Introduction to Networking and the Internet

Survey: “What do you like most?”

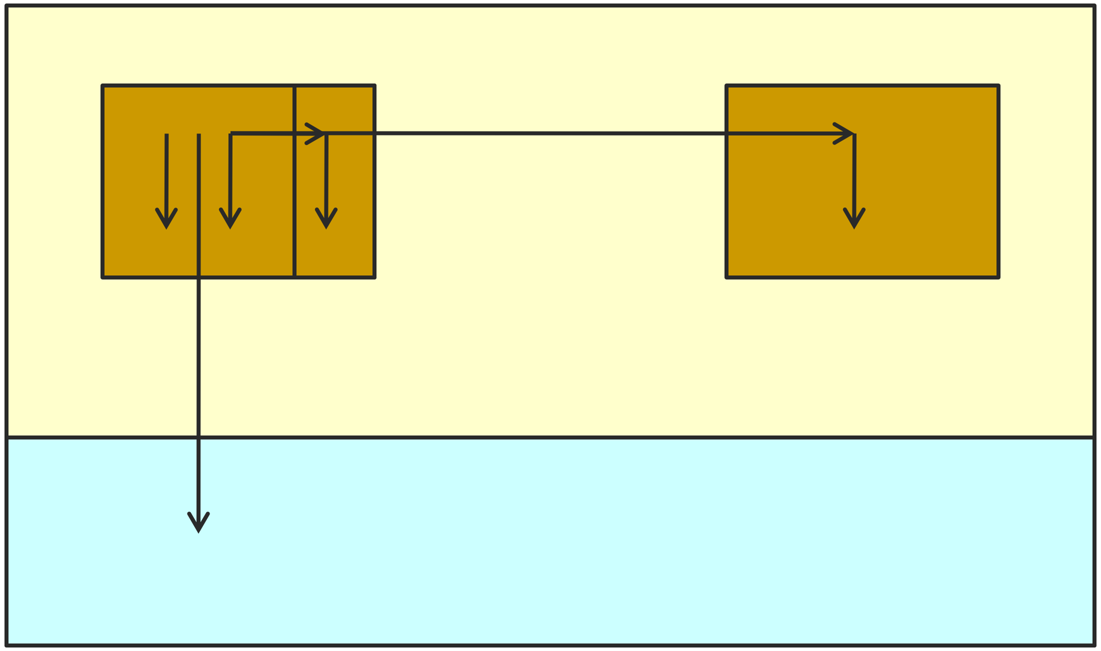
- Material
- Learned a lot from MPs
- MPs relevant to real world
- Staff available, helpful
- Course is fun
- Once I pass, I don’t have to take it again

Survey: “What do you like least?”

- Exams
  - Midterm too long
- MPs
  - MP instructions vague
  - MPs too long, hard
  - Solo MPs, no group work
  - Buggy
  - Busywork
  - Should be worth more
- HW
  - Slow grading/feedback
- Lectures
  - Not recorded
  - Boring
  - More theory, less man pages
  - Slides not useful alone
  - Examples too simplistic
- Organization
  - office hours should be more spread out

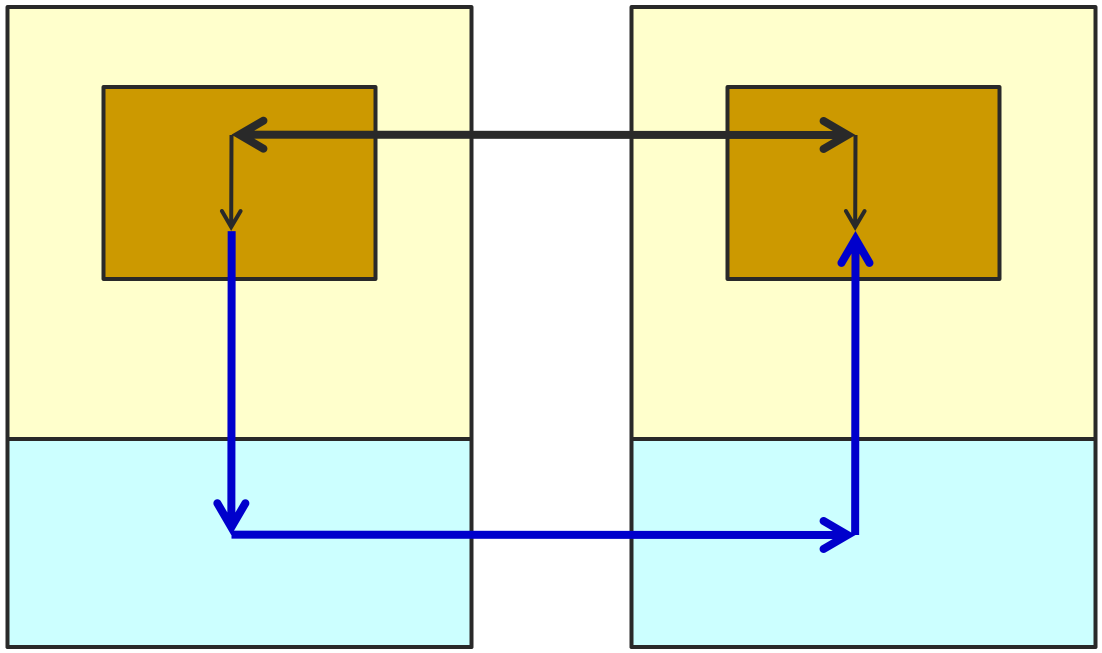
Where are we?

- Function calls, system calls, threads and processes



What's next?

- Networked communication and distributed applications

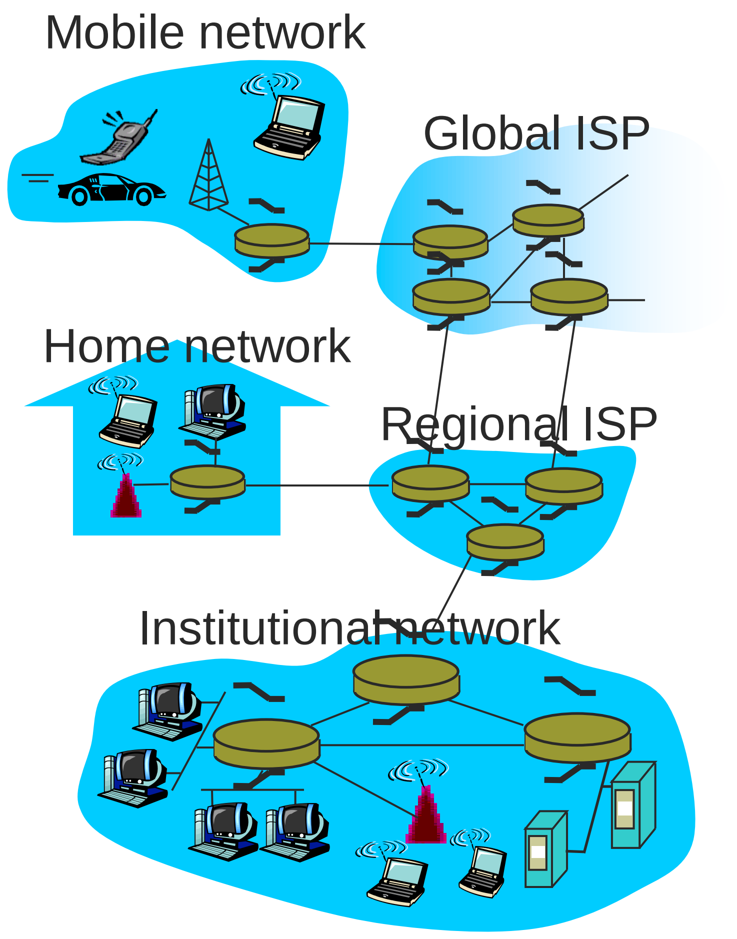


Introduction

- What is the Internet?
- Network edge
- What is a protocol?
- Protocol layers, service models

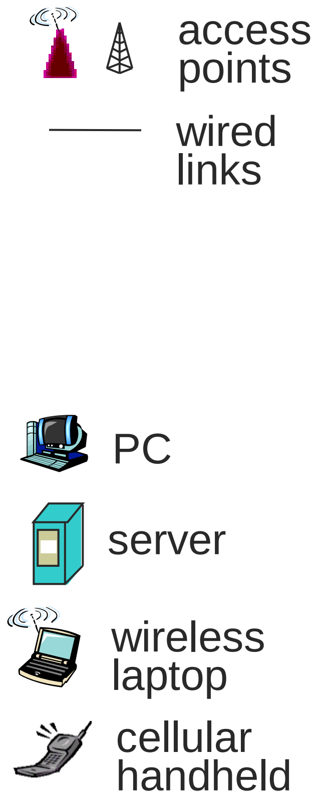
What is the Internet?

