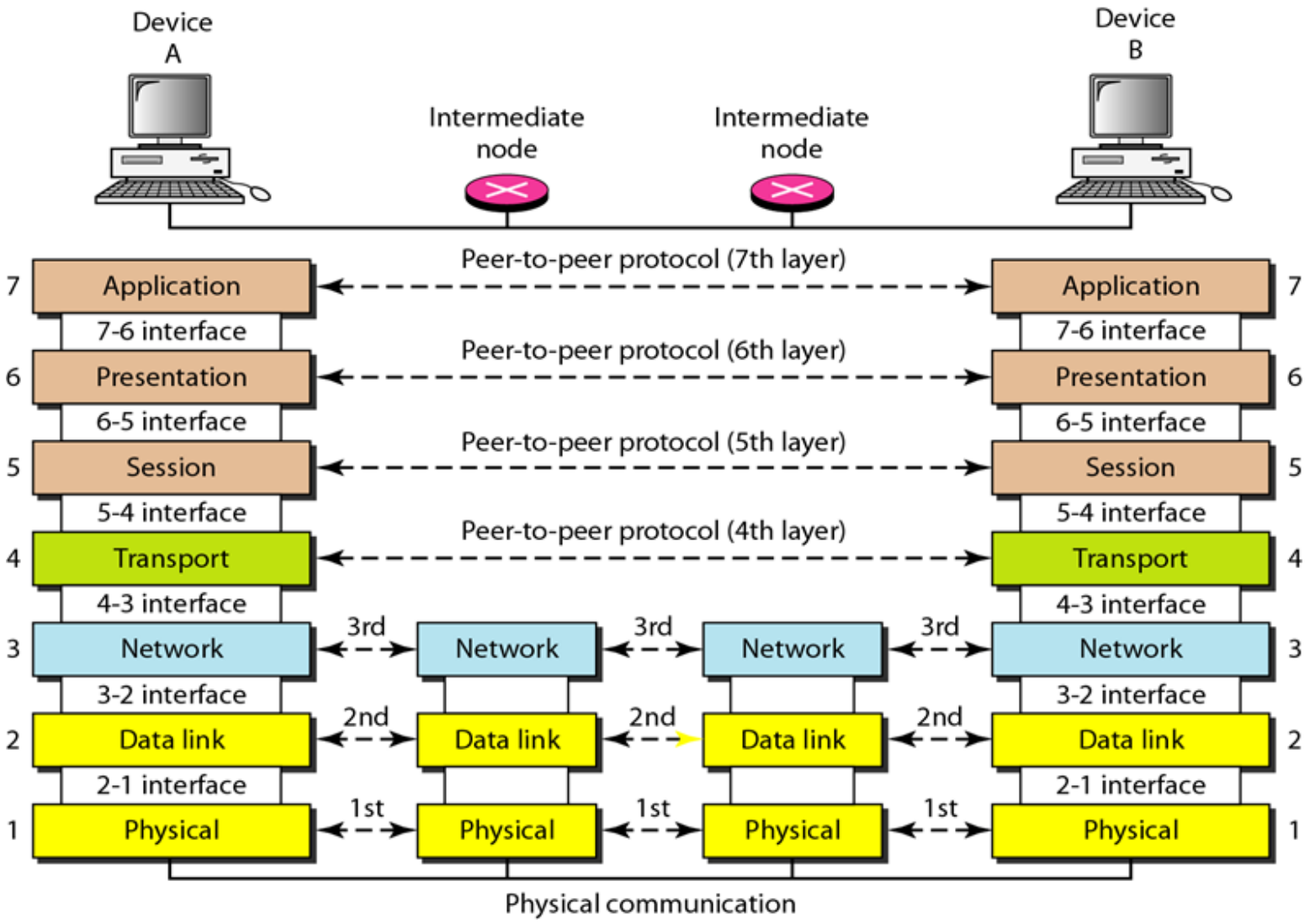
Vocabulary

- **HTML** - HyperText Markup Language: the language of the web, code for web pages
- **HTTP** - HyperText Transfer Protocol: designed for communication on the web, for sending and receiving information
- **ASCII** – American Standard Code for Information Interchange – raw text that any computer can recognize
- **Abstraction** - a simplified representation of something more complex. Abstractions allow you to hide details to help you manage complexity, focus on relevant concepts, and reason about problems at a higher level.
- **Protocol** - A set of rules governing the exchange or transmission of data between devices.

Protocol, Character Set, Data Frame

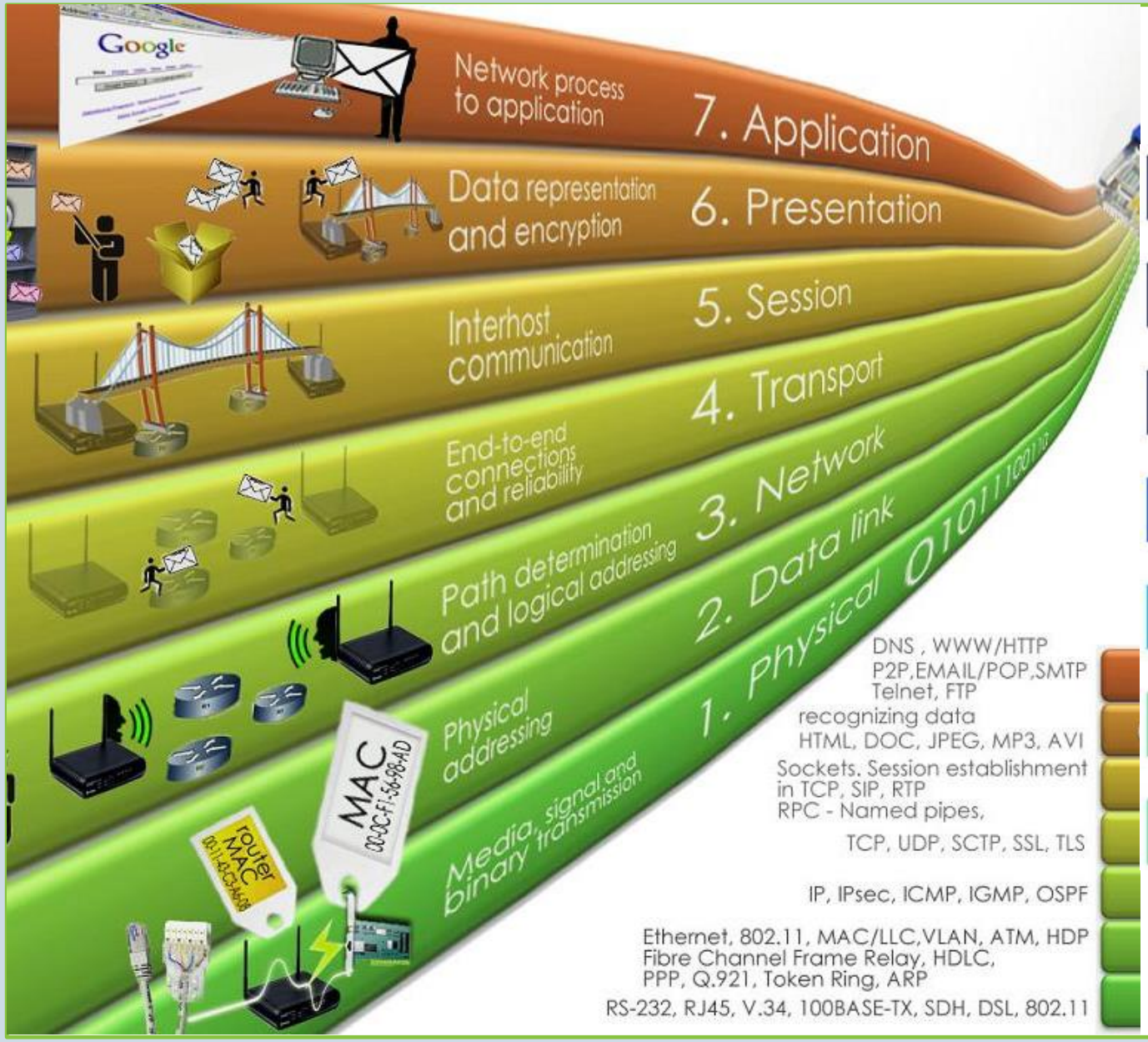
- One of the most powerful uses of the internet is sending text to people. Since the internet can only send bits around we need a way to encode text with bits...
- If it were up to you, how would you encode text in binary? Quickly, jot down an idea for encoding text.


OSI Model
Data
Layer
Host Layers
Media Layers
Data
Application
Network Process to Application
Data
Presentation
Data representation and Encryption
Data
Session
Interhost communication
Segments
Transport
End-to-End connections and Reliability
Packets
Network
Path Determination and IP (Logical addressing)
Frames
Data Link
MAC and LLC (Physical addressing)
Bits
Physical
Media, Signal and Binary Transmission

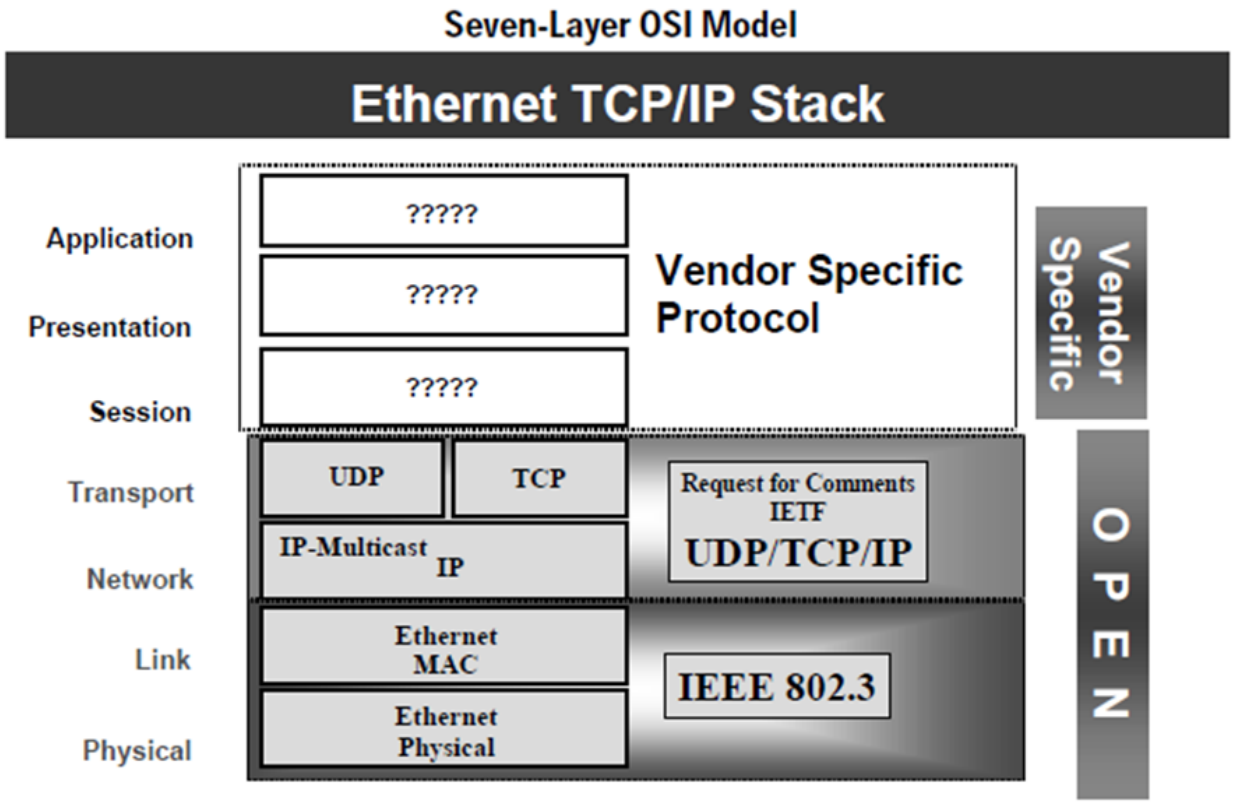

Device A
Intermediate node
Intermediate node
Device B
Peer-to-peer protocol (7th layer)
Peer-to-peer protocol (6th layer)
Peer-to-peer protocol (5th layer)
Peer-to-peer protocol (4th layer)
7 Application
7-6 interface
6 Presentation
6-5 interface
5 Session
5-4 interface
4 Transport
4-3 interface
3 Network
3-2 interface
2 Data link
2-1 interface
1 Physical
3rd
2nd
1st
Physical communication

