Network Models
OSI vs. TCP/IP

Network Models

- Using a formal model allows us to deal with various aspects of Networks abstractly.
- We will look at two popular models
  - OSI reference model
  - TCP/IP model
- Both models are based on the concept of *layering*.

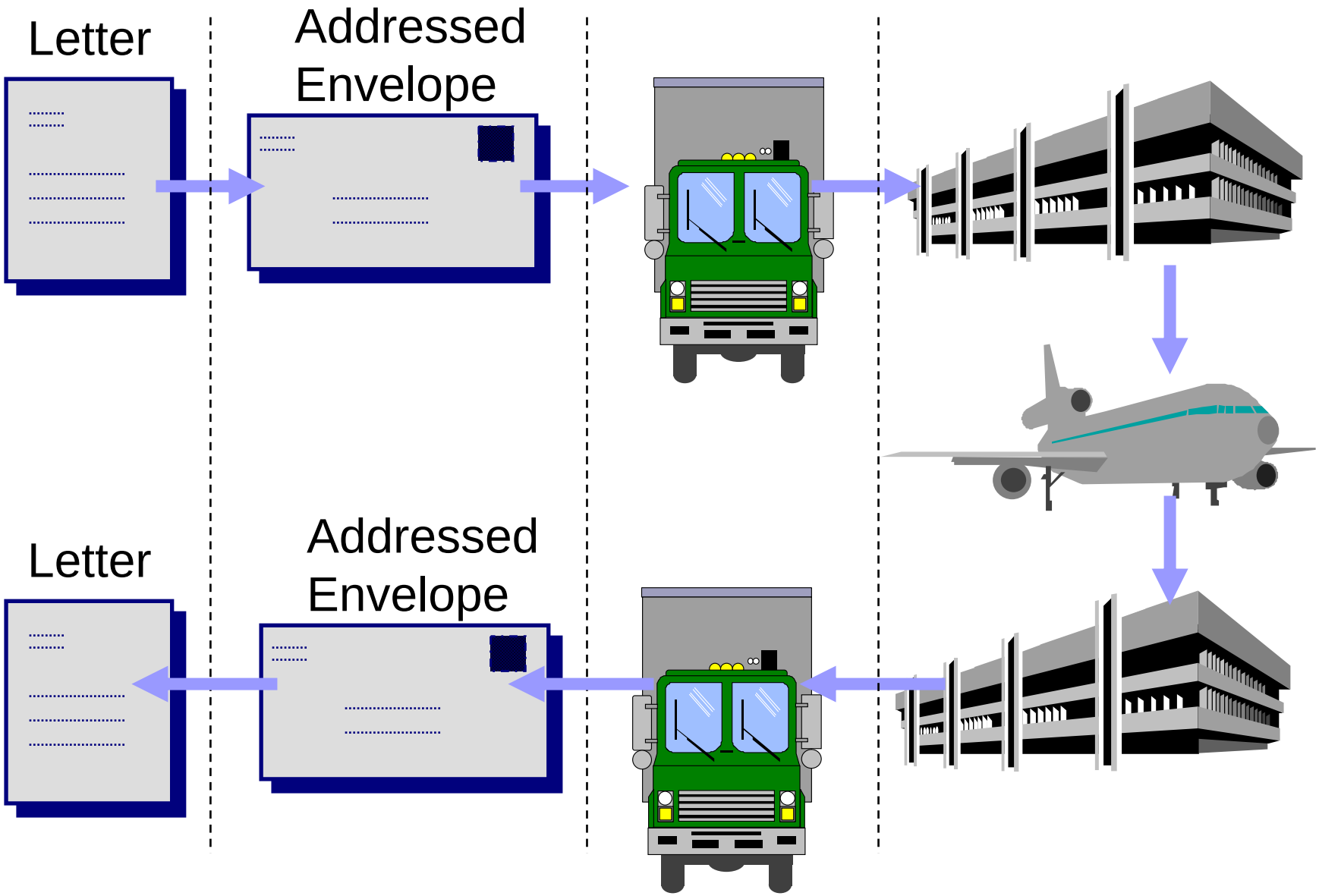
Layering

- Divide a task into sub-tasks and then solve each sub-task independently.
- Establishing a well defined interface between layers makes porting easier.
- Major Advantages:
  - Code Reuse
  - Extensibility

Layering Example: Federal Express

- Letter in envelope, address on outside
- FedX guy adds addressing information, barcode.
- Local office drives to airport and delivers to hub.
- Sent via airplane to nearest city.
- Delivered to right office
- Delivered to right person