- Communication infrastructure
  - Enables distributed applications
  - Web, VoIP, email, games, e-commerce, file sharing
- Communication services
  - Provided to applications
  - Reliable data delivery from source to destination
  - “best effort” (unreliable) data delivery



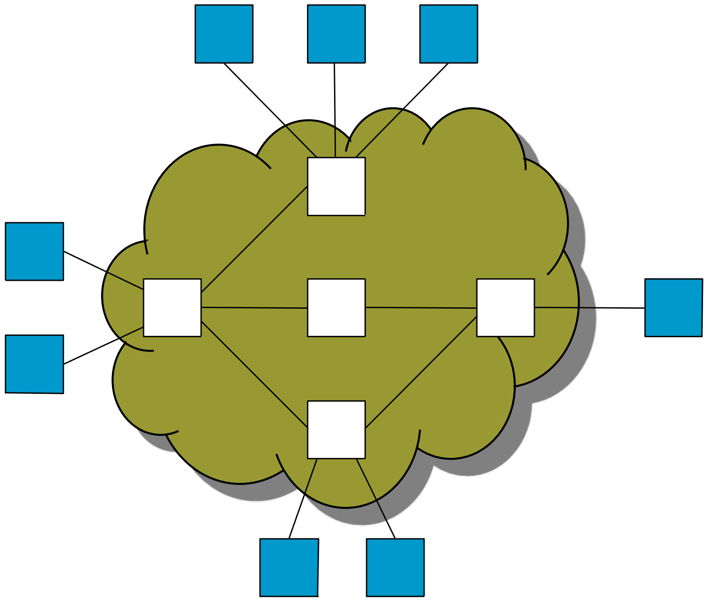
Connectivity

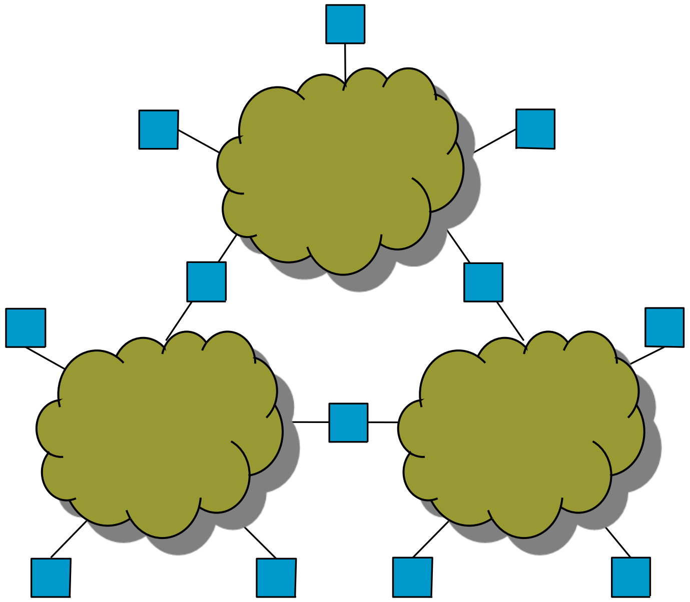
- Building Blocks
  - Links
    - coax cable, optical fiber, …
  - Nodes
    - workstations, routers, …



Indirect Connectivity

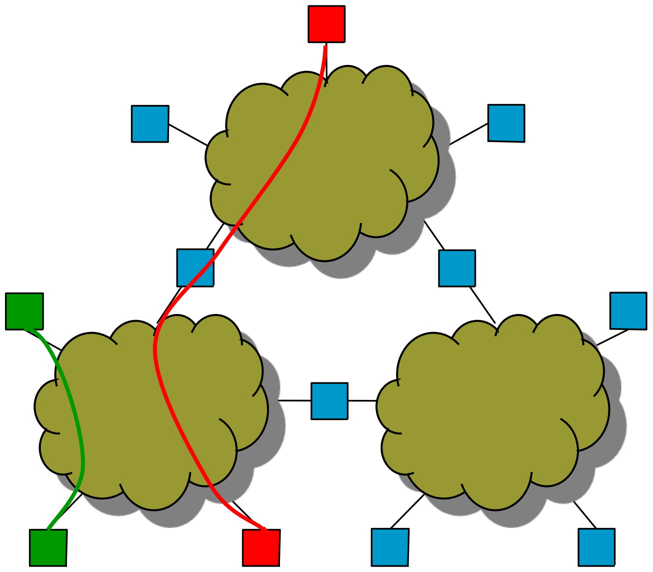
- Switched Networks
- Internetworks





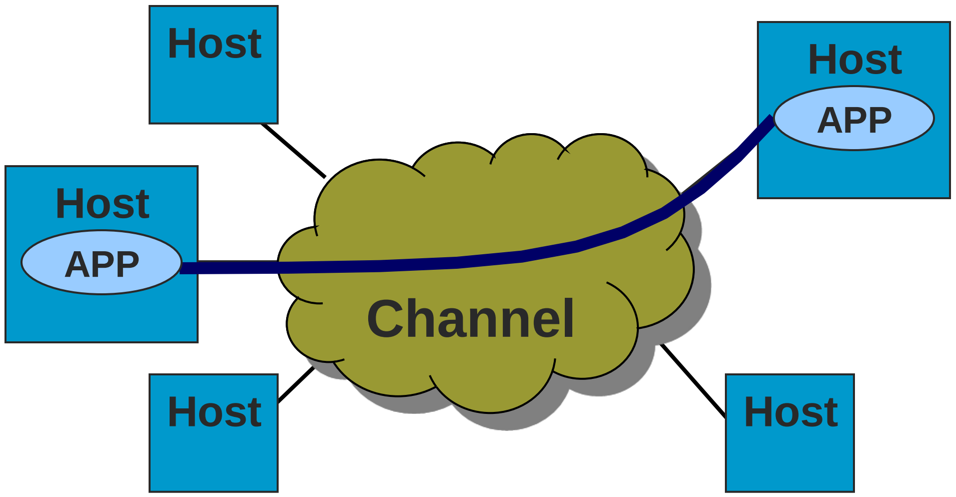
Network Service

- Goal
  - Transfer data between end systems
- Support For Common Services
  - Idea
    - Common services simplify the role of applications
    - Hide the complexity of the network without overly constraining the application designer
  - Semantics and interface depend on applications
    - Request/reply: FTP, HTTP
    - Message stream: audio, video



Channels

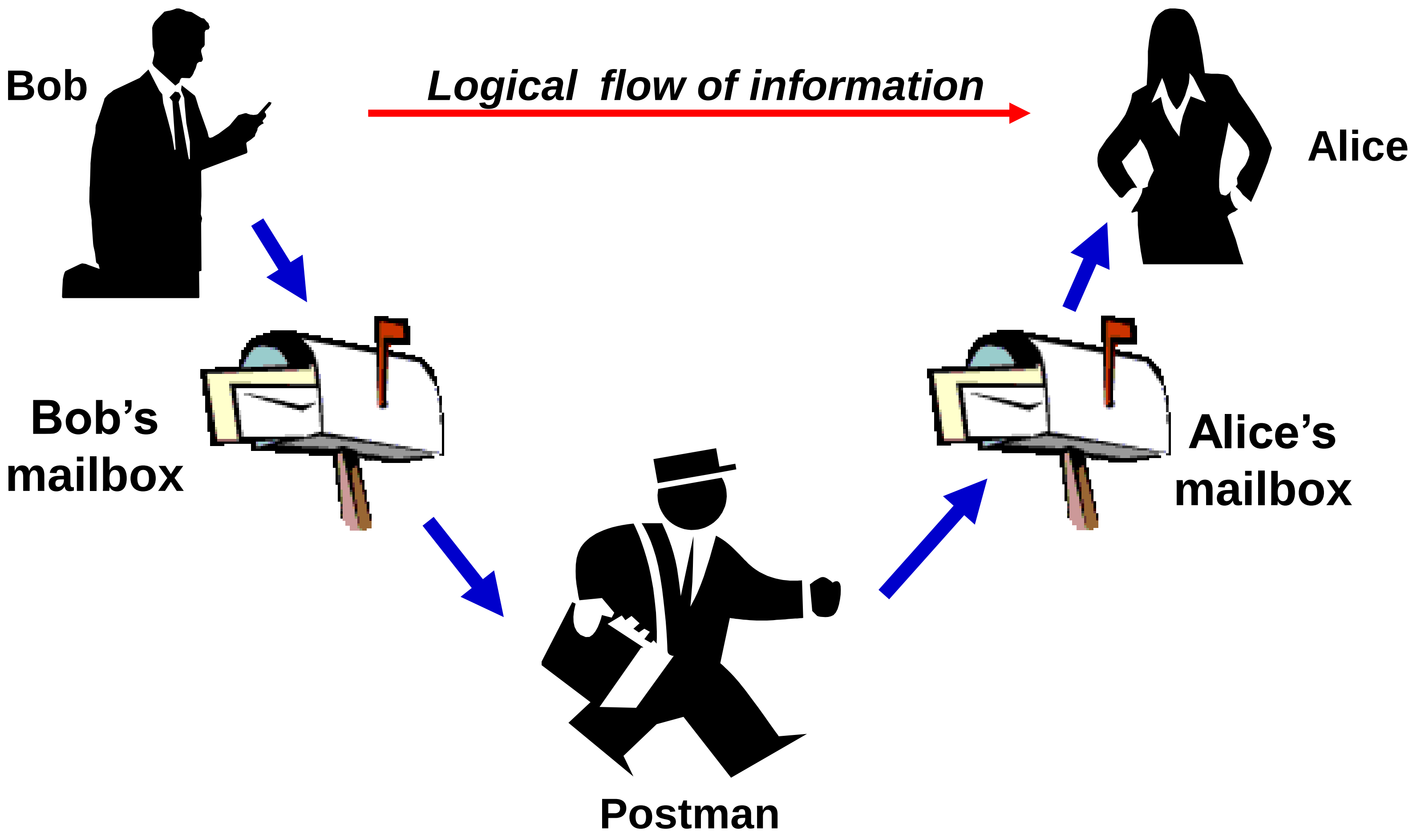
- Channel
  - The abstraction for application-level communication
- Idea
  - Turn host-to-host connectivity into process-to-process communication