7-Layer Architecture

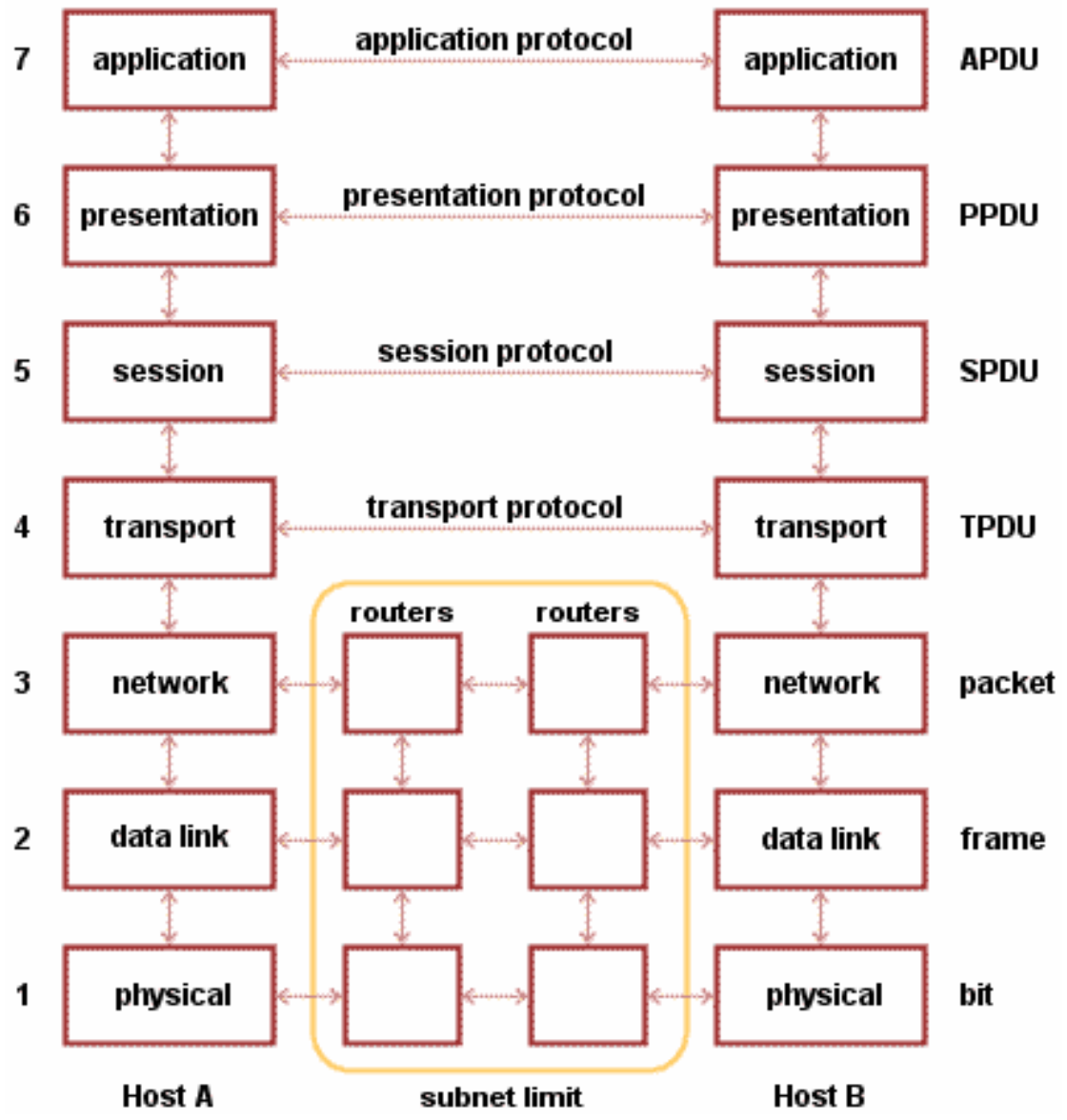
LECTURE 1



7	Application Layer	Human-computer interaction layer, where applications can access the network services
6	Presentation Layer	Ensures that data is in a usable format and is where data encryption occurs
5	Session Layer	Maintains connections and is responsible for controlling ports and sessions
4	Transport Layer	Transmits data using transmission protocols including TCP and UDP
3	Network Layer	Decides which physical path the data will take
2	Data Link Layer	Defines the format of data on the network
1	Physical Layer	Transmits raw bit stream over the physical medium

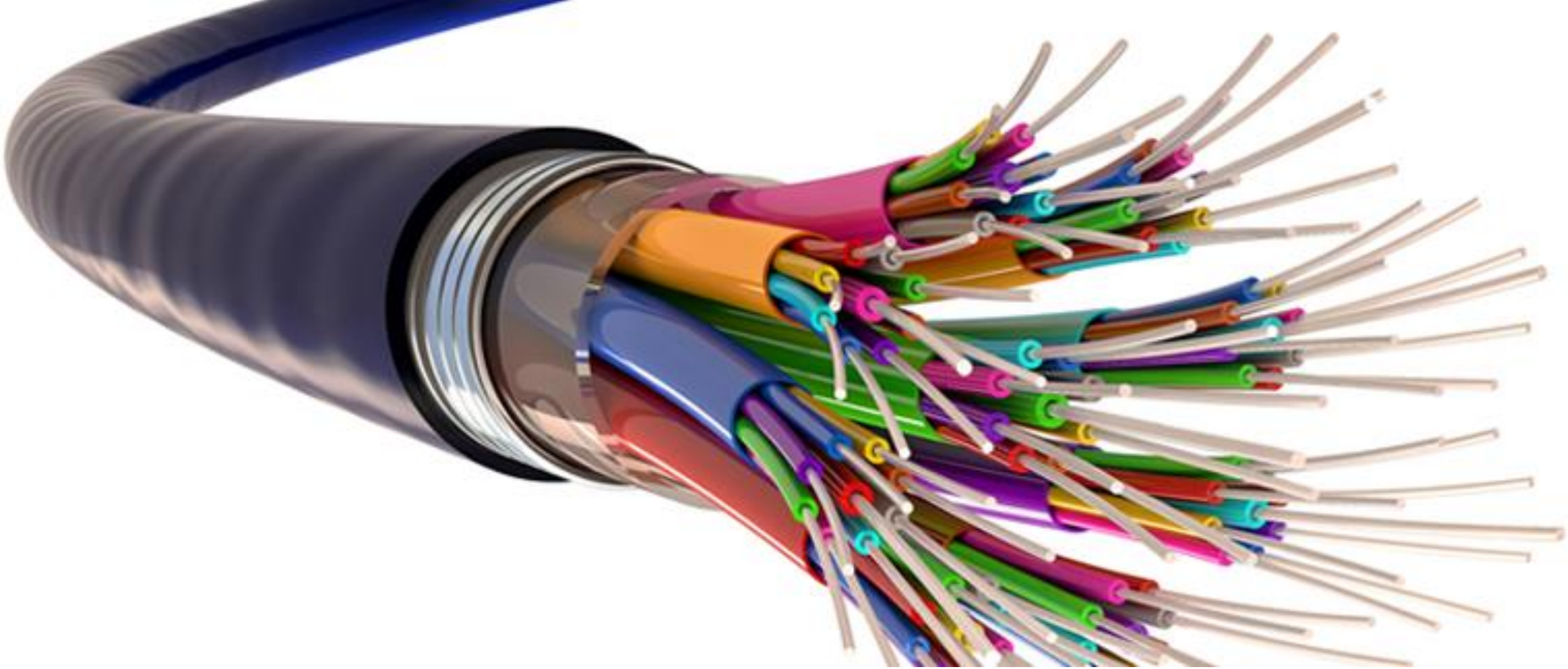


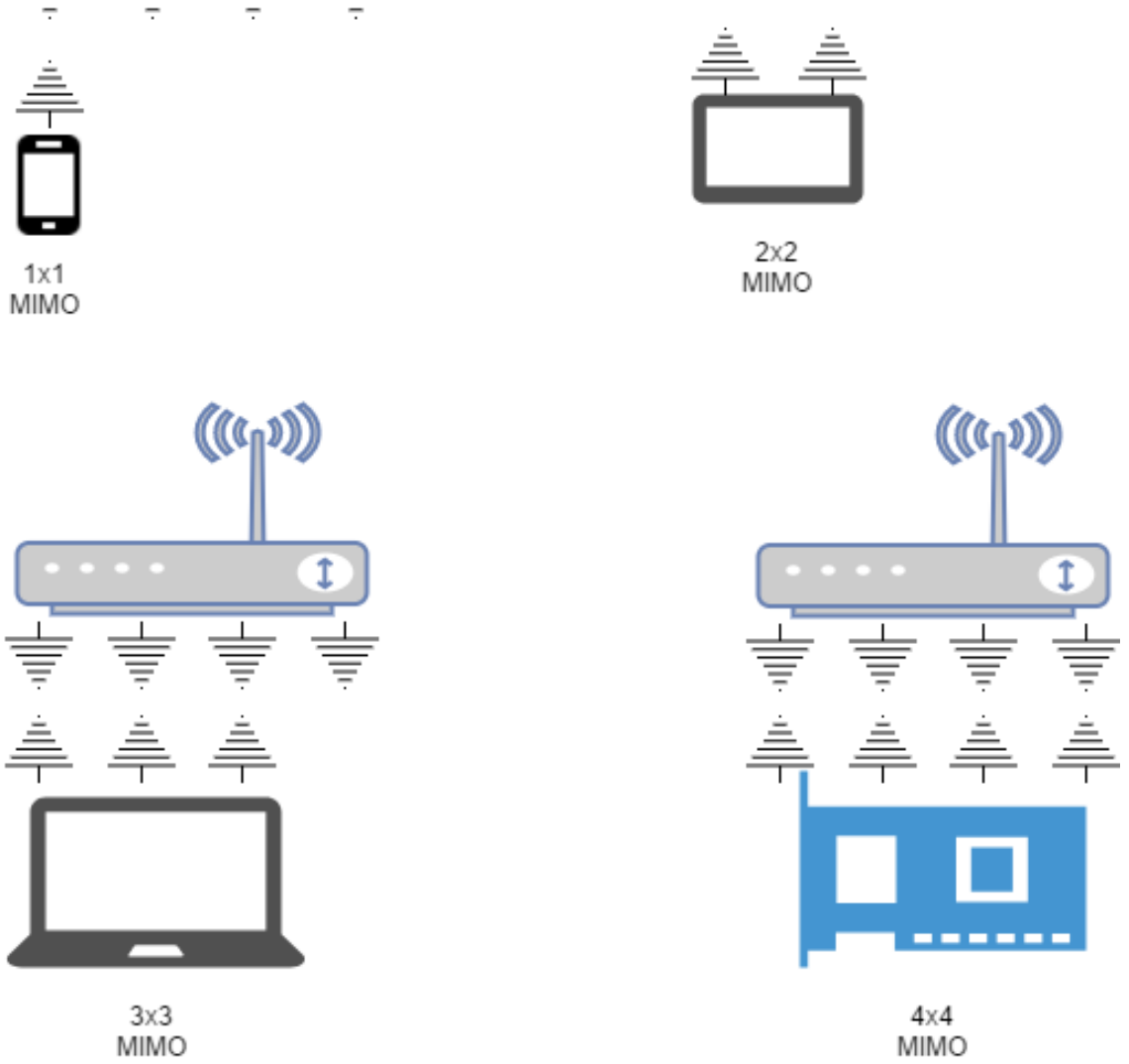
OSI Model LAN Protocols



Logical Internet Model

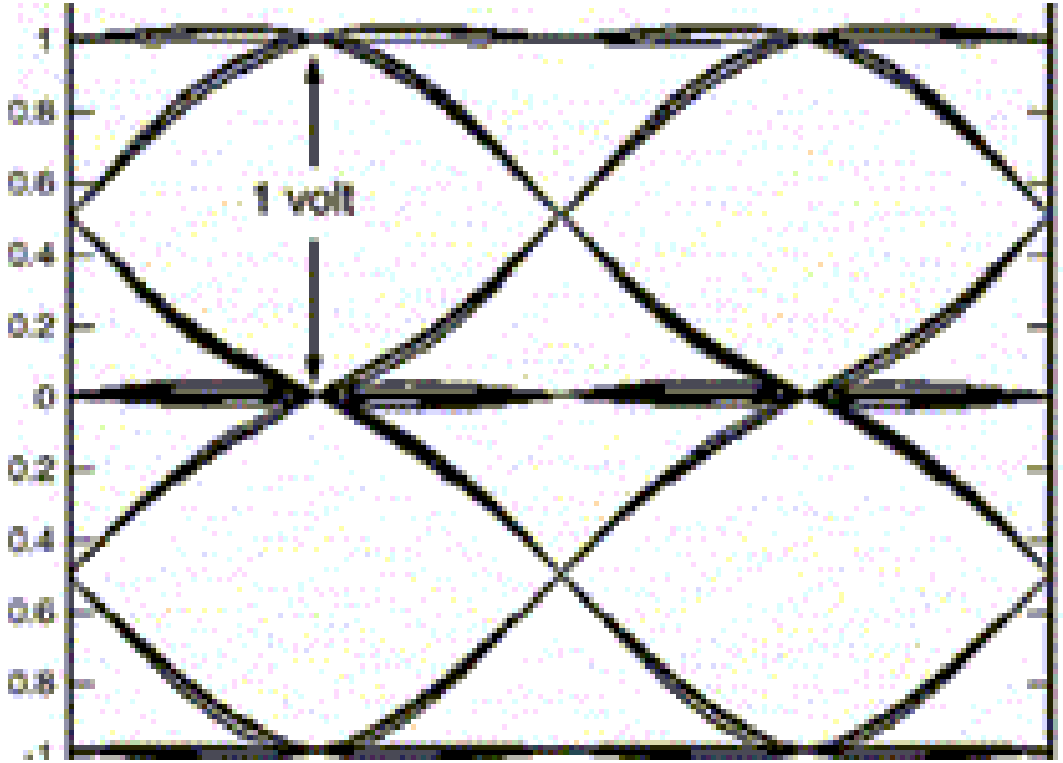
Internet
Wired(LAN)