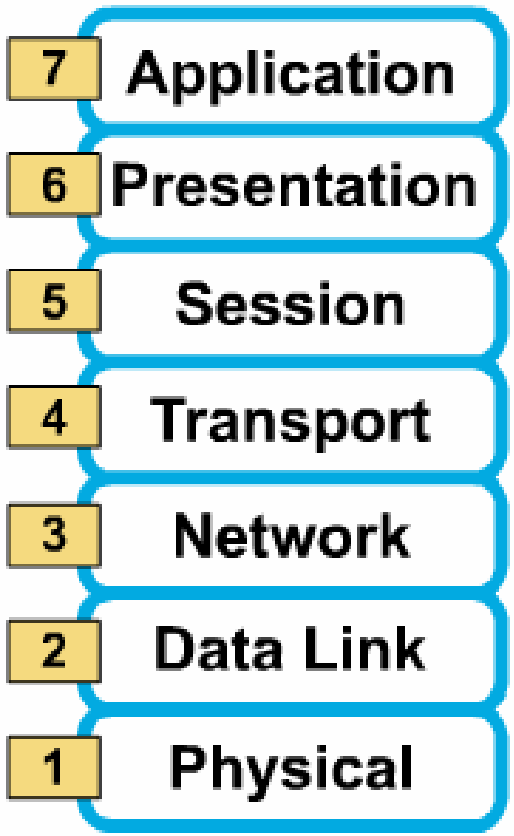
FedX Layers



OSI Reference Model

The International Standards Organization (ISO) proposal for the standardization of the various protocols used in computer networks (specifically those networks used to connect open systems) is called the Open Systems Interconnection Reference Model (1984), or simply the OSI model.

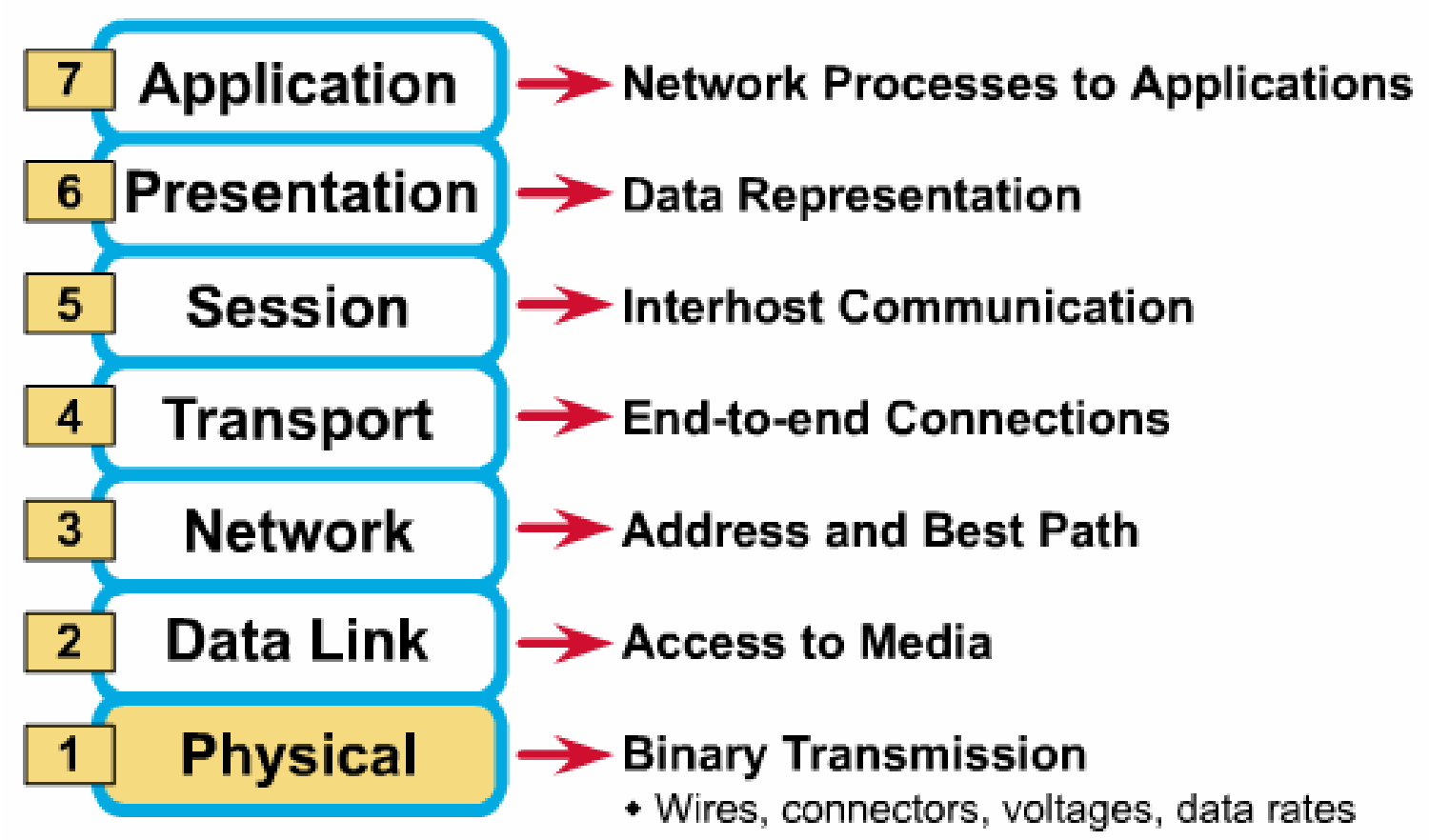
Why a Layered Model?