Inter-process Communication

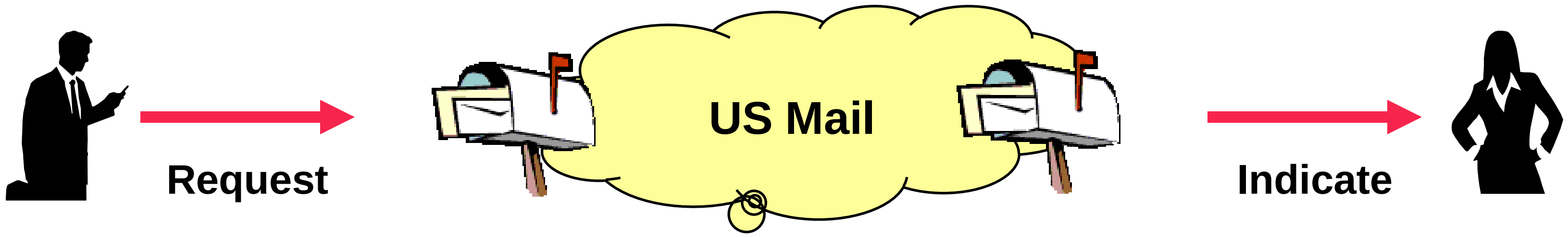
- Problems typically masked by communication channel abstractions
  - Bit errors (electrical interference)
  - Packet errors (congestion)
  - Link/node failures
  - Message delays
  - Out-of-order delivery
  - Eavesdropping
- Goal
  - Fill the gap between what applications expect and what the underlying technology provides

Example: Sending a Letter

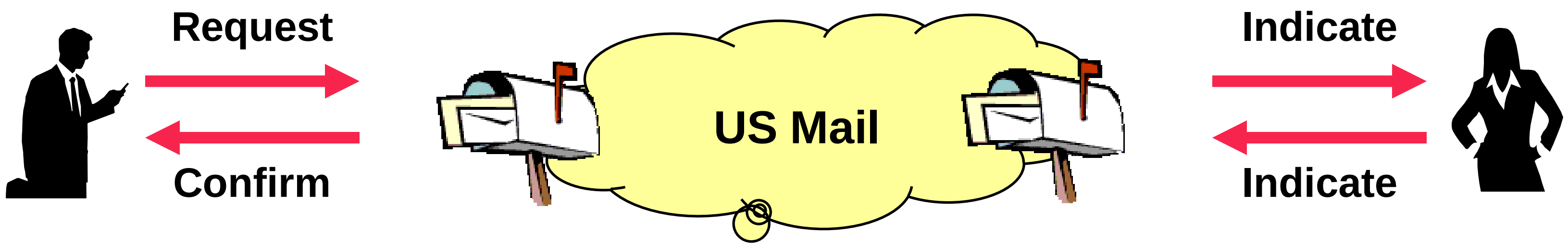


Services

- Unconfirmed service



- Acknowledged service



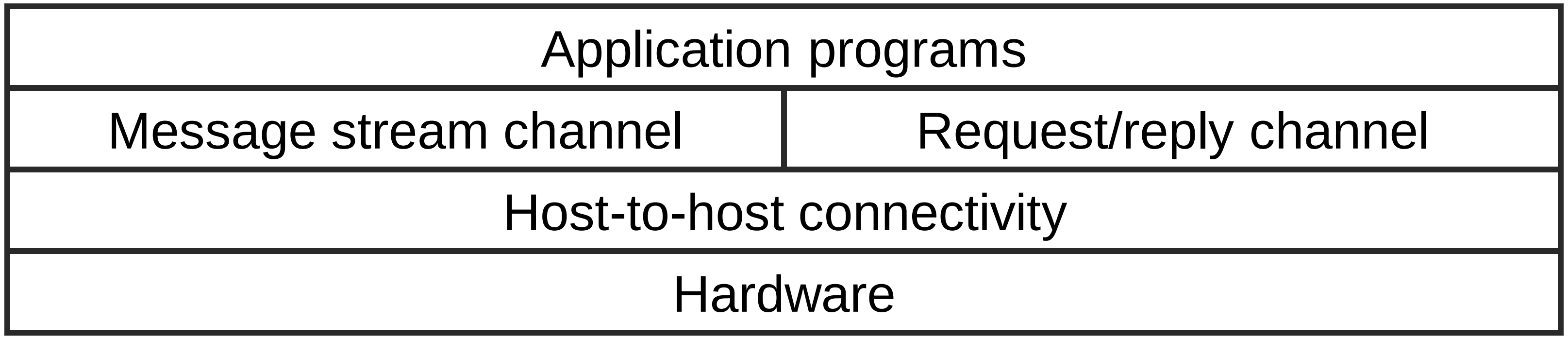
Network Architecture

- Networks are complex!
- Many “pieces”
  - Hosts
  - Routers
  - Links of various media
  - Applications
  - Protocols
  - Hardware, software

- Question
  - Is there any hope of organizing structure of network?
  - Or at least our discussion of networks?

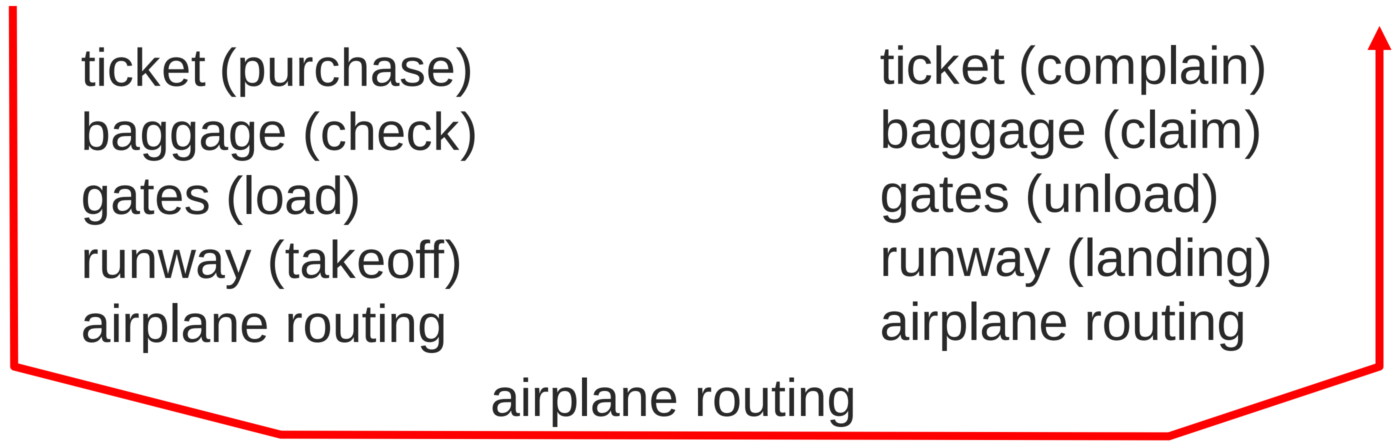
Abstraction through Layering

- Abstract system into layers:
  - Decompose the problem of building a network into manageable components
    - Each layer provides some functionality
  - Modular design provides flexibility
    - Modify layer independently
    - Allows alternative abstractions