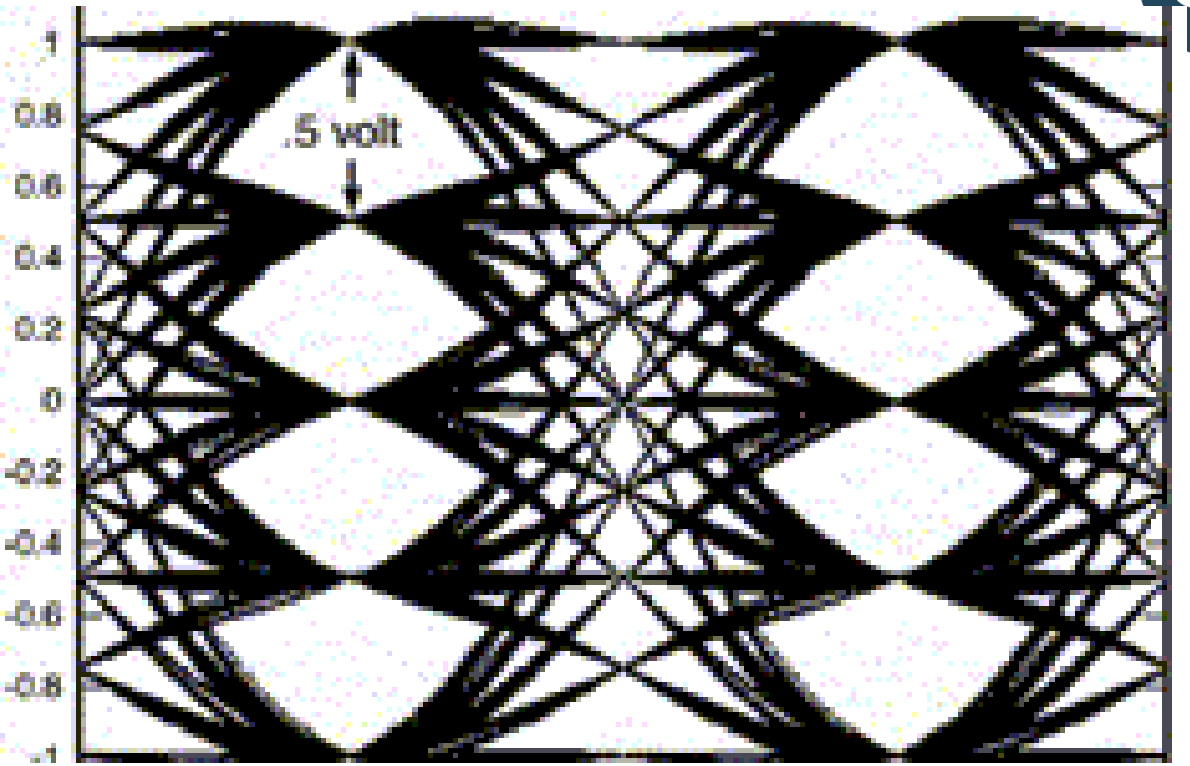




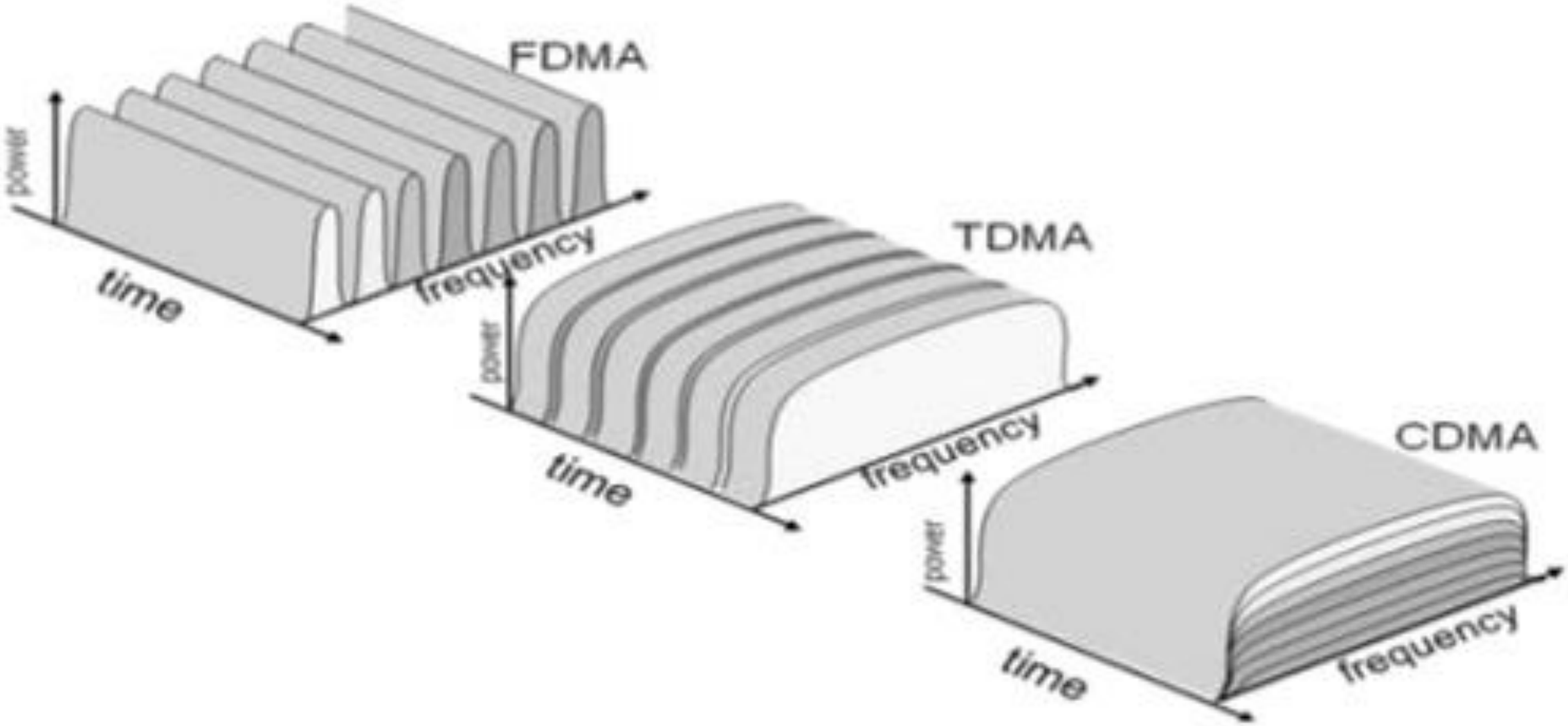
Internet WIFI

5G





100 BT and 1GeT

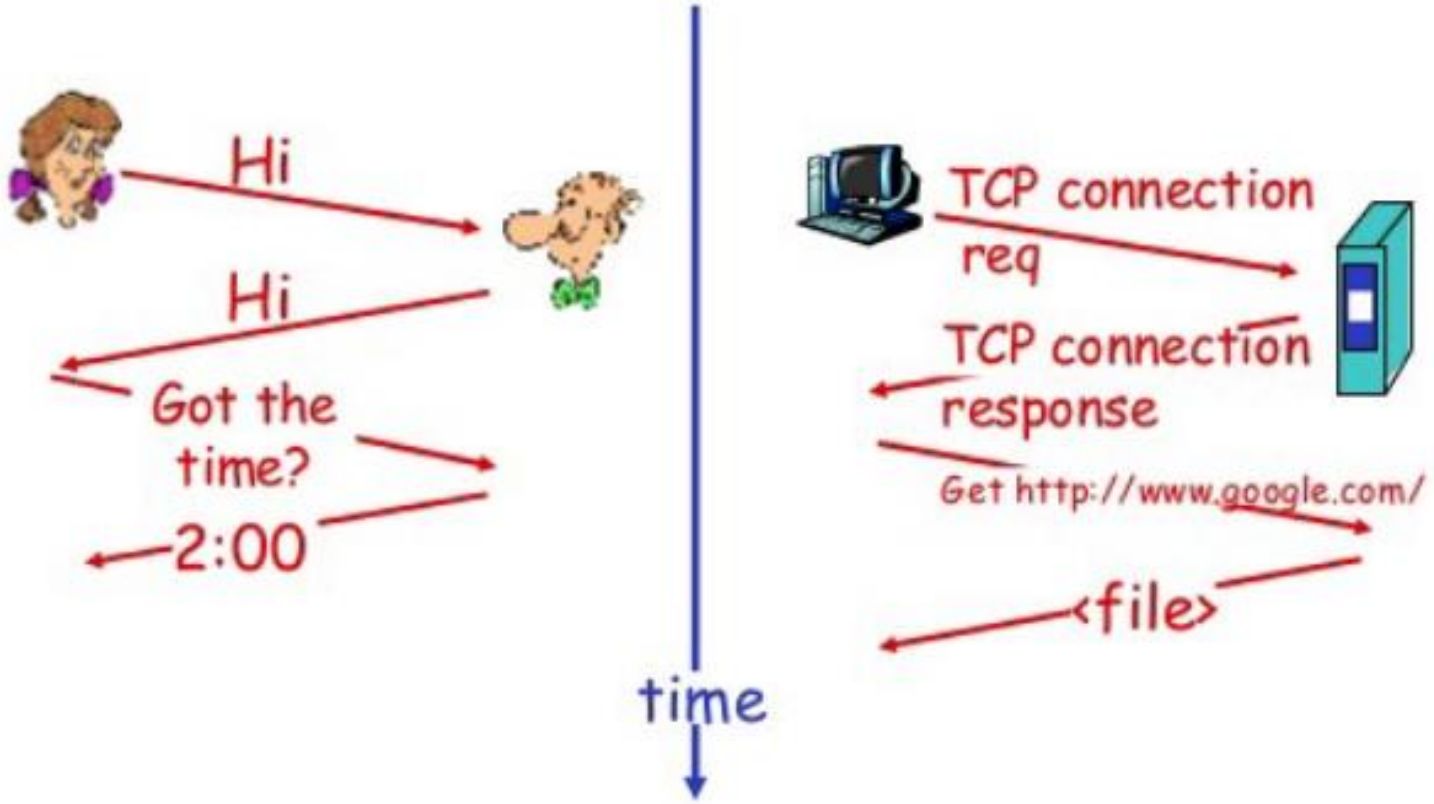

FDMA
power
time
frequency
TDMA
power
time
frequency
CDMA
power
time
frequency

Protocol

LECTURE 1

What's a protocol?

a human protocol and a computer network protocol:



Protocols

Protocols (Rules)

What to communicate, how to communicate and when to communicate.

Key Elements of Protocols

1. Syntax: Structure of format of data
2. Semantics: Meaning of each section of bits.
3. Timing: When to send and how fast?

HTML

- A lot of you have taken web design, and you used this language: HyperText Markup Language, or HTML
- This is the **language of the web**, where the content and formatting of a web page are written.
  Look at the background of a web page for the code.

Protocol called: HTTP

- HyperText Transfer Protocol
- this is the foundation for communication on the web – it was designed to send and receive web page data over the internet.

ASCII

- ASCII makes it all happen.
- ASCII – American Standard Code for Information Interchange
- This is the universally recognized raw text format that any computer can recognize

ASCII

- ASCII encoding is the standard number-to-text encoding scheme used in computers and on the Internet.
- This protocol is 7 bits long and so can encode 128 different symbols, each corresponding to one of the binary numbers between 0 and 127.
- Originally the encoding was designed for older printers and so the codes associated with the numbers 0-31 were designated as "control codes" which instruct the printer to perform certain actions.

ASCII

- ASCII Codes 32-126, however, contain many of the most commonly used symbols you use on the internet, including the lowercase and capital letters, the numbers 0-9, and most commonly-used punctuation marks.
- This printable character set is also used to represent all of the text formatting we use to add variety and emphasis to "plain-text" documents. The ability to represent formatting using only ASCII symbols is the result of cleverly designed protocols.

ASCII

- ASCII codes were originally 7 bits long and so there are 128 possible values.
- 0-31 are “control characters” that are largely defunct and go unused; they were formerly used to control various aspects of machines and printers.
- 127 is the symbol for delete.