- Reduces complexity
- Standardizes interfaces
- Facilitates modular engineering
- Ensures interoperable technology
- Accelerates evolution
- Simplifies teaching and learning

All People Seem To Need Data Processing

Layers with Functions



The Seven Layers of the OSI Reference Model

- The application (upper) layers
  - Layer 7: Application
  - Layer 6: Presentation
  - Layer 5: Session
- The data-flow (lower) layers
  - Layer 4: Transport
  - Layer 3: Network
  - Layer 2: Data link
  - Layer 1: Physical

The Application (Upper) Layers

- Application
  - User interface
  - Examples – Telnet, HTTP
- Presentation
  - How data is presented
  - Special processing, such as encryption
  - Examples – ASCII, EBCDIC, JPEG
- Session
  - Keeping different applications' data separate
  - establishes, manages, and terminates sessions between applications.

The Data-Flow (Lower) Layers

- Transport
  - Reliable or unreliable delivery
  - Error correction before transmit
  - Examples: TCP, UDP, SPX
- Network
  - Provide logical addressing which routers use for path determination
  - Examples: IP, IPX

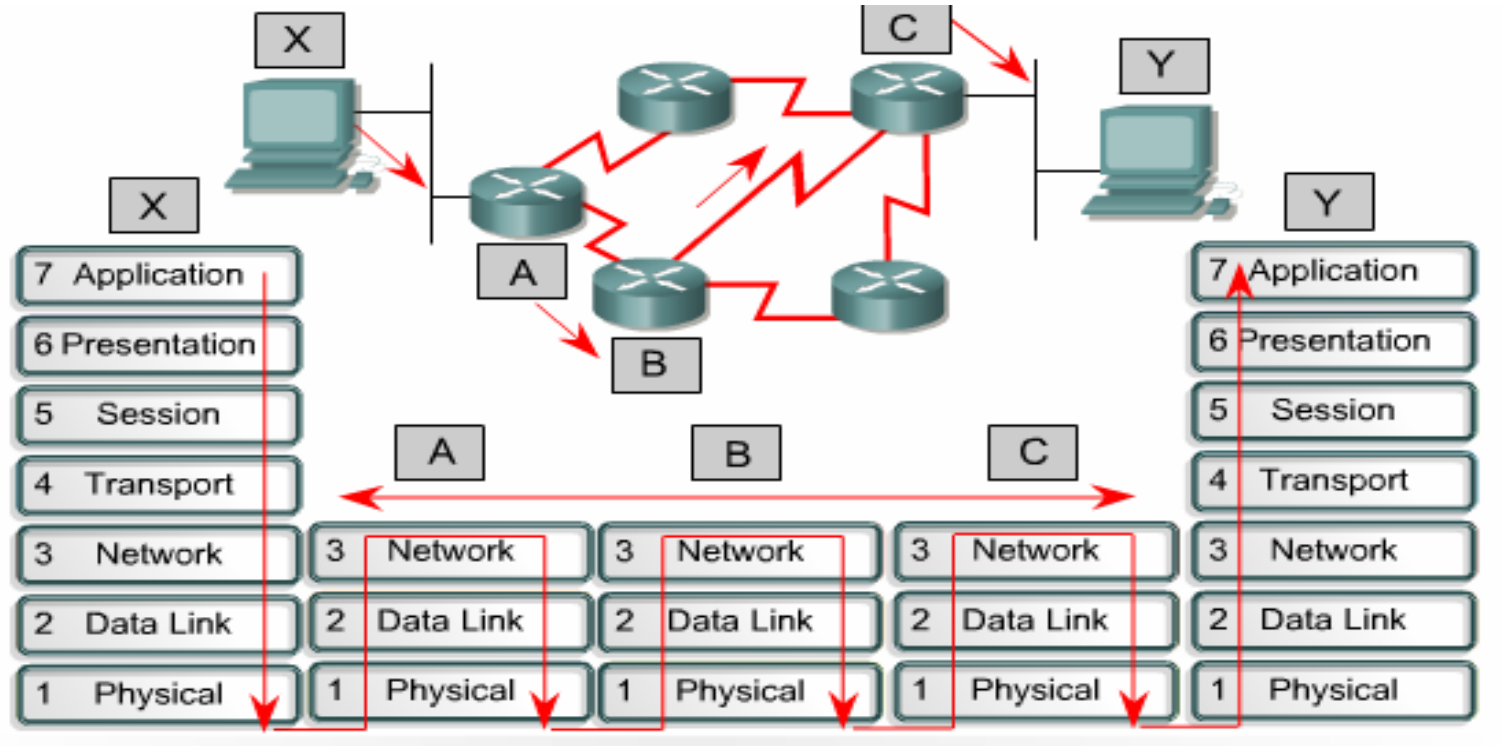
The Lower Layers (cont.)

- Data link
  - Combines bits into bytes and bytes into frames
  - Access to media using MAC address
  - Error detection not correction
  - Examples: 802.3/802.2, HDLC
- Physical
  - Moves bits between devices
  - Specifies voltage, wire speed, and pinout cables
  - Examples: EIA/TIA-232, V.35

Layering & Headers

- Each layer needs to add some control information to the data in order to do it’s job.
- This information is typically prepended to the data before being given to the lower layer.
- Once the lower layers deliver the the data and control information - the peer layer uses the control information.