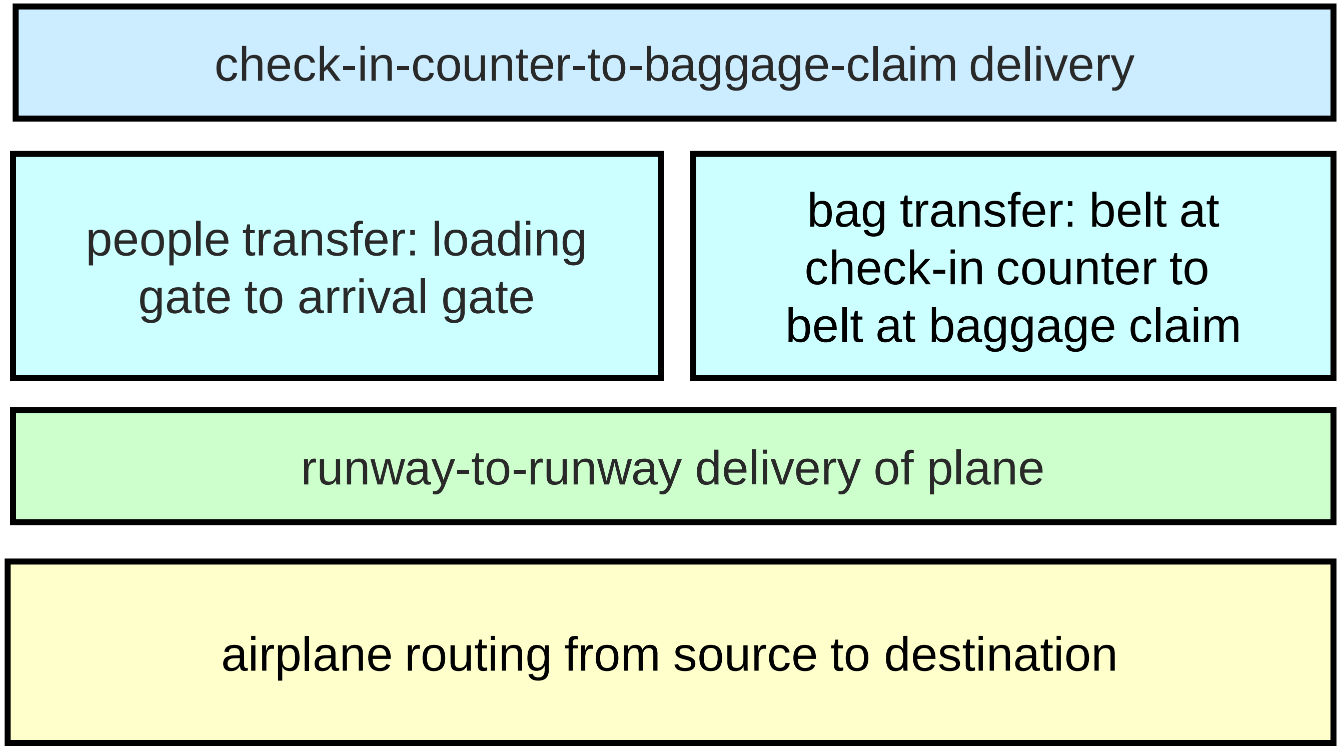


Example: Air Travel

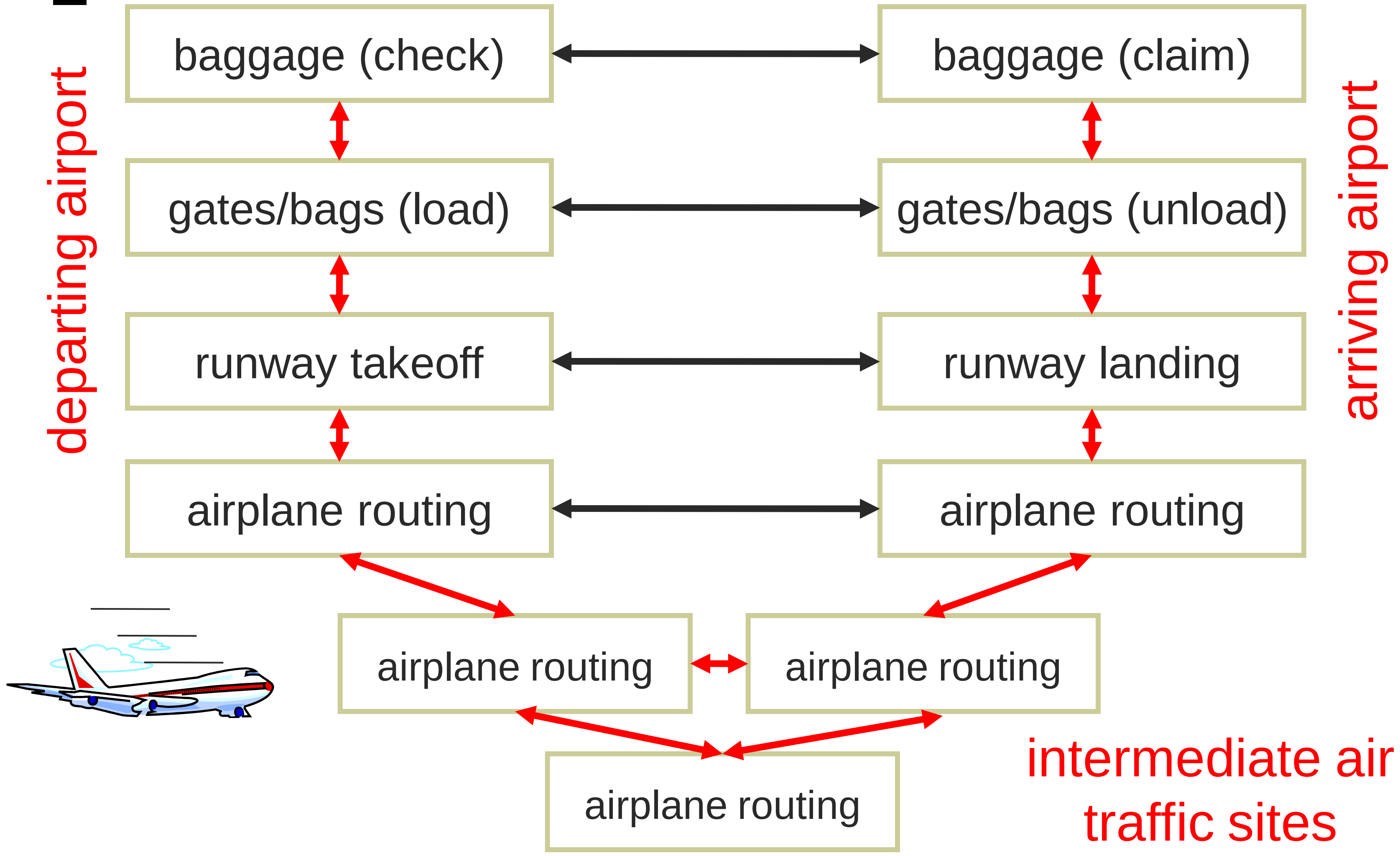
- Layers
  - Each layer implements a service
  - Via its own internal-layer actions
  - Relying on services provided by layer below



Air Travel: Services


check-in-counter-to-baggage-claim delivery
people transfer: loading gate to arrival gate
bag transfer: belt at check-in counter to belt at baggage claim
runway-to-runway delivery of plane
airplane routing from source to destination

Distributed Layering



Why layering?

- Complexity
  - Explicit structure allows identification, relationship of complex system’s pieces
- Modularity
  - Eases maintenance, updating of system
    - Change of implementation of layer’s service transparent to rest of system
    - e.g., change in gate procedure doesn’t affect rest of system
- Protocol
  - Instantiation of a layer!

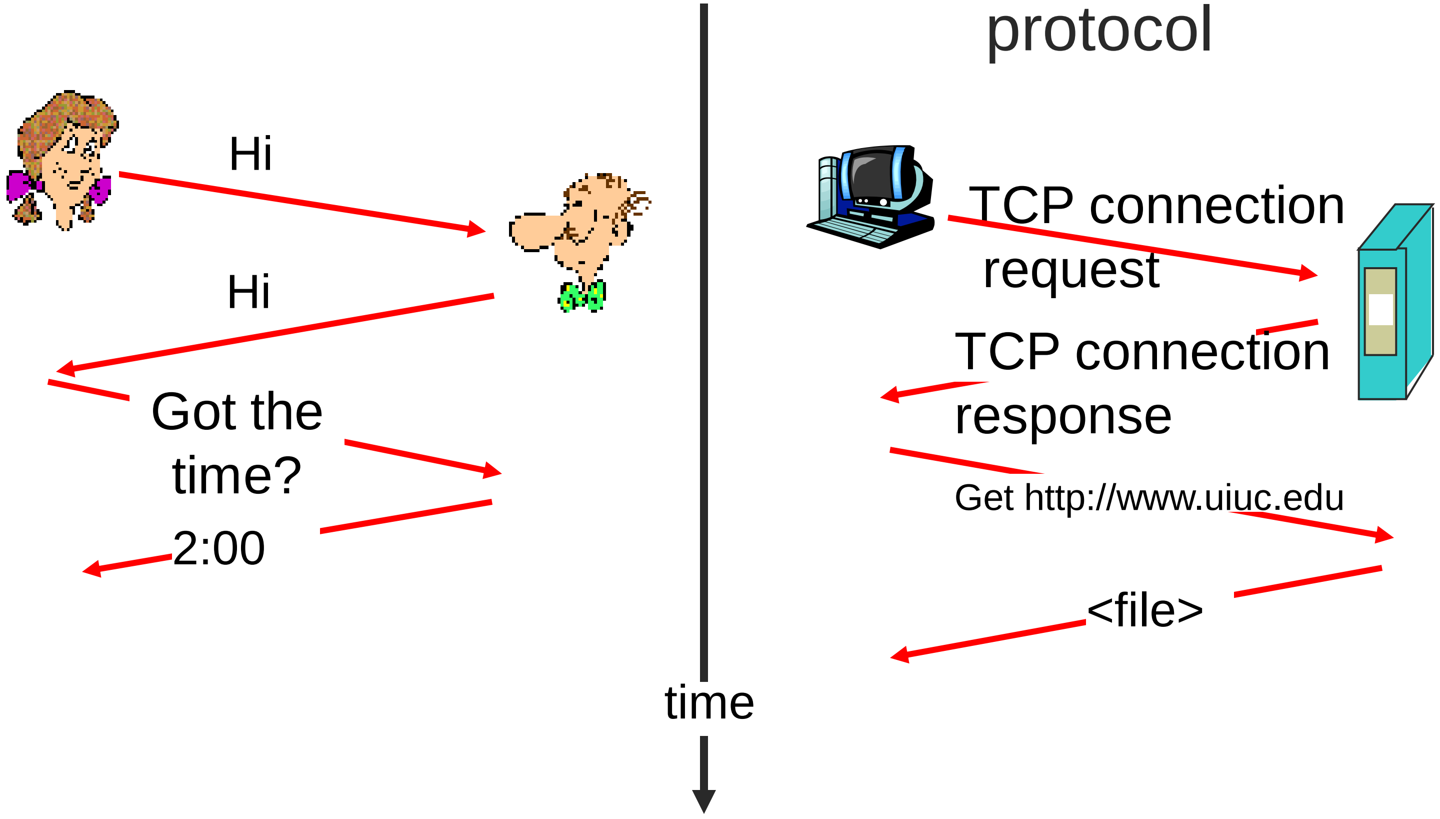
What is a Protocol?