ASCII

- Over time, 8 bits became a standard “chunk-size” for encoding information. ASCII made the transition to this 8-bit encoding by just adding an extra 0 to the front of the old 7-bit codes.

Application Layer Protocols

- File Transfer Protocol (FTP)
- Hypertext Transfer Protocol (HTTP)
- Simple Mail Transfer Protocol (SMTP)
- Post-Office Protocol (POP3)
- Domain Name System (DNS)

Application Layer Protocols

Uniform Resource Locator:

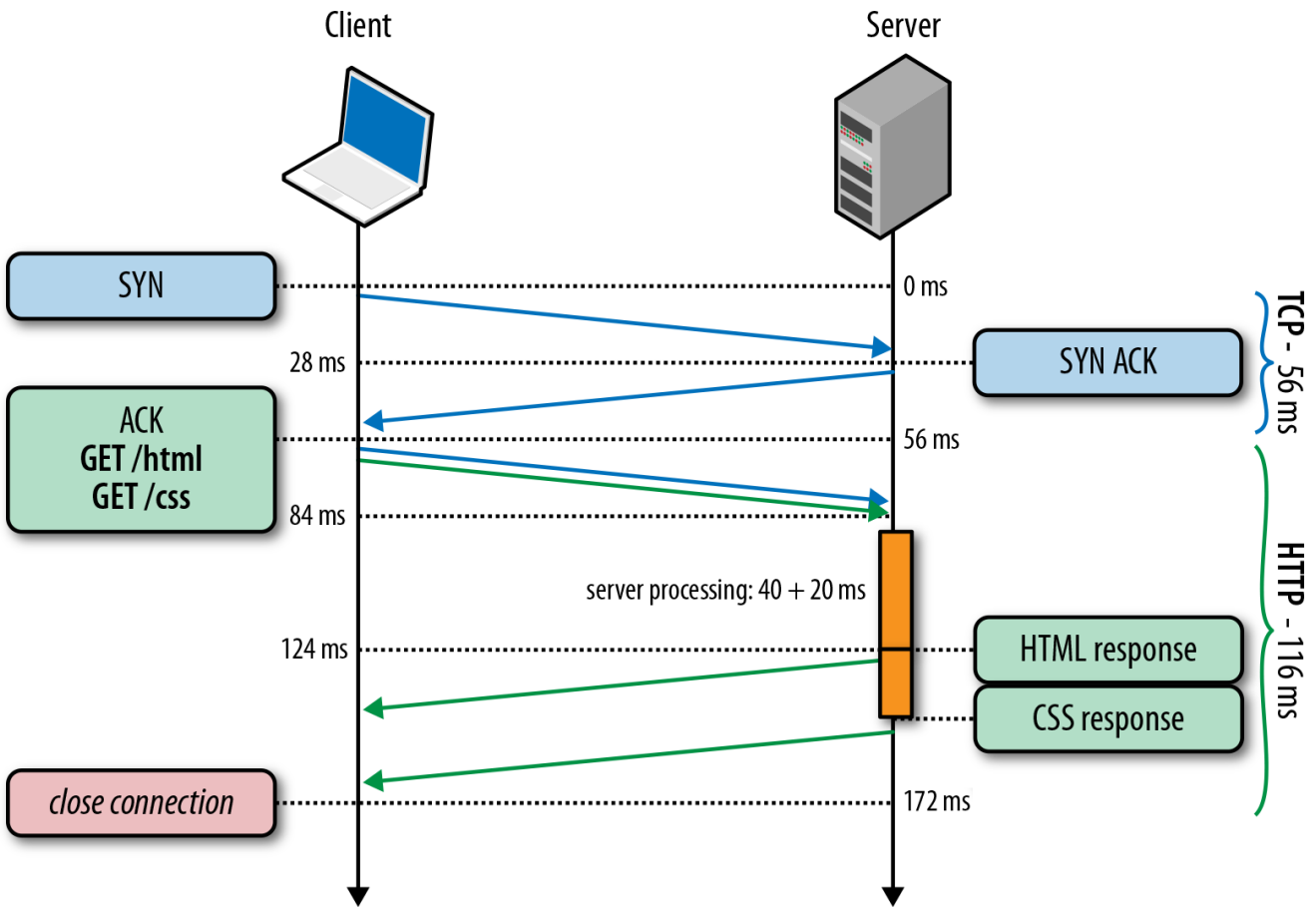
<scheme>:<hierarchical part>[? query][# fragment]

Important Schemes: ftp, http, mailto, file, app, dns, https, im, info, mms

URL Sample: type http in google
https://**www.google.com**/search?**q**=Application+Layer+Protocols&oq=Application+Layer+Protocols&aqs=chrome.0.69i59.6246j0j8&sourceid=chrome&es_sm=122&ie=UTF-8#q=http

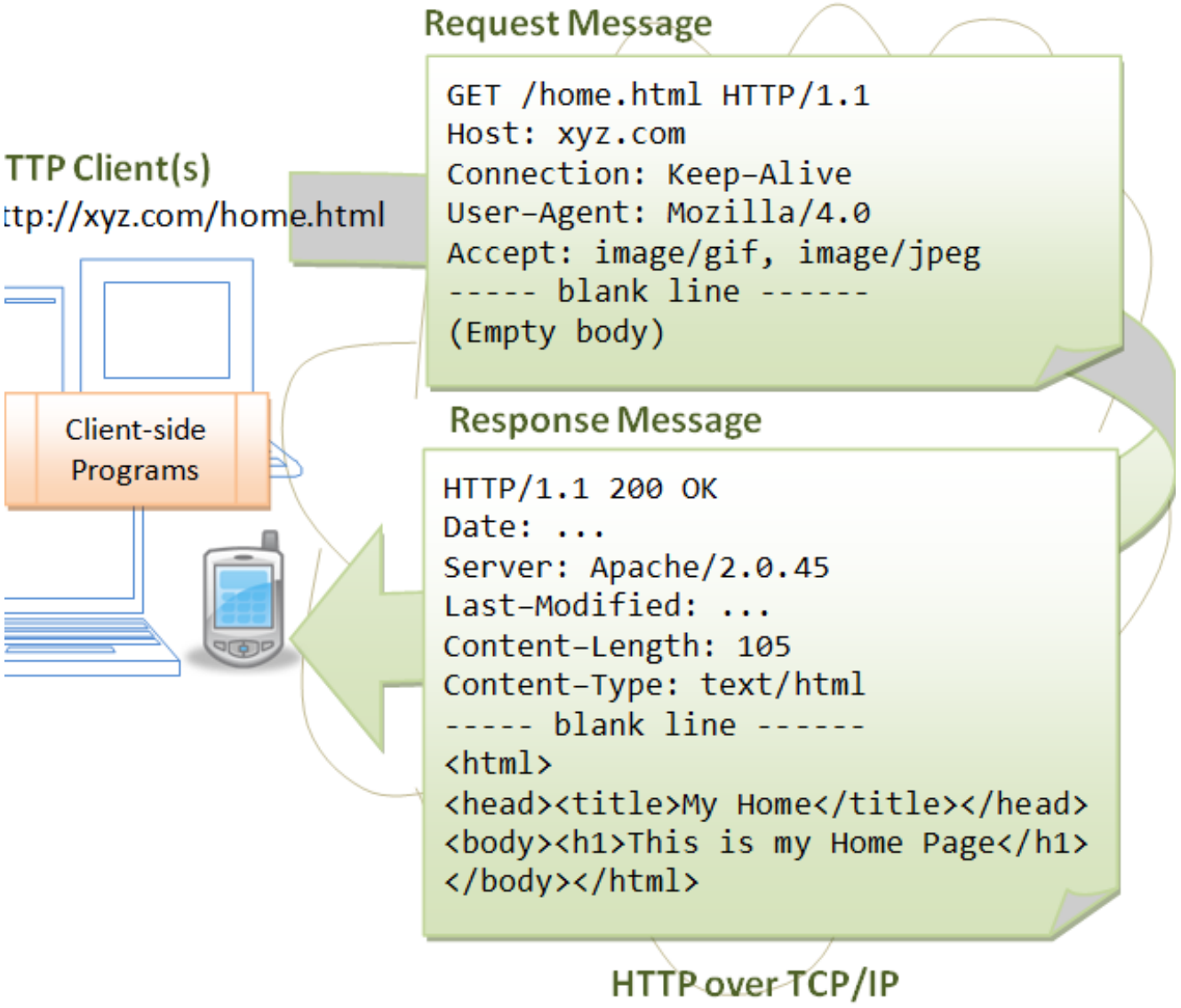
http://www.example.com/2012/samples/first.html

Scheme-Domain Name-----Directory--------Document



HTTP Connection Session Overview

HTTP Connection: Detailed Information



The Need for Addressing

LECTURE 1

Vocabulary

- **IP Address** - A number assigned to any item that is connected to the Internet.
- **Packets** - Small chunks of information that have been carefully formed from larger chunks of information.
- **Protocol** - A set of rules governing the exchange or transmission of data between devices.

Objective

- Explore more deeply how communication between multiple computers can work over the Internet.
- Time to play a simplified game of Battleship!

Game of Battleship

- The first game is played unplugged, in groups.
- The second game is played using the Internet Simulator, so that multiple students can connect to each other and see each other's messages.
- You must devise a messaging protocol that makes it clear who is sending the message and who the intended recipient is.

1 2 3 4 5 6 7 8 9 10
A
B
C
D
E
F
G
H
I
J

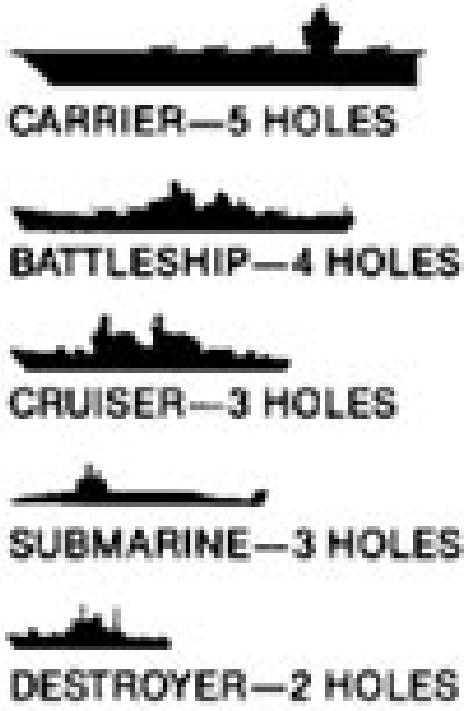
Rules: Game of Battleship

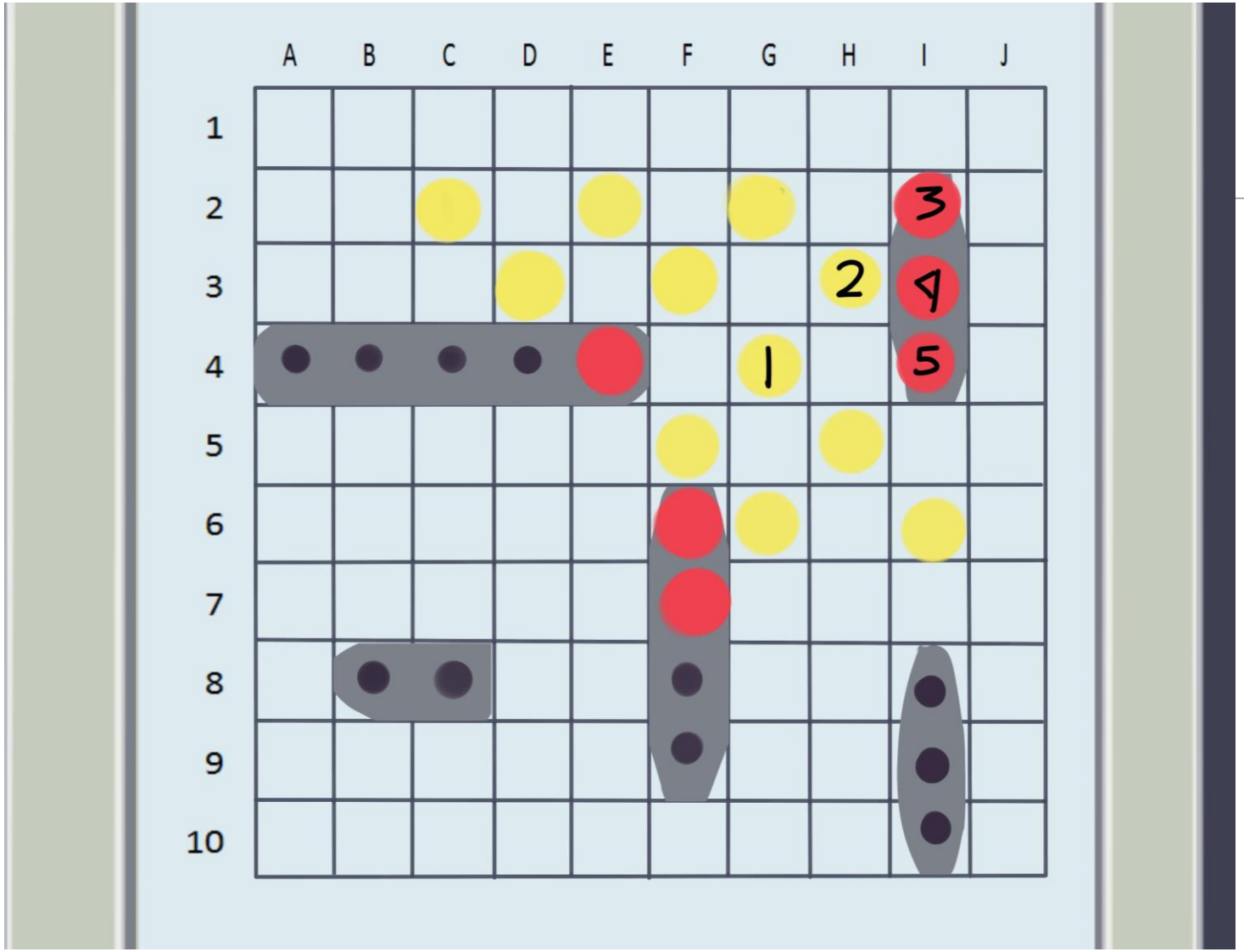
Components:

- 2 game units
- 10 plastic ships
- 4 runners of white pegs
- 2 runners of red pegs
- Label sheet

Each player's fleet contains 5 different ships:

- Carrier (5 holes)
- Battleship (4 holes)
- Cruiser (3 holes)
- Submarine (3 holes)
- Destroyer (2 holes)





Setup

You and your opponent sit facing each other and neighbor can see the other's ocean grid.