Packet Propagation



Each router provides its services to support upper layer functions.

Addresses

- Each communication endpoint must have an address.
- Consider 2 processes communicating over an internet:
  - the network must be specified
  - the host (end-system) must be specified
  - the process must be specified.

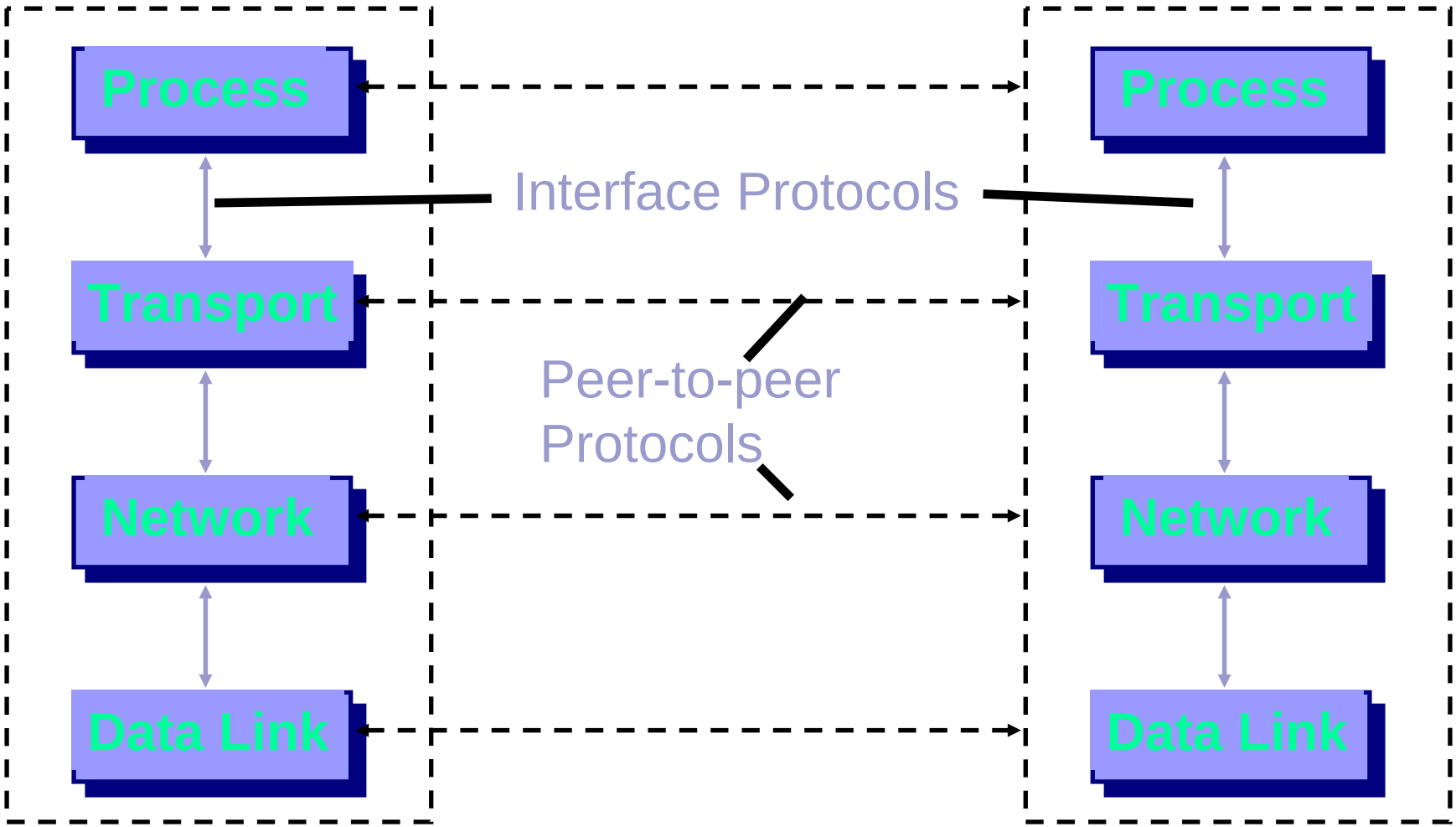
Addresses at Layers

- Physical Layer: no address necessary
- Data Link Layer - address must be able to select any host on the network (MAC).
- Network Layer - address must be able to provide information to enable routing (IP).
- Transport Layer - address must identify the destination process (PORT).