- Protocols are defined by
  - Specific msgs sent
  - Specific actions taken when msgs received, or other events
- Human protocols
  - “what’s the time?”
  - “I have a question”
  - Introductions

- Protocols define
  - Format
  - Order of msgs sent and received among network entities
  - Actions taken on msg transmission, receipt
- Network protocols
  - Machines rather than humans
  - All communication activity in Internet is governed by protocols

What is a Protocol?

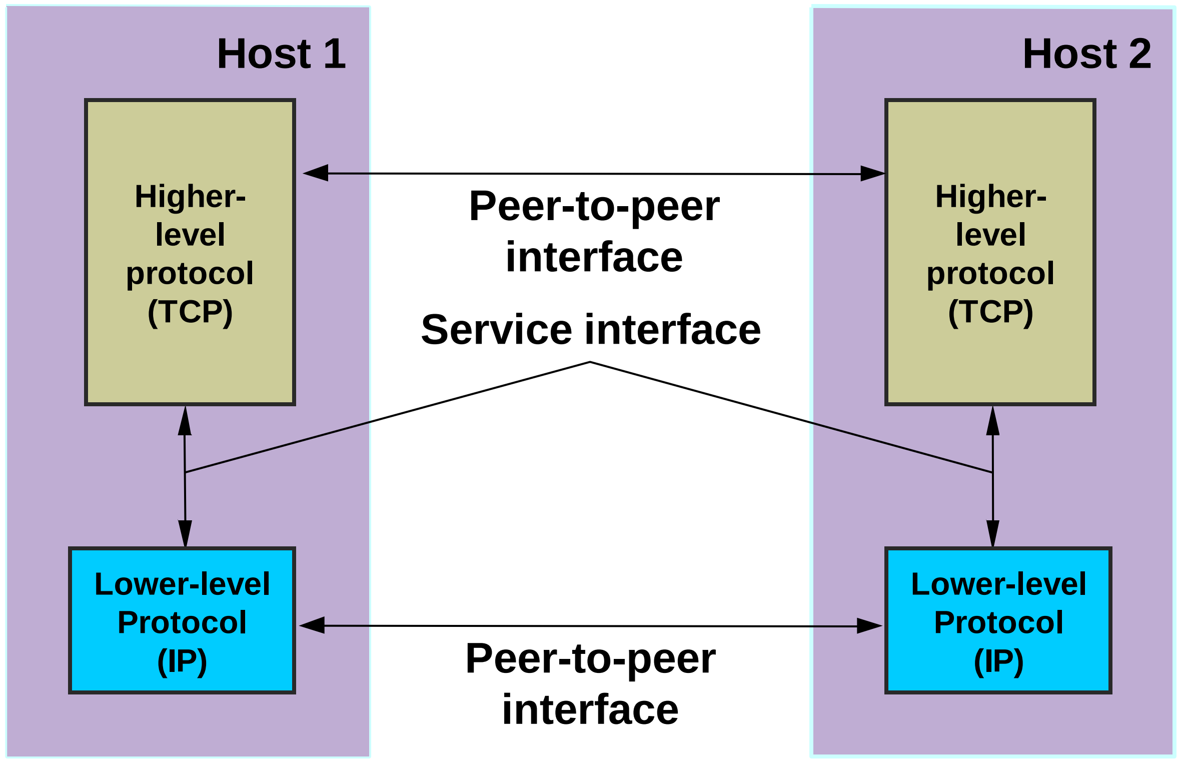
- A human protocol
- A computer network protocol



Network Protocols

- Definition
  - A protocol is an abstract object that makes up the layers of a network system
  - A protocol provides a communication service that higher-layer objects use to exchange messages
    - Service interface
      - To objects on the same computer that want to use its communication services
    - Peer interface
      - To its counterpart on a different machine
      - Peers communicate using the services of lower-level protocols

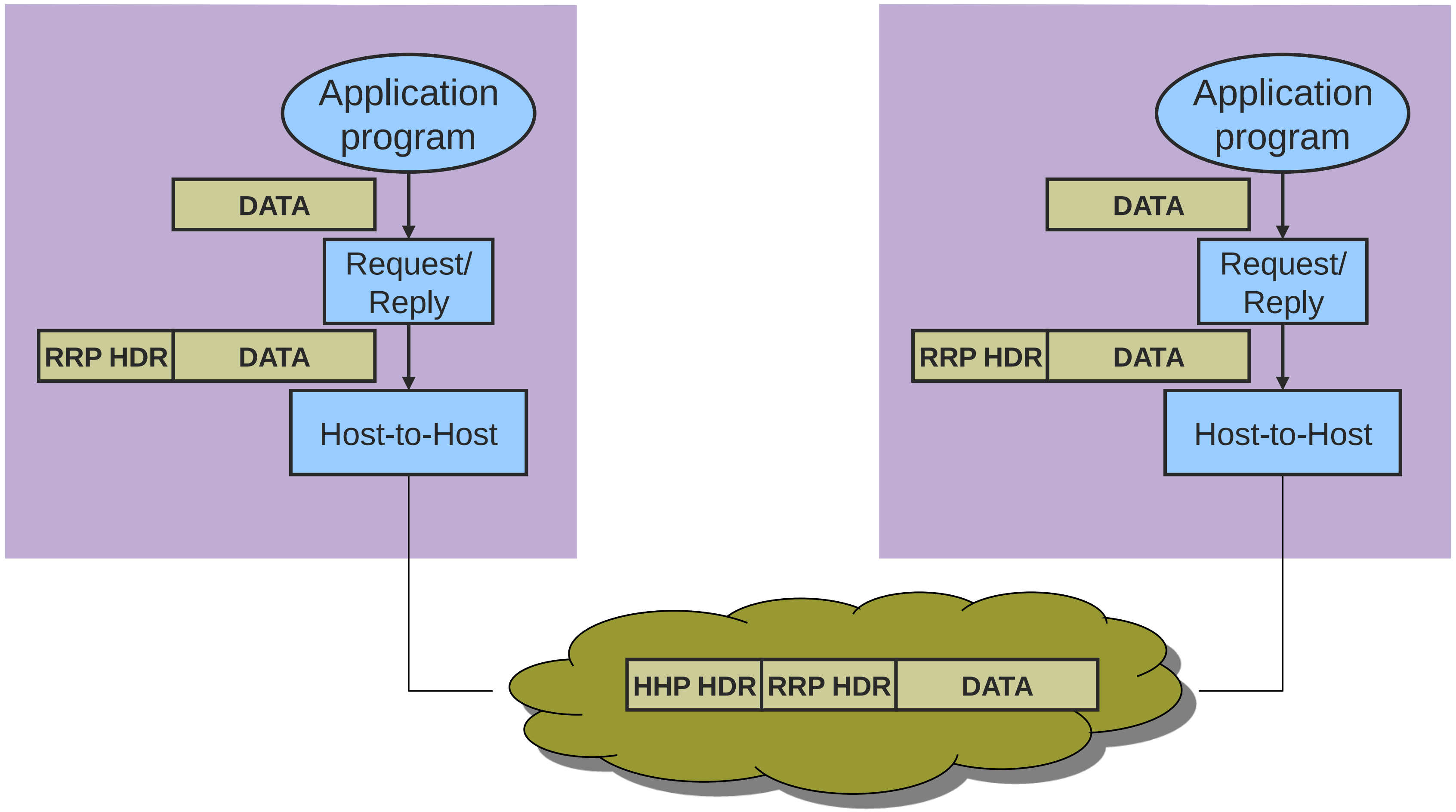
Interfaces


Host 1
Host 2
Higher-level protocol (TCP)
Higher-level protocol (TCP)
Peer-to-peer interface
Service interface
Lower-level Protocol (IP)
Lower-level Protocol (IP)
Peer-to-peer interface