Place your fleet of 5 ships on the ocean grid.

Rules for placing ships:

- Place each ship in any horizontal or vertical position but not diagonally.
- Don't place a ship so that any part of it overlaps letters, numbers, the edge of the grid or another ship.
- Don't change the position of any ships once the game has begun.

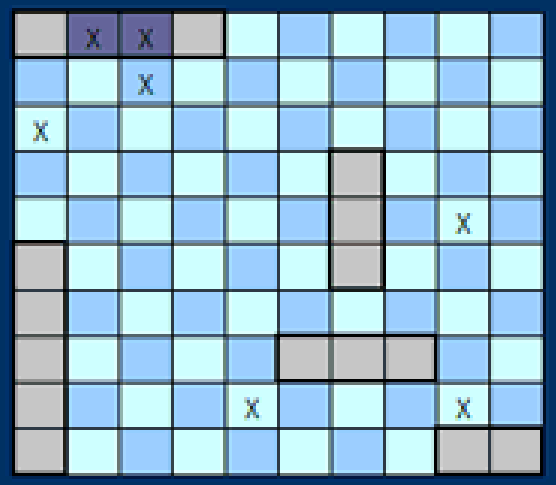
Game Play

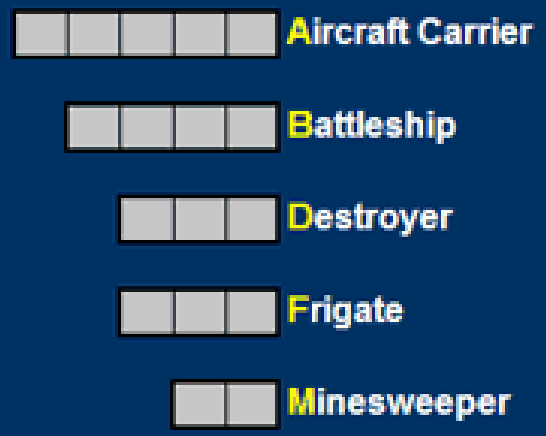
- Decide who will start.
- You and your opponent will alternate turns, calling out one shot per turn to try to hit each other's ships.
- On your turn, pick a target hole and call out its location by **letter** and **number**.

Battleships

Start New Game

Player 1 Board



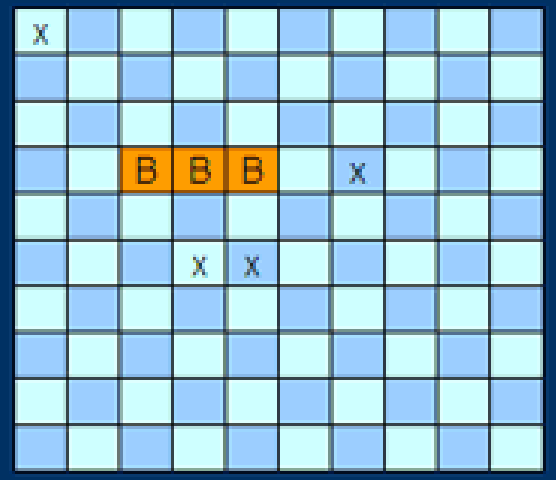


Search your opponent's board

Computer hit your Battleship!

You hit the computer's Battleship!

Computer's Board



How to play

eC Learning Channel

Game Play

When you call a shot, your opponent must tell you whether your shot is a hit or a miss.

It's a hit!

1. If you call out a shot location that is occupied by a ship on your opponent's ocean grid, your shot is a hit!
2. Your opponent tells you which ship you have hit (cruiser, submarine, battleship, etc...).
3. Record your hit by placing a red peg on your target grid. Your opponent places a red peg in the corresponding hole of the ship on his ocean grid.

Game Play

It's a miss!

1. If you call out a shot location not occupied by a ship on your opponent's ocean grid, its a miss.
2. Record your miss by placing a white peg in the corresponding target hole on your target grid, so you will remember this location and won't call this shot again.

End of the Game

Sinking a Ship

- Once all the holes in any one ship are filled with red pegs, the ship will sink. The owner of that ship must announce which ship was sunk.

End of the Game

- If you are the first player to sink your opponent's entire fleet of 5 ships, you win the game.

Demonstration:

AGAME.COM

- Understand the rule.
- Craft the communication protocol.
- Try to develop a protocol for human communication system. And, internet simulator version

What is needed in the game?

- You devise a **binary protocol** for playing this game which will entail developing an **addressing system** for players, as a well as a formal packet structure for transmitting data about the state of the game.

IP Addresses

- “Internet Protocol” addressing system, commonly known as **IP Addresses**.
- Many computers are connected together by networks. So if a computer sends some bits out, those bits pass through many computers. How does a computer know who those bits are for?
- If a response is necessary, how does it send it back? This is a complicated question, but the first part of the answer is that you need an addressing system, and some kind of fixed structure for messages that everyone agrees on (a **protocol**) so messages can be interpreted properly. (We'll get to routing later).