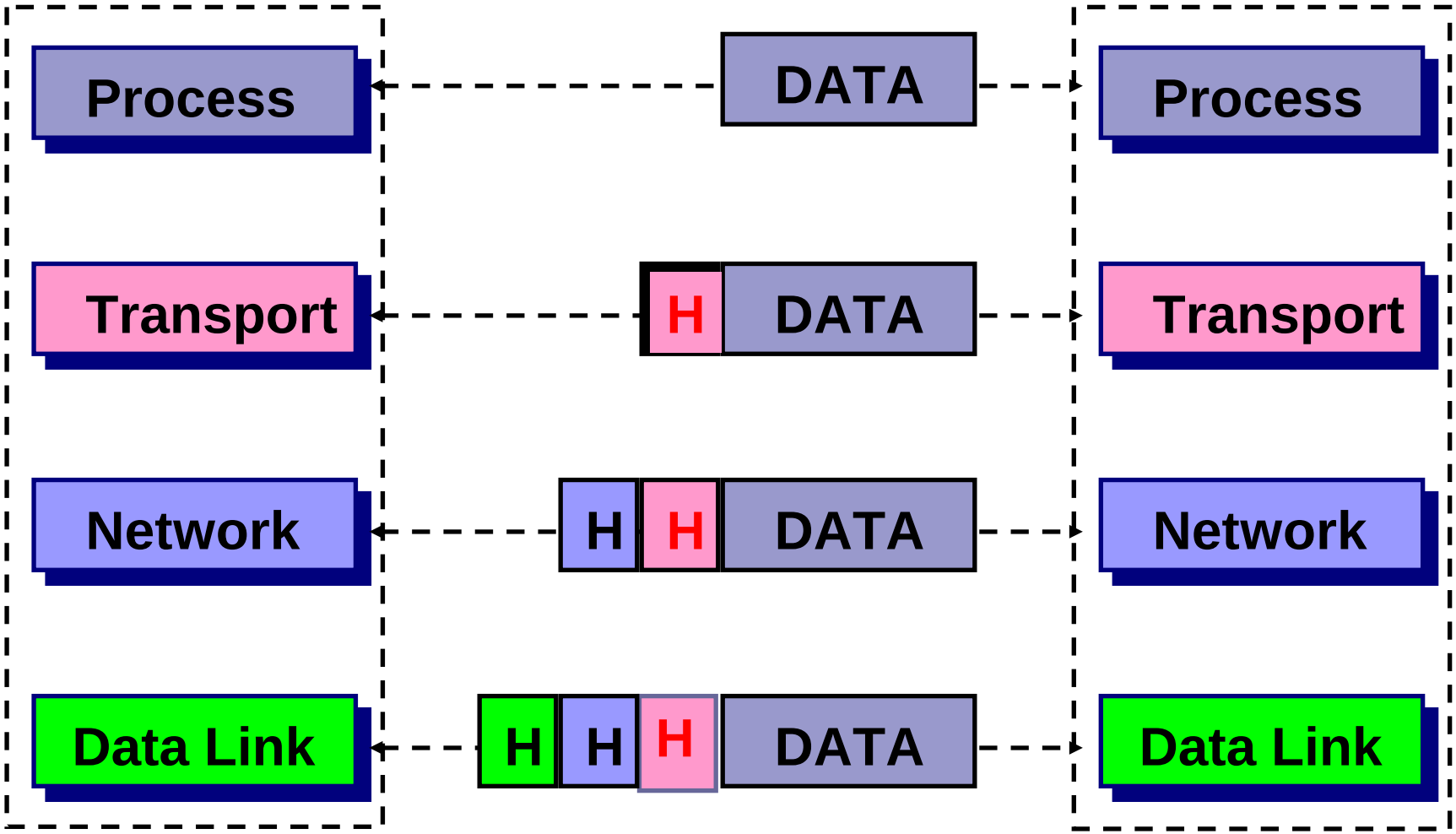
Broadcasts

- Many networks support the notion of sending a message from one host to all other hosts on the network.
- A special address called the “broadcast address” is often used.

TCP/IP Network Model



Headers
(Encapsulation ➔ De-Encapsulation)



Terms Often Used

- Frames: think data link layer
- Packets: think network layer
- Datagrams: think IP
- Segments: think TCP
- Cells: think ATM
- Layer <x>: refer to reference models