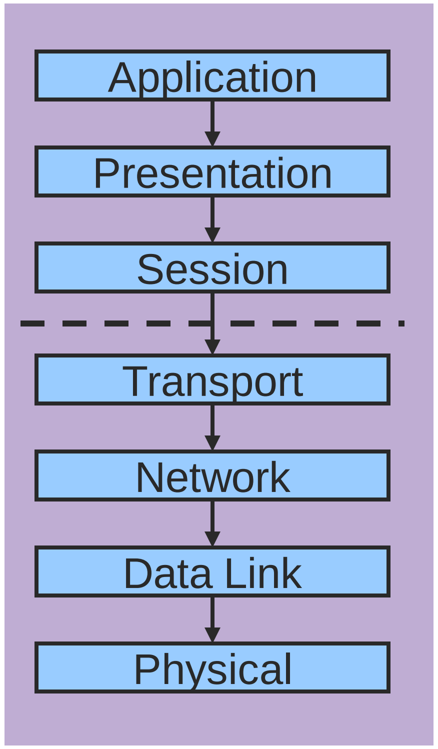
Layering Concepts

- Encapsulation
  - Higher layer protocols create messages and send them via the lower layer protocols
  - These messages are treated as data by the lower-level protocol
  - Higher-layer protocol adds its own control information in the form of headers or trailers
- Multiplexing and Demultiplexing
  - Use protocol keys in the header to determine correct upper-layer protocol

Encapsulation



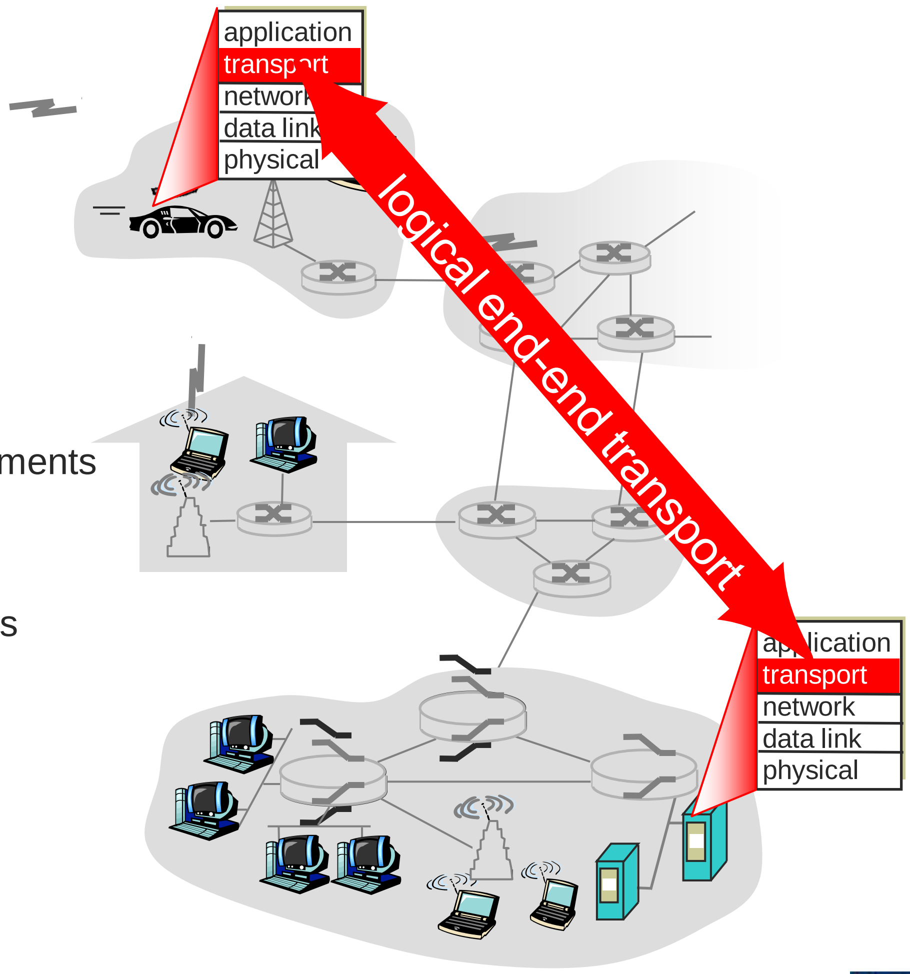
OSI Protocol Stack



Application:	Application specific protocols
Presentation:	Format of exchanged data
Session:	Name space for connection mgmt
Transport:	Process-to-process channel
Network:	Host-to-host packet delivery
Data Link:	Framing of data bits
Physical:	Transmission of raw bits

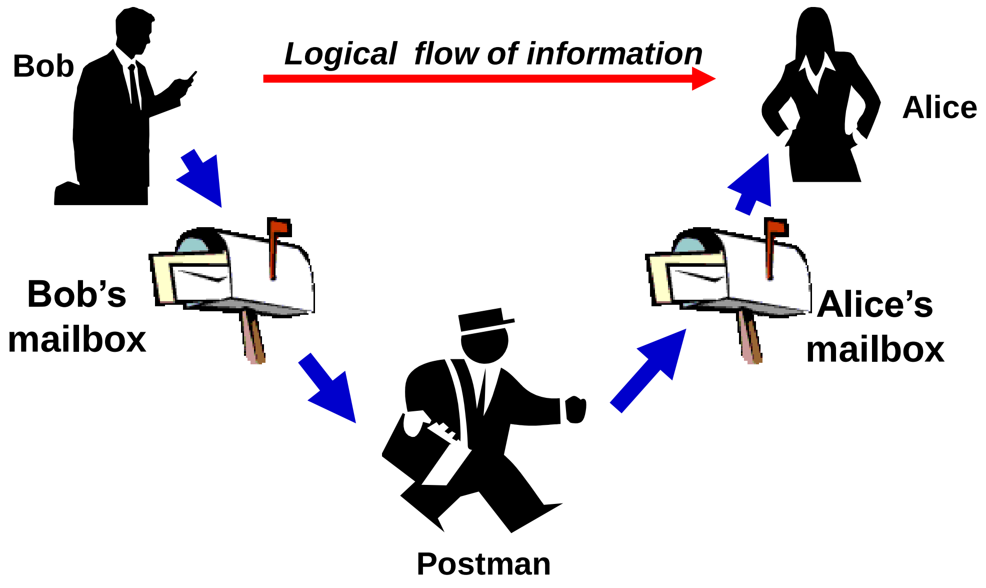
Example: Transport Layer

- Provide logical communication between application processes running on different hosts
- Transport protocols run in end systems
  - Send side:
    - Break application messages into segments
    - Pass to network layer
  - Receive side:
    - Reassemble segments into messages
    - Pass to application layer
- More than one transport protocol available to applications
  - Internet: TCP and UDP

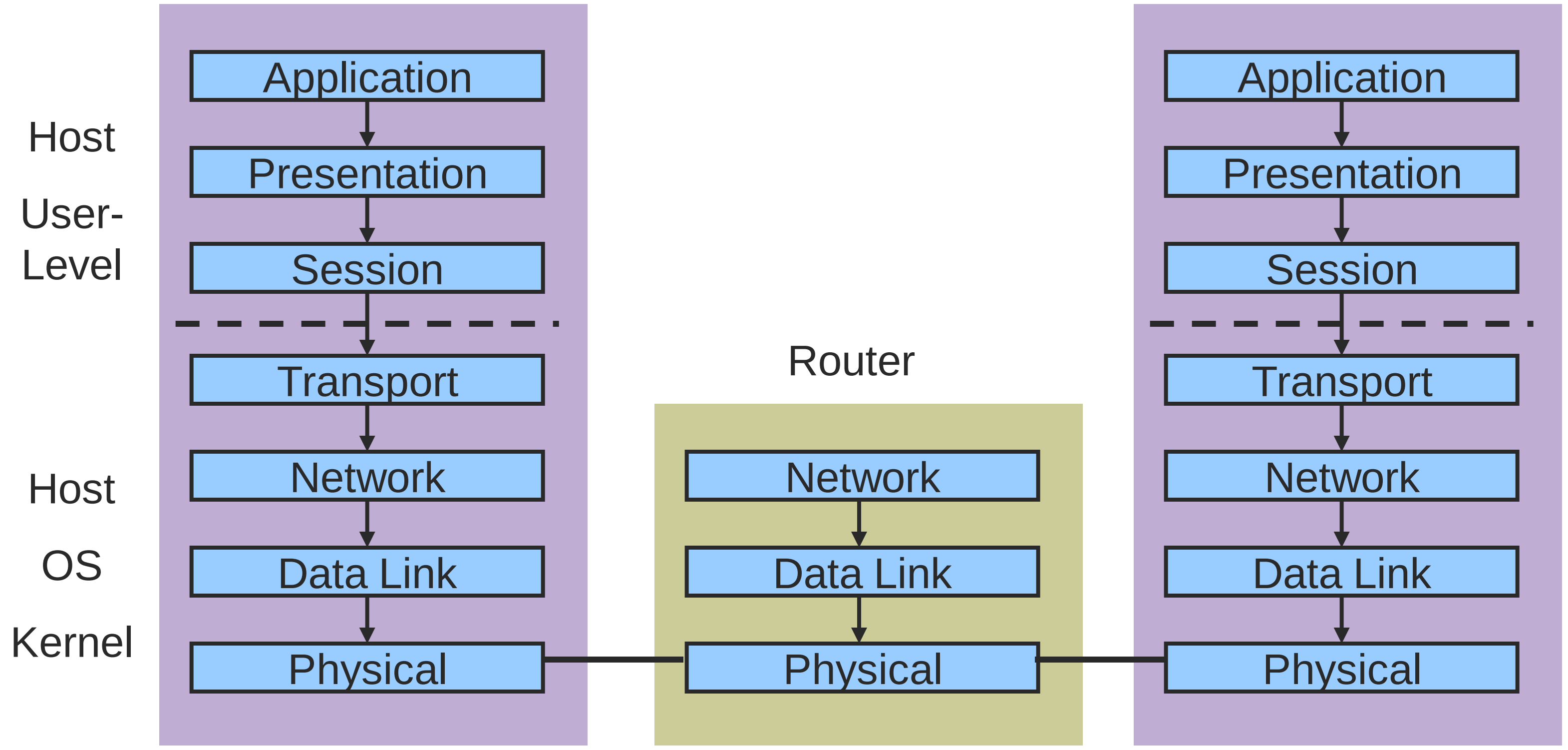


Transport vs. Network Layer

- Transport layer
  - Logical communication between processes
  - Relies on, enhances, network layer services
- Network layer
  - Logical communication between hosts