Video

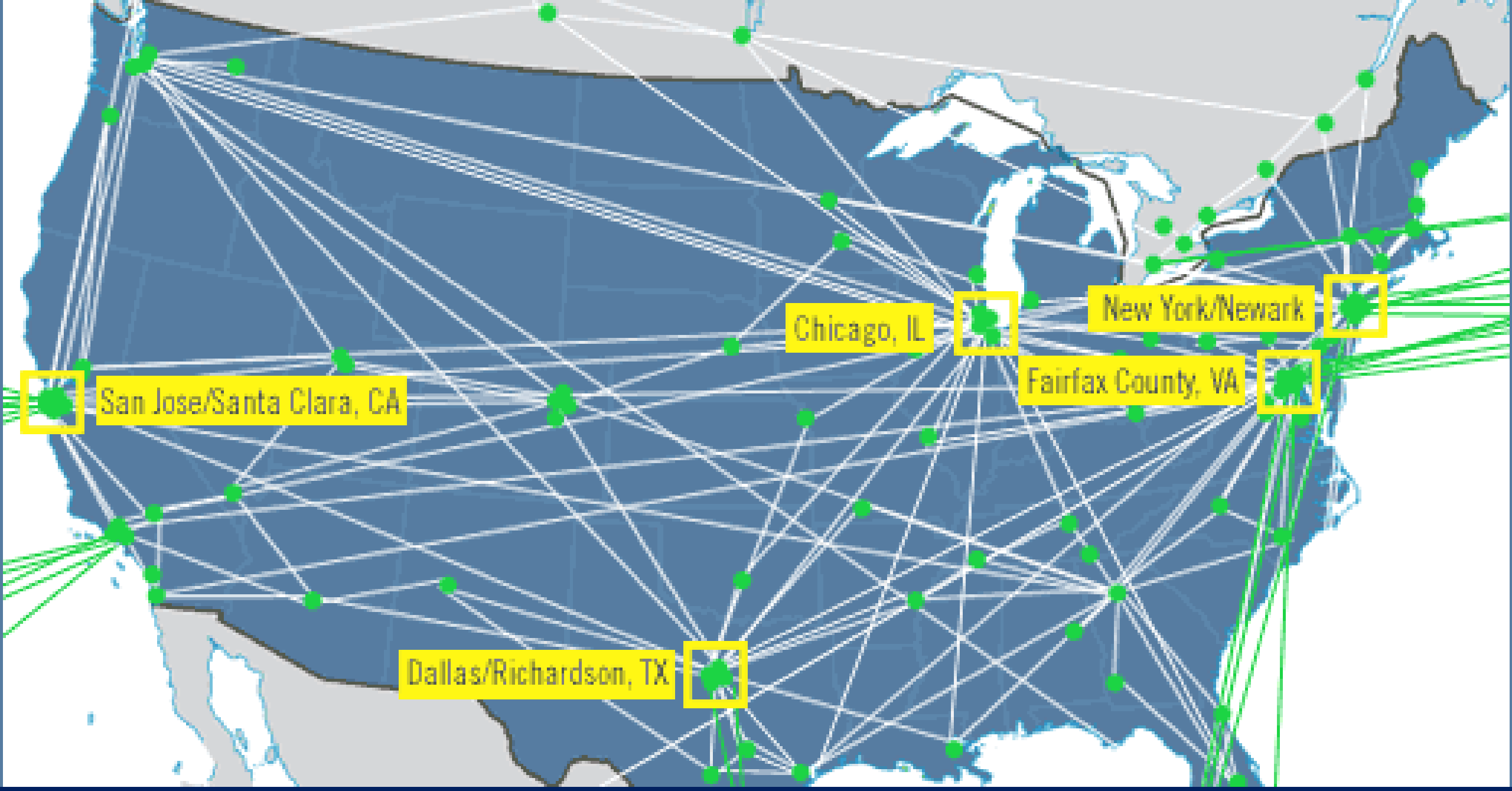
IP Address and DNS

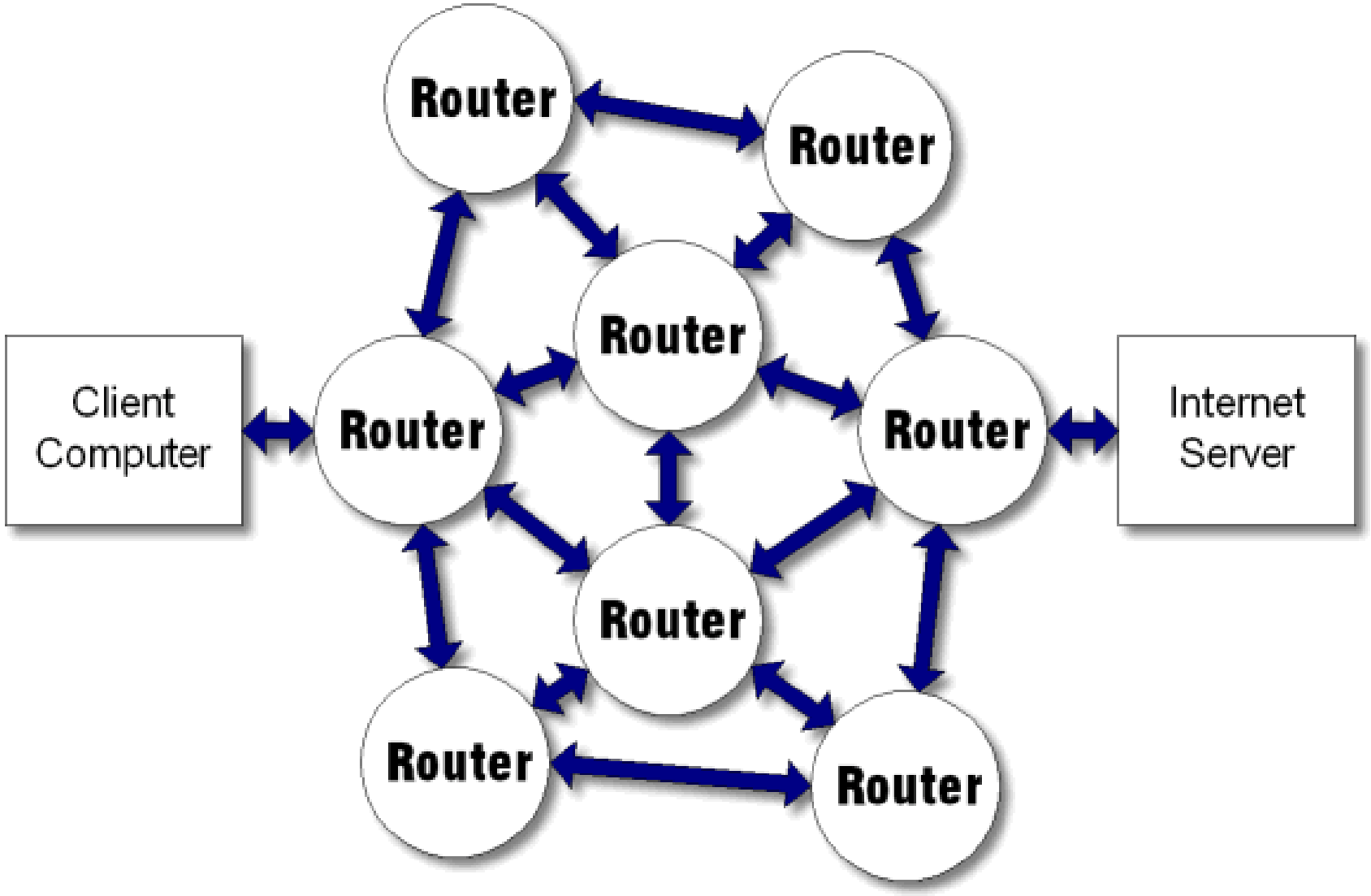
Fault Tolerant

LECTURE 1

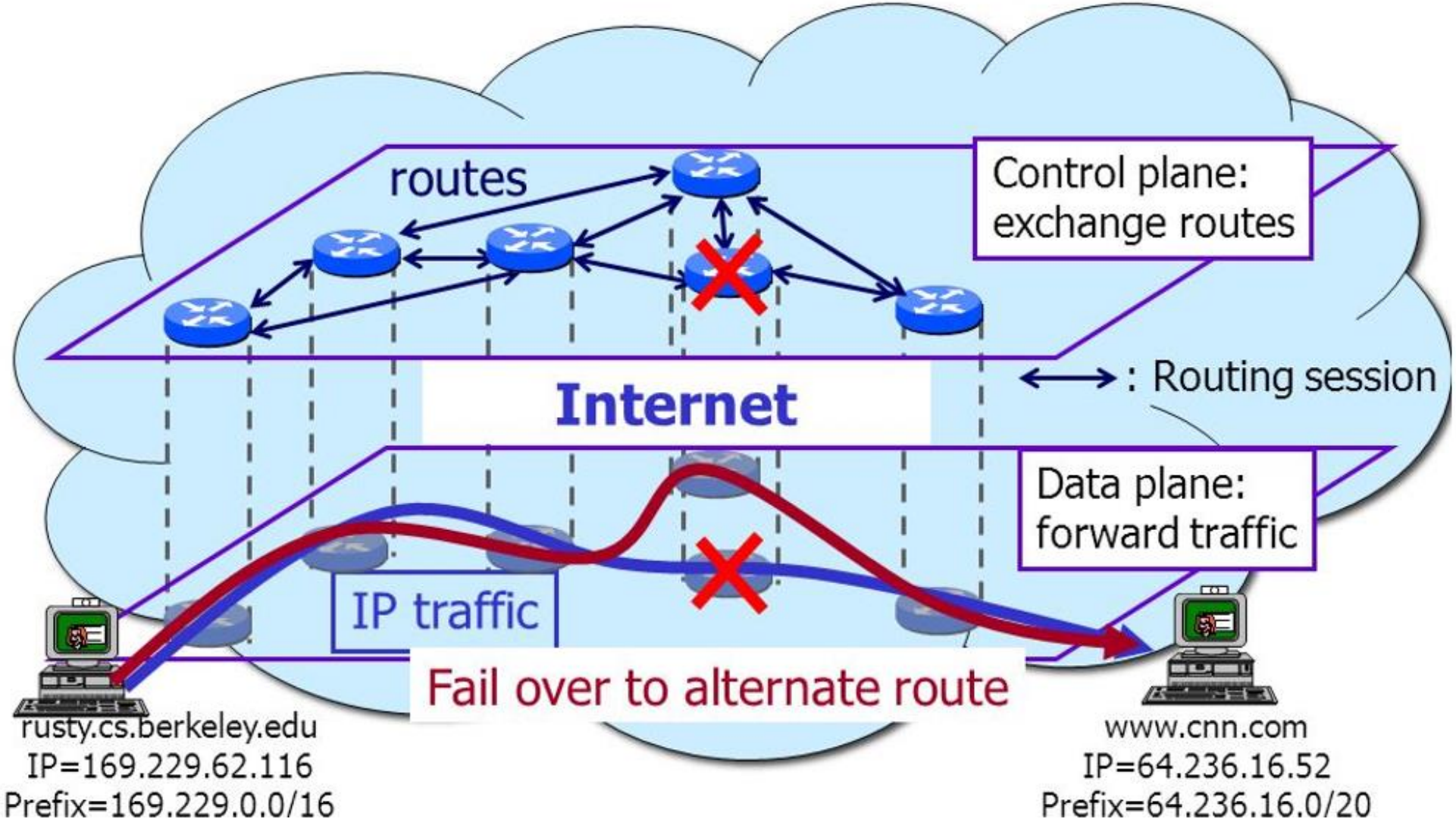
Vocabulary

- **Network Redundancy** - having multiple backups to ensure reliability during cases of high usage or failure
- **Router** - A type of computer that forwards data across a network


Chicago, IL
New York/Newark
Fairfax County, VA
San Jose/Santa Clara, CA
Dallas/Richardson, TX


Router
Router
Router
Client Computer
Router
Router
Internet Server
Router
Router
Router

Internet Routing



Objectives

- We are exploring the benefits (and potential security concerns) associated with routing traffic across the Internet.
- We will use a version of the Internet Simulator that allows messages to be sent only to an intended recipient, as indicated by the IP address.

Traffic in the Network

- The Internet Simulator also allows students to examine the traffic that goes through all of the (simulated) routers on the network.
- You will discover that messages go through many different routers, may not always take the same path to reach the final destination, and that the routers (and their owners) can see all of this traffic!

Background Knowledge

- CS Background: A **router** is a computer designed to receive and redirect packets of information based upon the addressing information (e.g. an IP address) contained in the packet.
- Routers will either deliver a packet to its final destination or forward it to one of several other routers it is connected to.

Background Knowledge

- By monitoring current network conditions, a **router** can determine which of these will allow the packet to reach its destination fastest.
- There will often be redundant paths between two locations on the Internet, and so if one path is experiencing traffic or otherwise out of service, additional paths will be available.
- This **redundancy** makes the Internet more reliable and also helps the Internet to scale, accommodating new users (and routers!) as they are connected to the system.

Scenario:

- Imagine you were going to send a letter to a friend living in another state. List the steps you imagine your letter would have to take through the different parts of the postal system.
- Don’t worry if you’re not sure about your answers, just make an educated guess.

Scenario:

- Did you come up with something like this:
- **Example:** Mailbox → Mail Carrier → Local Post Office → Regional Collection Center (of origin) → Regional Collection Center (of destination) → Local Post Office → Mail Carrier → Grandparent's House

Scenario:

- When we send messages through a network we don’t actually want everyone on the network to receive them.
- If we include information about who the message is intended for then we can allow portions of the network to focus on sorting and routing messages, so that they can continue on their way to their intended target.

Scenario:

- In the mail system, mail facilities, post offices, or a mail carrier fills this role.
  In a network of computers, certain computers called "routers" do the same thing, directing messages towards the target computer based on the IP addresses included in the message.

Activity

Internet Transmission with Router

- Today's activity introduces the **newest incarnation of the Internet Simulator**
- Messages now have “To” and “From” addresses in a format similar to IP addresses (just an 8-bit address instead of a full 32 bits)
- This allows messages to be sent to a single intended recipient.

Activity

Internet Transmission with Router

- The Internet Simulator also simulates the routing of messages across a network, with messages possibly being routed across multiple routers in unpredictable sequences before finally being delivered.
- This is done to simulate the way traffic travelling across the Internet is constantly rebalanced in response to over- or under-usage of some channels.
- A message will usually make it to its destination, but we can't know for sure how it will get there.

Video

https://www.youtube.com/watch?v=ewrBalT_eBM

How does Internet works?

Find the New Internet Simulator

Choose a Router:

- Add a router if you need more space.
- Then join a router with a few of the people sitting closest to you.
- Ideally, you'll have 3-4 classmates with you on your router.

Discussion

- some messages may be dropped
- messages may take different paths to the same location
- routers can theoretically read all traffic moving across them.

Discussion

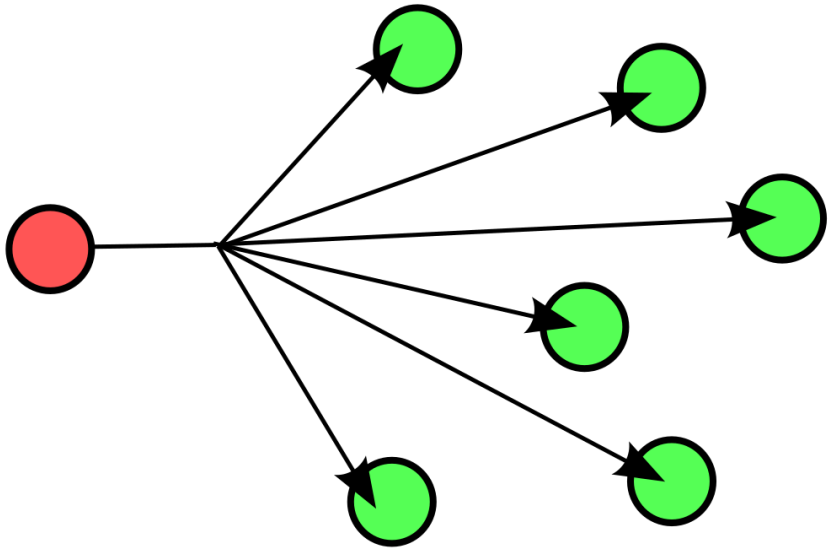
Read the Traffic:

- Once you have conducted your conversations, pause and demonstrate the ability to view router traffic in the Router tab.
- You should use the data generated here to answer the first set of questions on the Activity Guide.
- Click the “Log Browser” button to view the table of router data.

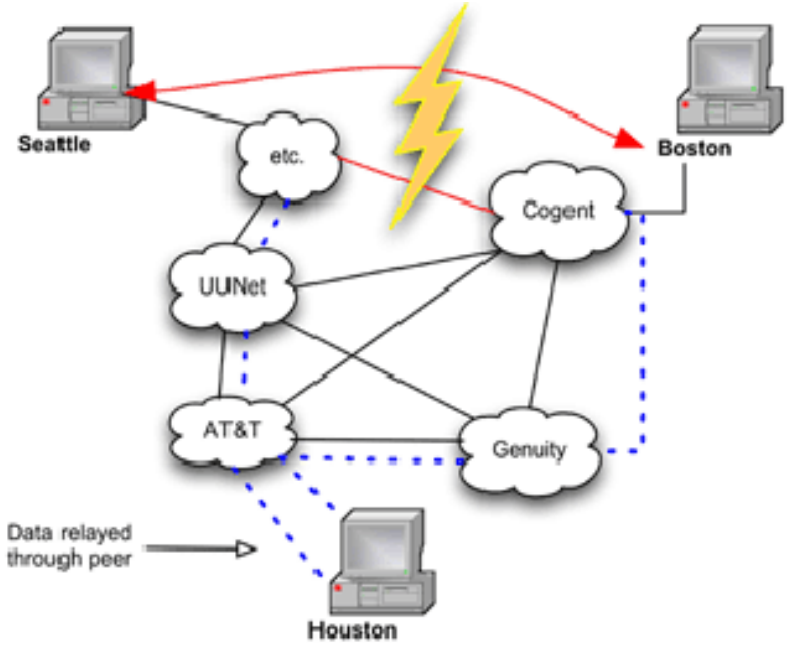
Discussion

Routing messages functions very differently from **broadcasting**.

- The messages are only sent to the user they are intended for.
- This is just like when you write a letter to a friend, where an address on the outside of the letter tells the mail service how to properly route your letter.