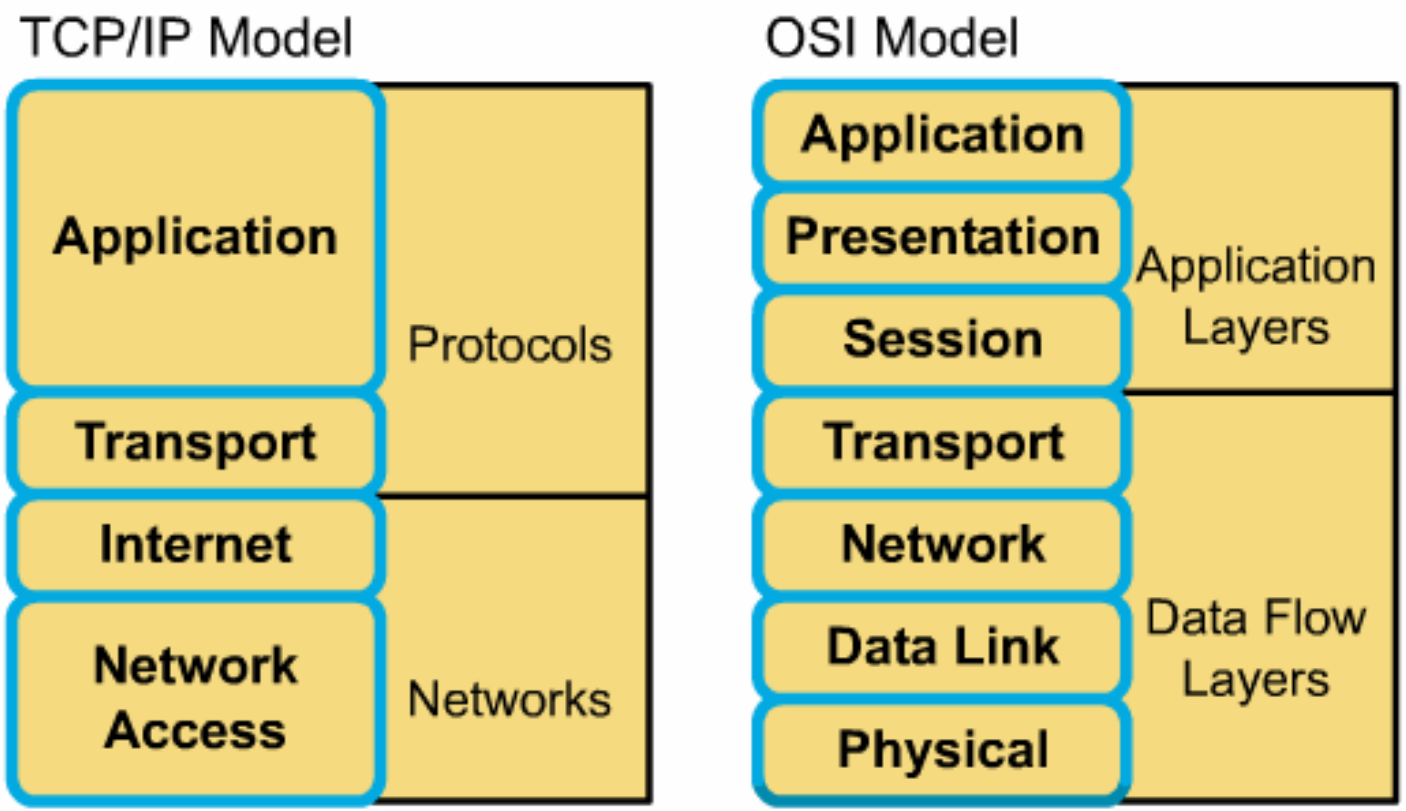
What's a Protocol?

- An agreed upon convention for communication.
  - both endpoints need to *understand* the protocol.
- Protocols must be formally defined and unambiguous!
- We will study some of the important existing TCP/IP protocols.

Interface Vs. Peer-to-peer Protocols

- Interface protocols describe the communication between layers on the same endpoint.
- Peer-to-peer protocols describe communication between peers at the same layer.

OSI Model and TCP/IP Model



Differences of the OSI and TCP/IP models

- TCP/IP combines the presentation and session layer into its application layer.
- TCP/IP combines the OSI data link and physical layers into one layer.
- TCP/IP appears simpler because it has fewer layers.
- TCP/IP transport layer using UDP does not always guarantee reliable delivery of packets as the transport layer in the OSI model does.

Programs & Processes

- A *program* is an executable file.
- A *process* or *task* is an instance of a program that is being executed.
- A single program can generate multiple processes.

Client - Server

- A *server* is a process - not a machine !
- A server waits for a request from a client.
- A client is a process that sends a request to an existing server and (usually) waits for a reply.