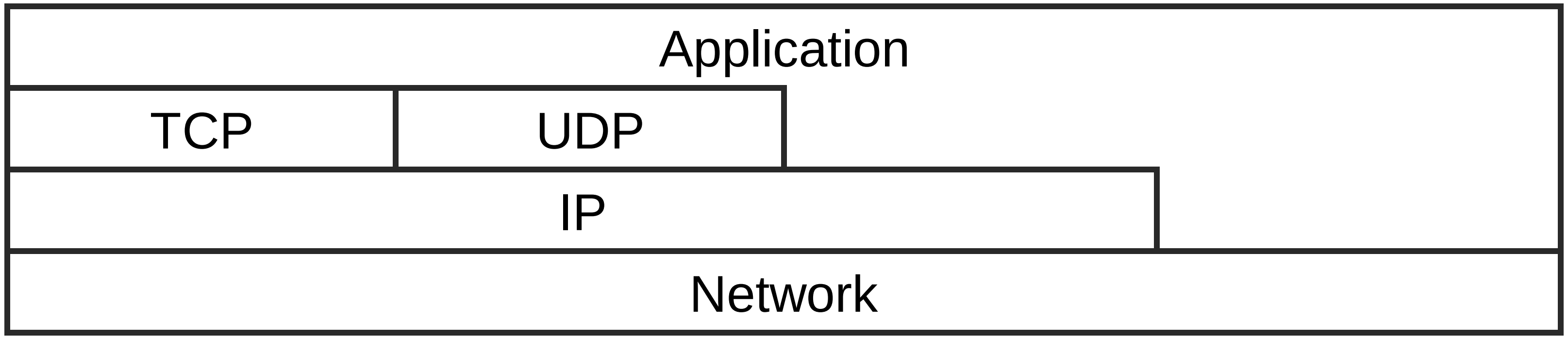


OSI Protocol Stack



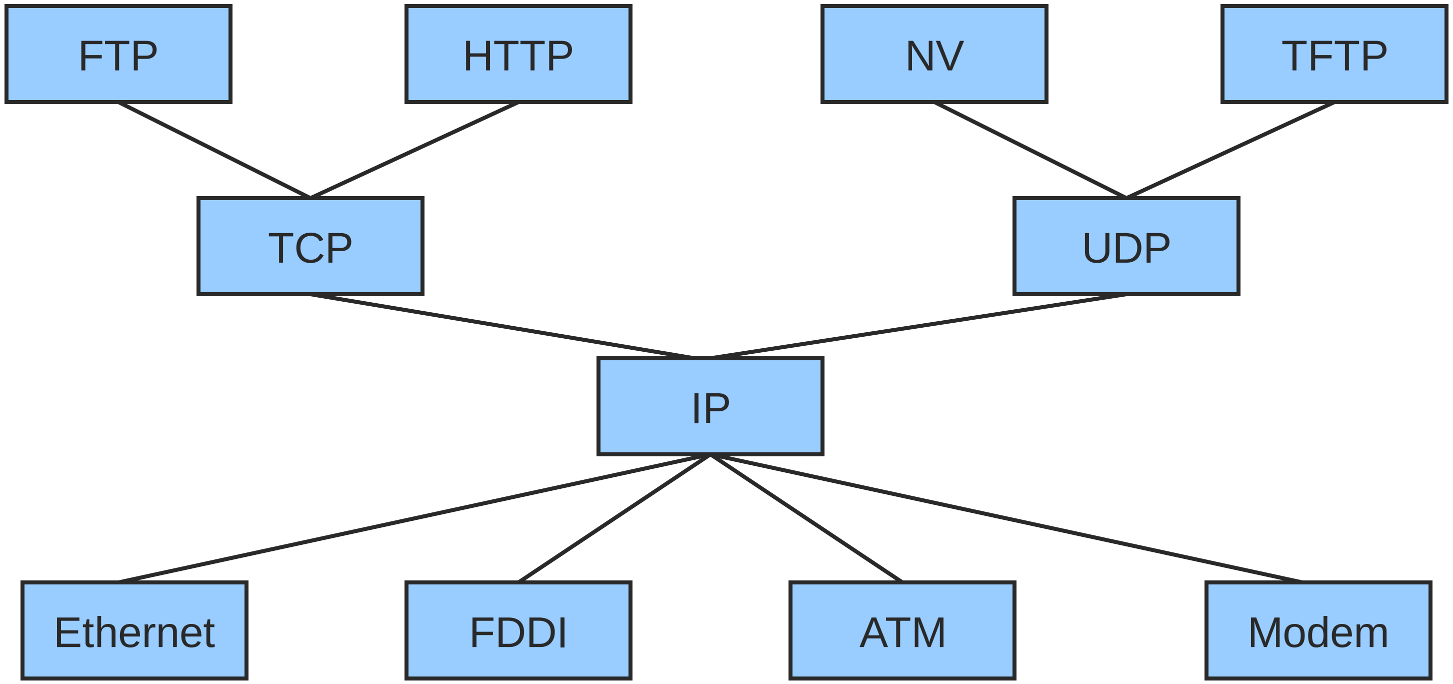
Internet Architecture

- Features
  - No strict layering



Internet Architecture – Hourglass Design

- Features
  - Hourglass shape – IP is the focal point



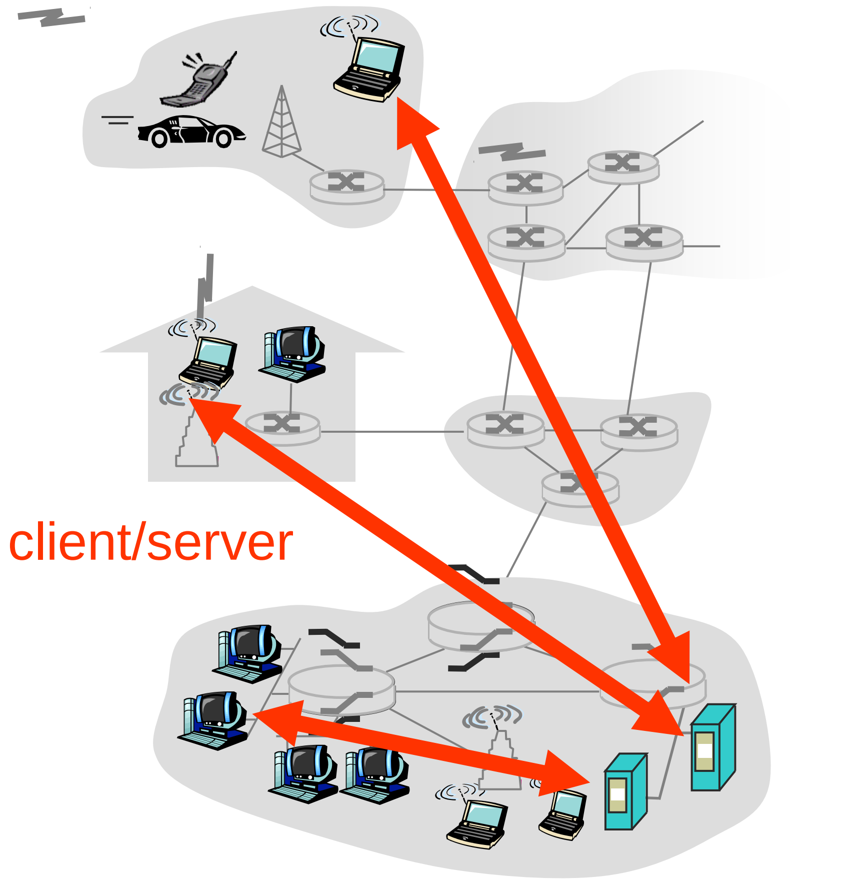
Network Applications

Creating a Network Application

- Write programs that
  - Run on (different) end systems
  - Communicate over network
    - e.g., web server software communicates with browser software
- No need to write software for network-core devices
  - Network-core devices do not run user applications