Broadcasting



Routing

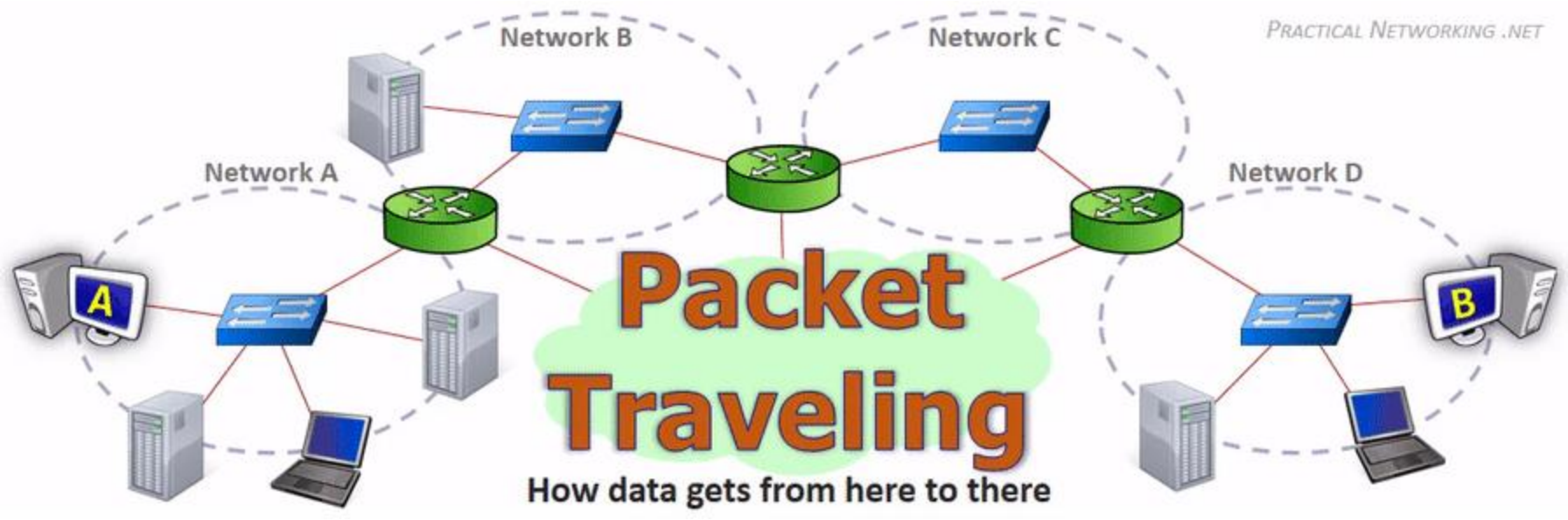
Discussion

The Router Logs

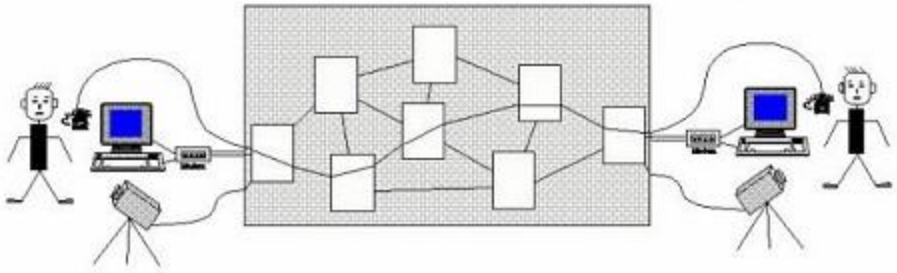
- The view we have in this table of traffic wouldn't be available to a normal user on the network -- this is a simulation after all -- but it WOULD be visible to the router.

Discussion

- A router needs to read the header of the message to route it correctly *
- But the contents of the message are right there too, and if it wanted, a router could read all the messages going across it, just like you did.
- This should raises some questions about privacy and security -- we'll talk more about this later.

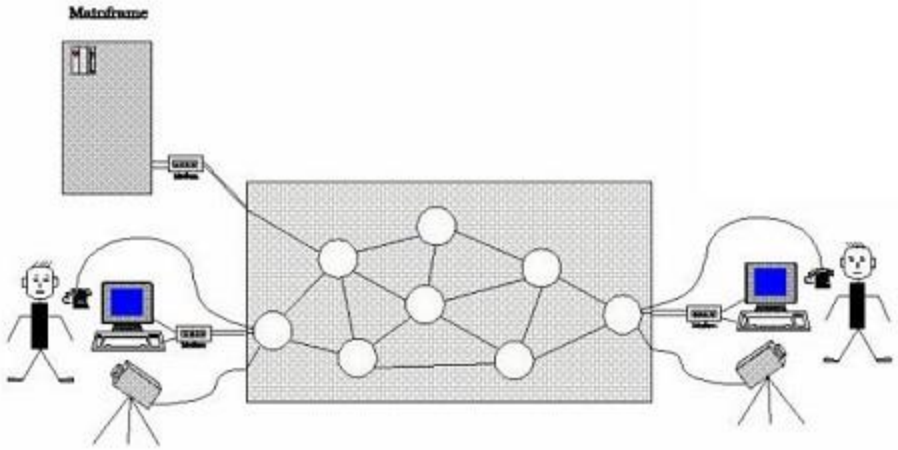

Network B
Network C
PRACTICAL NETWORKING .NET
Network A
Network D
A
B
Packet Traveling
How data gets from here to there