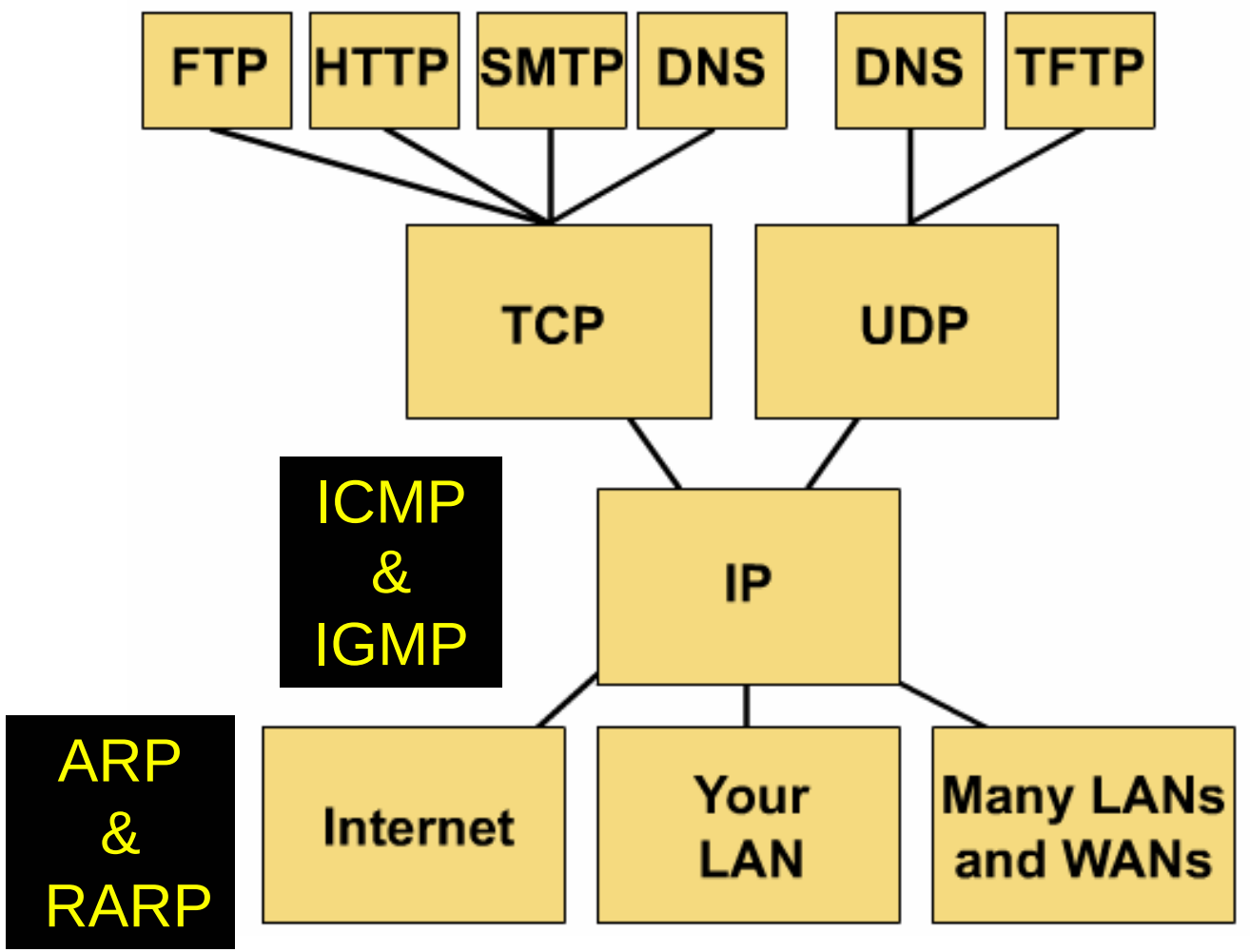
Client - Server Examples

- Server returns the time-of-day.
- Server returns a document.
- Server prints a file for client.
- Server does a disk read or write.
- Server records a transaction.

Servers

- Servers are generally more complex (more interesting).
- 2 Basic types of servers:
  - *Iterative* - server handles one client at a time.
  - *Concurrent* - server handles many clients at a time.