Client-server Architecture

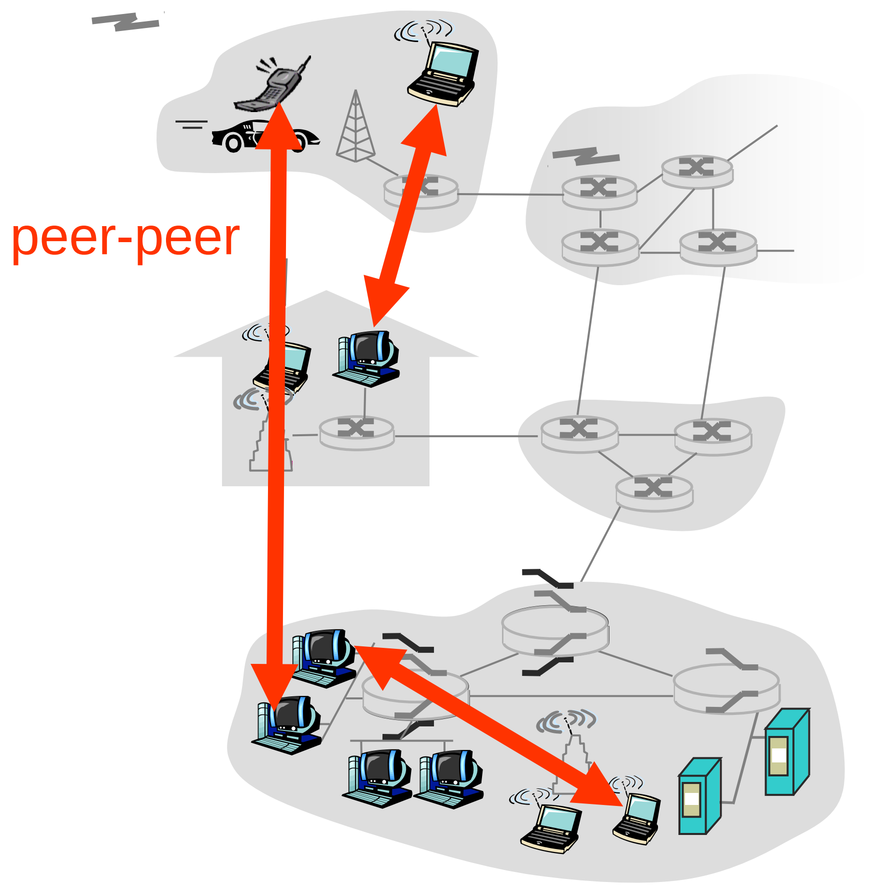
- Server
  - Always-on host
  - Well-known IP address
- Clients
  - Communicate with server
  - May be intermittently connected
  - May have dynamic IP addresses
  - Do not communicate directly with each other



P2P Architecture

- No always-on server
- Arbitrary end systems directly communicate
- Peers are intermittently connected and change IP addresses

- Highly scalable but difficult to manage



Hybrid Client-server and P2P

- Skype
  - Voice-over-IP P2P application
  - Centralized server: finding address of remote party
  - Client-client connection: direct (not through server)
- Instant messaging
  - Chatting between two users is P2P
  - Centralized service: client presence detection/location
  - User registers its IP address with central server when it comes online
  - User contacts central server to find IP addresses of buddies

Communicating Processes

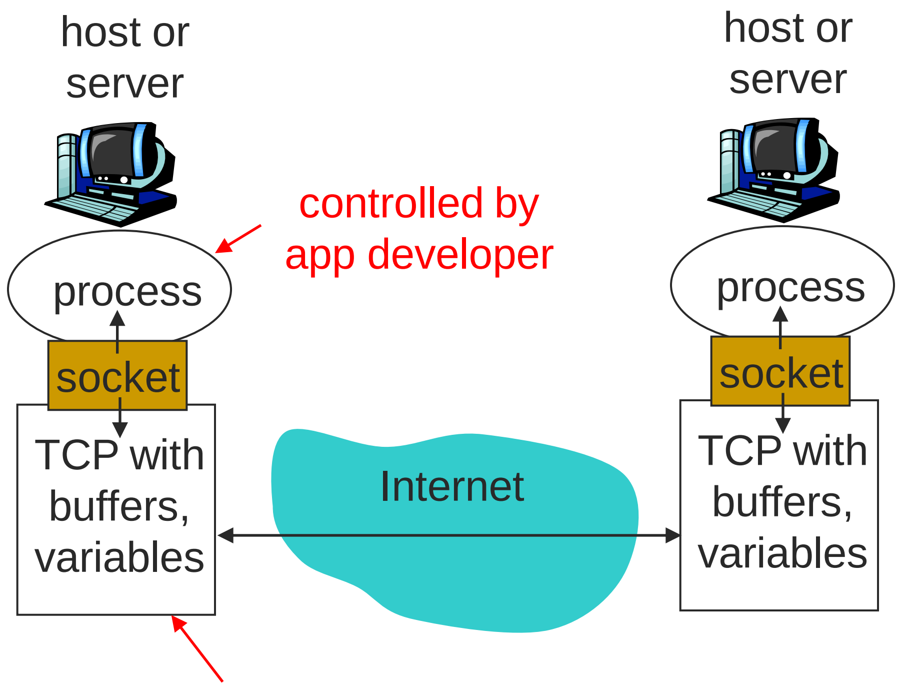
- Process
  - Program running within a host
- Inter-process communication
  - Two processes communicating within same host
- Message Passing
  - Two processes communicating between different hosts
- Client process
  - Initiates communication
- Server process
  - Waits to be contacted

Addressing Processes

- Receiving messages
  - Process must have identifier
  - Host device has unique 32-bit IP address
- Question
  - Does the IP address of host suffice for identifying the process?
  - Answer: No, many processes can be running on same host
- Process Identifier
  - Include both IP address and port number associated with process on host.
- Example port numbers
  - HTTP server: 80
  - Mail server: 25

Sockets

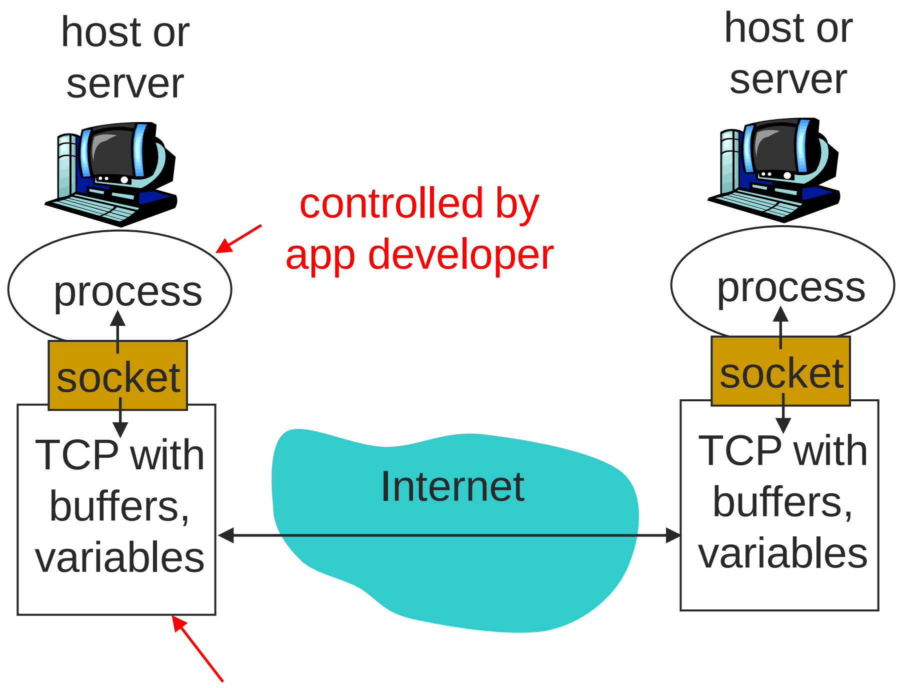
- Process sends/receives messages to/from its socket
  - Analogous to a door
  - Sending process shoves messages out the door
  - Transport infrastructure on other side of door brings message to socket at receiving process



host or server
host or server
controlled by app developer
process
socket
TCP with buffers, variables
Internet
process
socket
TCP with buffers, variables

Sockets

- API
  - Choice of transport protocol
  - Ability to set a few parameters


host or server
host or server
controlled by app developer
process
process
socket
socket
TCP with buffers, variables
TCP with buffers, variables
Internet

Transport Services

- Data loss
  - Some applications (e.g., audio) can tolerate some loss
  - Other apps (e.g., file transfer, telnet) require 100% reliability
- Timing
  - Some applications (e.g., IP phones, interactive games) require low delay to be "effective"
- Throughput
  - Some applications (e.g., multimedia) have a minimum throughput to be "effective"
  - other applications ("elastic apps") make use of whatever throughput they get
- Security
  - Encryption, data integrity, …

Internet Transport Protocols

TCP

- Connection-oriented
  - setup required between client and server
- Reliable transport
- Flow control
  - Won’t overwhelm receiver
- Congestion control
  - Won’t overwhelm network
- Does not provide
  - Timing, throughput guarantees, security

UDP

- Unreliable data transfer
- Does not provide
  - Connection setup, reliability, flow control, congestion control, timing, throughput guarantee, or security
- Question
  - Why bother? Why is there a UDP?