Circuit Switched Network

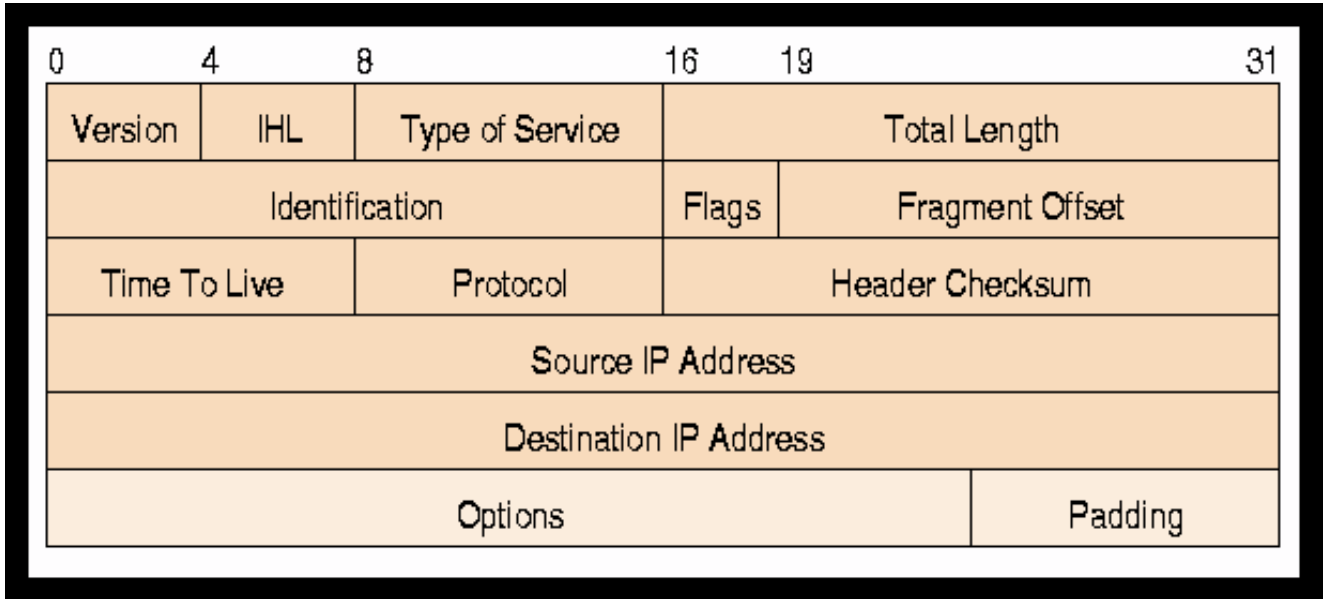


Receiver Caller

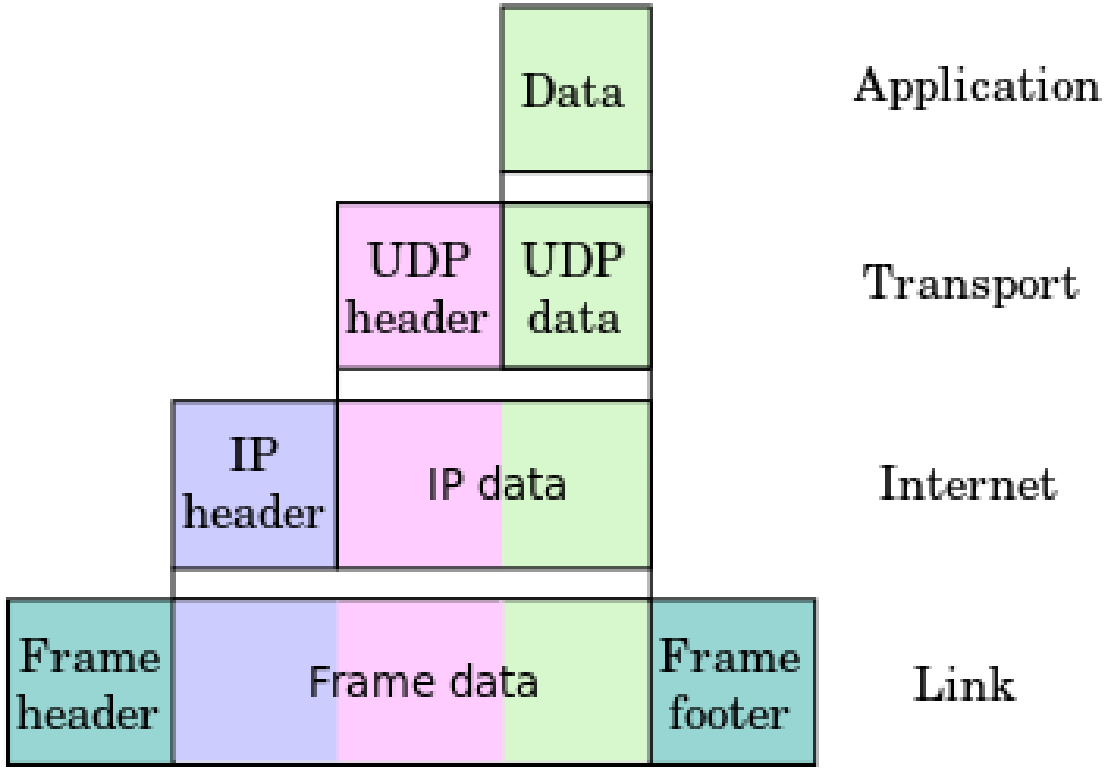
Packet Switched Network



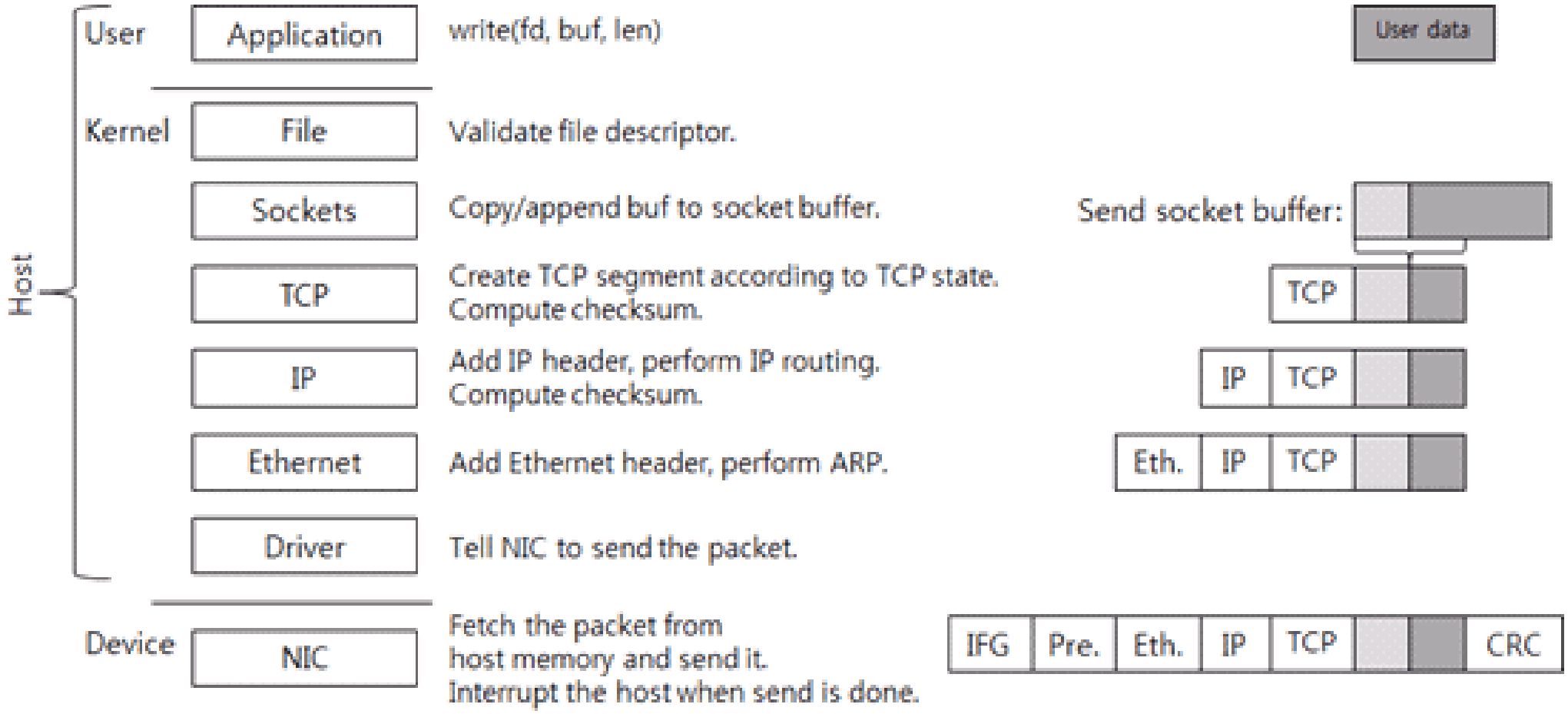
Receiver Caller



IP Packet Structure

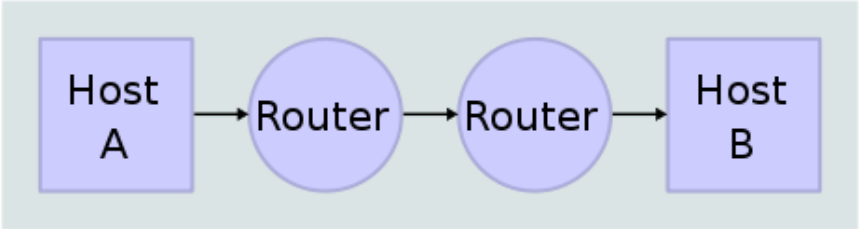


Encapsulation

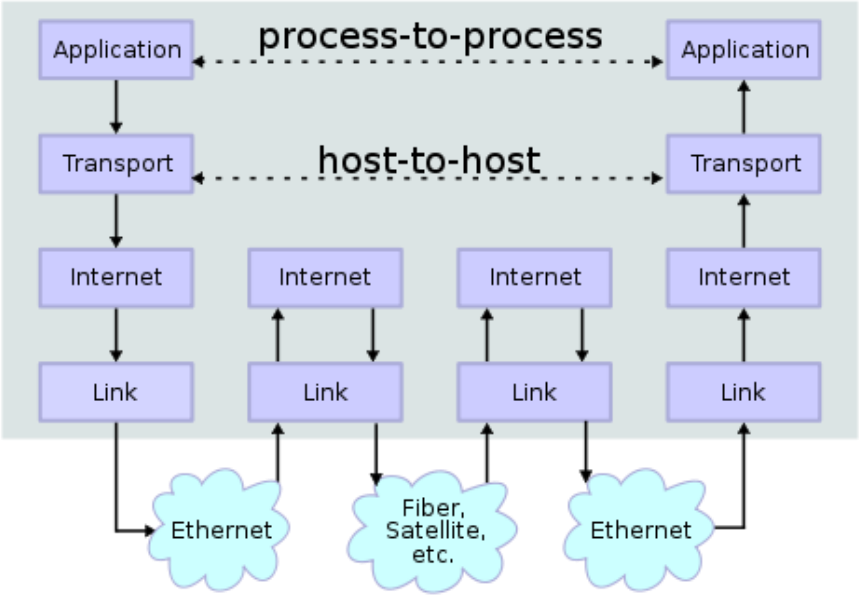


eC Learning Channel

Network Topology



Data Flow



Redundancy

- The simulations you used in today's lesson model many features of the actual Internet, most notably its redundant nature (there are many paths between locations).
- Identify these features with the following comments, and note that at the top of the next page you'll find a routing table that shows the different routes taken by the messages.

Source of Redundancy

- Encapsulation
- Retransmission
- Re-routing

Redundancy

- On the Internet Simulator we see messages appearing in the Router Logs multiple times (like in the image to the right).
- This is done to simulate a message passing through multiple routers on its way to its destination.
- A row is created each time it shows up at a new router.

Discussion

- If you trace carefully you'll notice that messages between two people **don't** always visit the same routers along the way.
- This is not a mistake; it’s modeled after the way the actual Internet was designed.
- Why might **the Internet have been designed to be flexible about how messages get from one person to another**? Why go through the trouble of creating multiple paths between users?

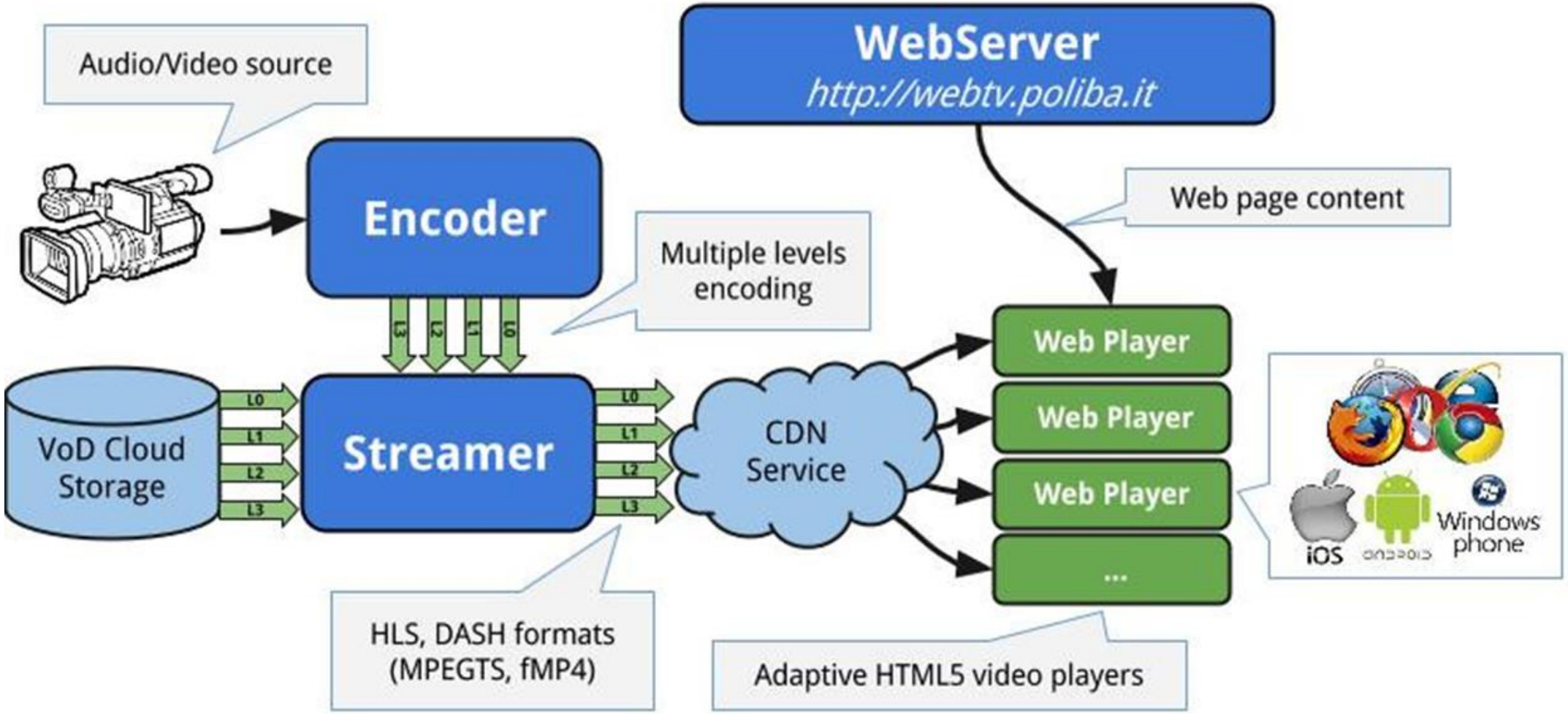
Discussion

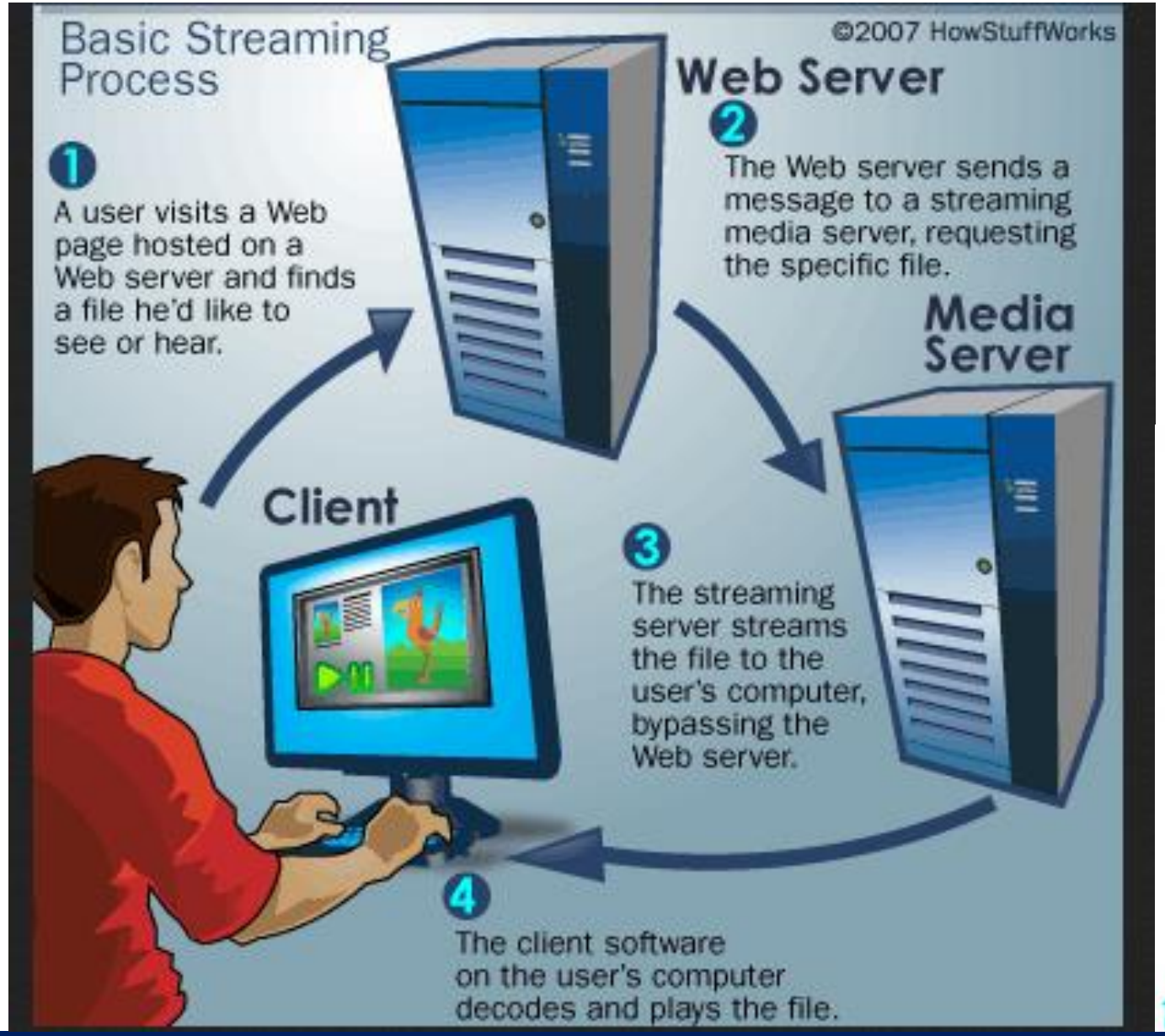
- **Efficiency:** Communication lines can become backed up with traffic.
- The ability to choose whichever one currently has the least traffic improves the speed of transmission.
- **Reliability:** Having many redundant paths means if one goes down, there are many others that a message can take.

Discussion

- Benefits of redundancy in a routed system, to provide many paths between users of the network.
- Routed messages aren't seen by all users, but they can still be seen by owners of the router.
- If you control all the routers, you entirely control the way messages travel through the network, and additionally can see all the traffic.

BUFFERING...


Audio/Video source
WebServer
http://webtv.poliba.it
Encoder
Web page content
Multiple levels encoding
L3
L2
L1
L0
VoD Cloud Storage
Streamer
CDN Service
Web Player
Web Player
Web Player
...
iOS
Windows phone
HLS, DASH formats (MPEGTS, fMP4)
Adaptive HTML5 video players


Basic Streaming Process
©2007 HowStuffWorks
Web Server
1
A user visits a Web page hosted on a Web server and finds a file he'd like to see or hear.
2
The Web server sends a message to a streaming media server, requesting the specific file.
Media Server
Client
3
The streaming server streams the file to the user's computer, bypassing the Web server.
4
The client software on the user's computer decodes and plays the file.

veoh
dish
NETFLIX
YouTube
amazon.com
hulu
vudu