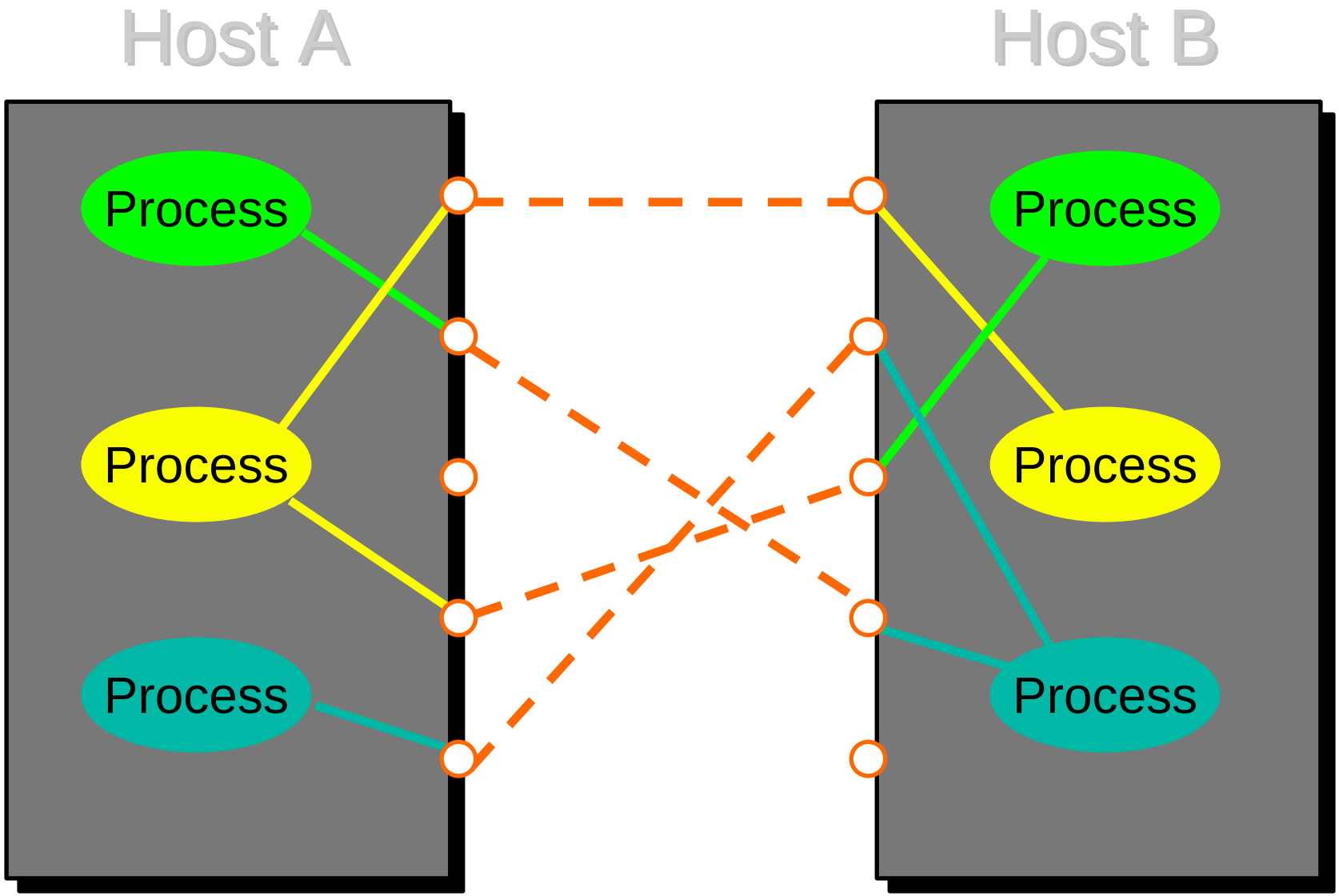
TCP/IP Protocol Graph



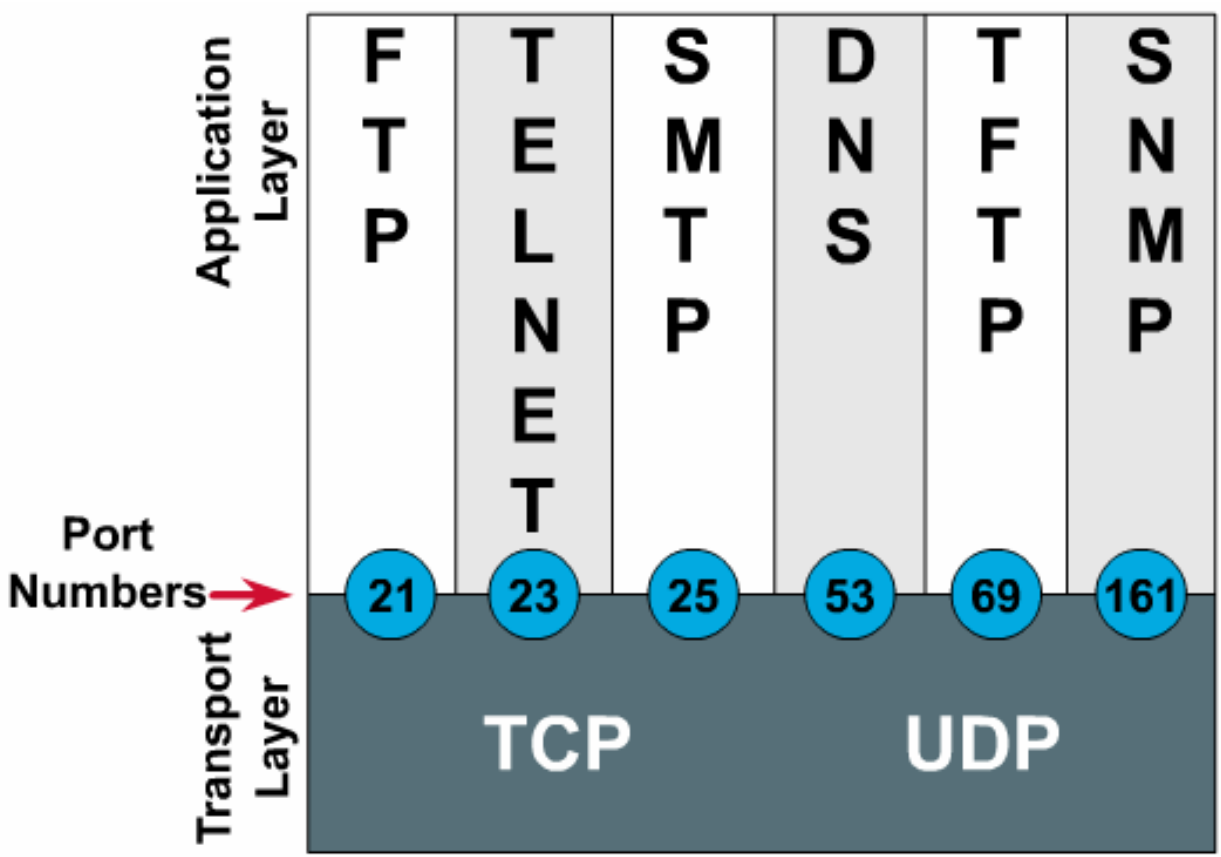
Ports

- TCP/IP uses an abstract destination point called a protocol port.
- Ports are identified by a positive integer.
- Operating systems provide some mechanism that processes use to specify a port.

Ports


Host A
Host B
Process
Process
Process
Process
Process
Process

Port Numbers



References

- Cisco Networking Academy Program (CCNA), Cisco Press.
- CSCI-5273 : Computer Networks, Dirk Grunwald, University of Colorado-Boulder
- CSCI-4220: Network Programming, Dave Hollinger, Rensselaer Polytechnic Institute.
- TCP/IP Illustrated, Volume 1, Stevens.
- Java Network Programming and Distributed Computing, Reilly & Reilly.
- Computer Networks: A Systems Approach, Peterson & Davie.
- http://www.firewall.cx
- http://www.javasoft.com