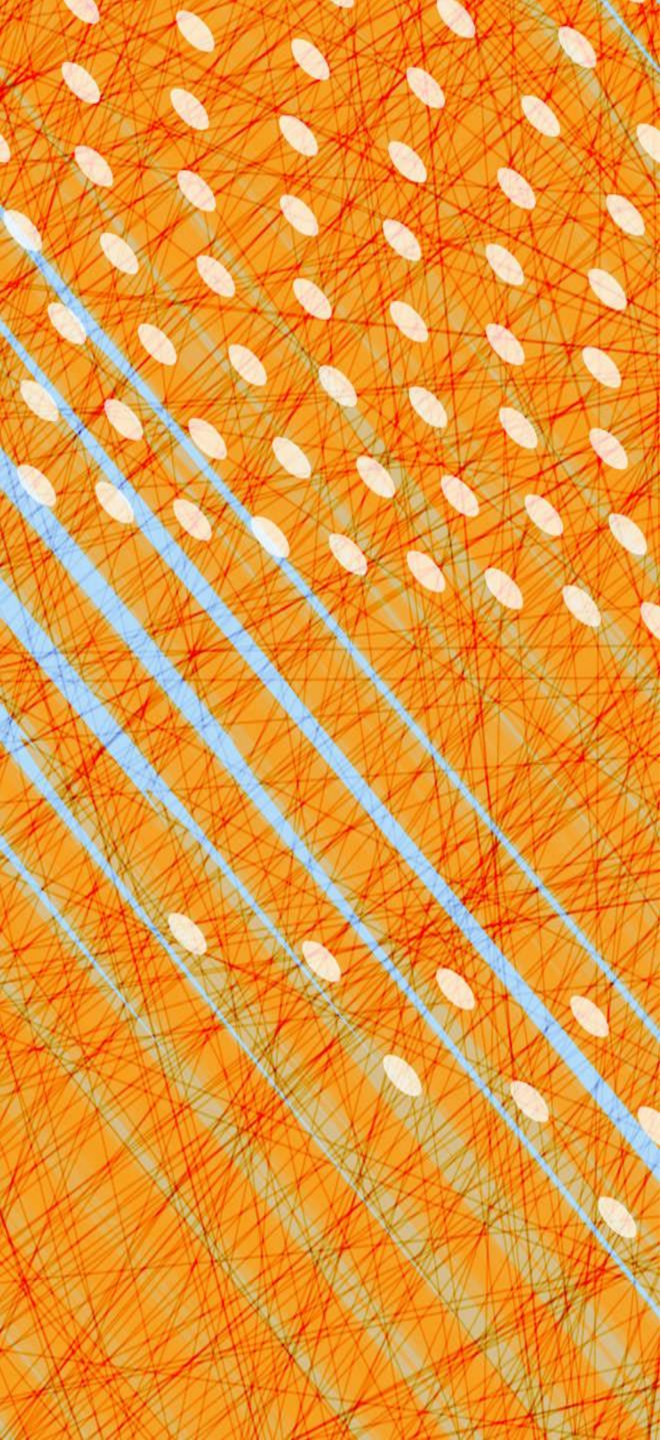
Networking

CS 4410
Operating Systems

Cornell CIS
COMPUTING AND INFORMATION SCIENCE

[R. Agarwal, L. Alvisi, A. Bracy, M. George, Kurose, Ross, E. Sirer, R. Van Renesse]



Introduction
Application Layer
Transport Layer
Link Layer
Network Layer
Routing

Basic Network Abstraction

- A process can create “endpoints”
- Each endpoint has a unique address
- A message is a byte array
- Processes can:
  - receive messages on endpoints
  - send messages to endpoints

Network “protocol”

Agreement between processes about the content of messages

- **Syntax:** Layout of bits, bytes, fields, etc.
  - message format
- **Semantics:** what fields, messages mean

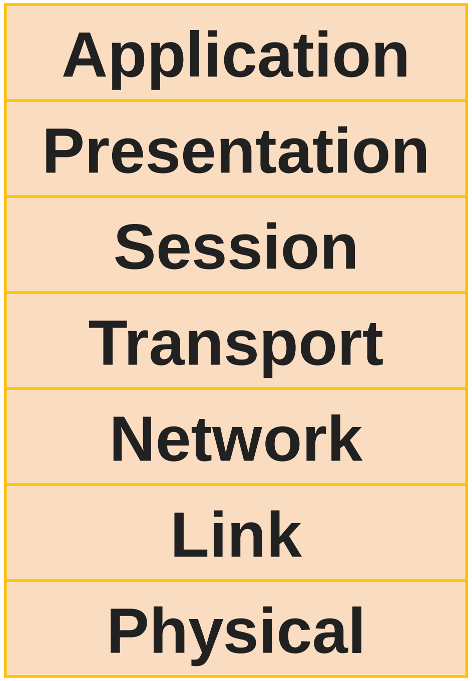
Example:

- HTTP “get” requests and responses

Network Layering

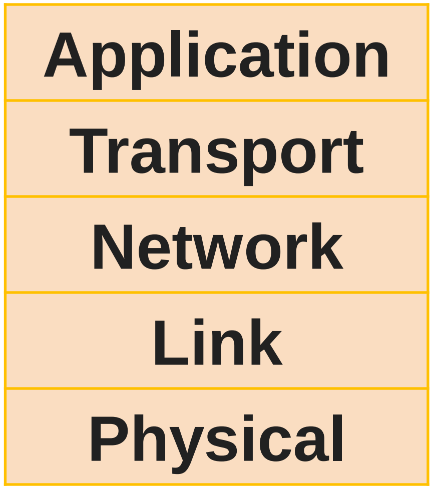
Network abstraction is usually *layered*

- Like Object Oriented-style inheritance
- Also like the hw/sw stack



Application
Presentation
Session
Transport
Network
Link
Physical

Proposed 7-Layer ISO/OSI reference model (1970's)



Application
Transport
Network
Link
Physical

Actual 5-Layer Internet Protocol Stack

OSI Layers

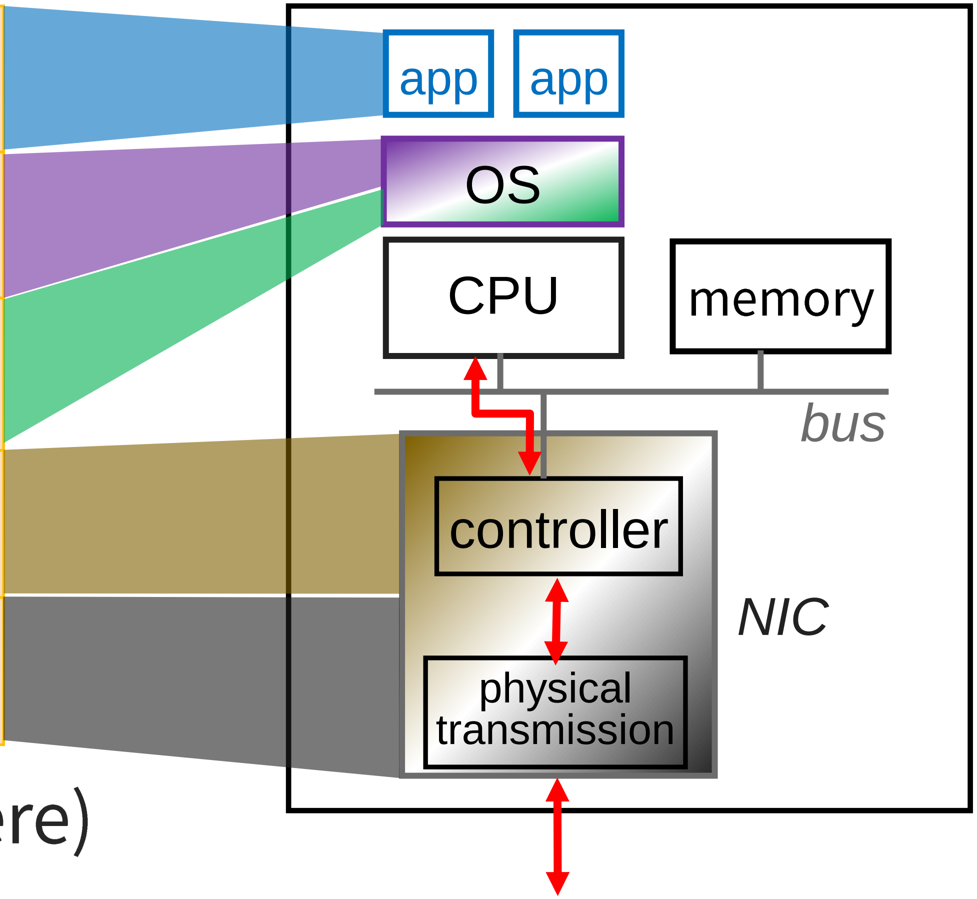
Application	Network-aware applications, clients & servers
Presentation	Translation between network and application formats (*e.g.*, RPC packages, sockets)
Session	Connection management
Transport	Data transfer, reliability, packetization, retransmission. Lets multiple apps share 1 network connection
Network	Path determination across multiple network segments, routing, logical addressing.
Link	Decides whose turn it is to talk, finds physical device on network.
Physical	Exchanges bits on the media (electrical, optical, *etc.*)

Internet Protocol Stack

Application	exchanges **messages**	HTTP, FTP, DNS
Transport	Transports messages; exchanges **segments**	TCP, UDP
Network	Transports segments; exchanges **datagrams**	IP, ICMP (ping)
Link	Transports datagrams; exchanges **frames**	Ethernet, WiFi
Physical	Transports frames; exchanges **bits**	wires, signal encoding

Who does what?

Application	HTTP, FTP, DNS (*these*^ are usually in libraries)
Transport	TCP, UDP
Network	IP, ICMP (ping)
Link	Ethernet, WiFi
Physical	wires, signal encoding



(Hard to draw firm lines here)

- Each host has 1+ Network Interface Cards (NIC)
- Attaches into host’s system buses
- Combination of hardware, software, firmware

Layers support **Modularity**

Each layer:

- relies on services from layer below
- exports services to layer above

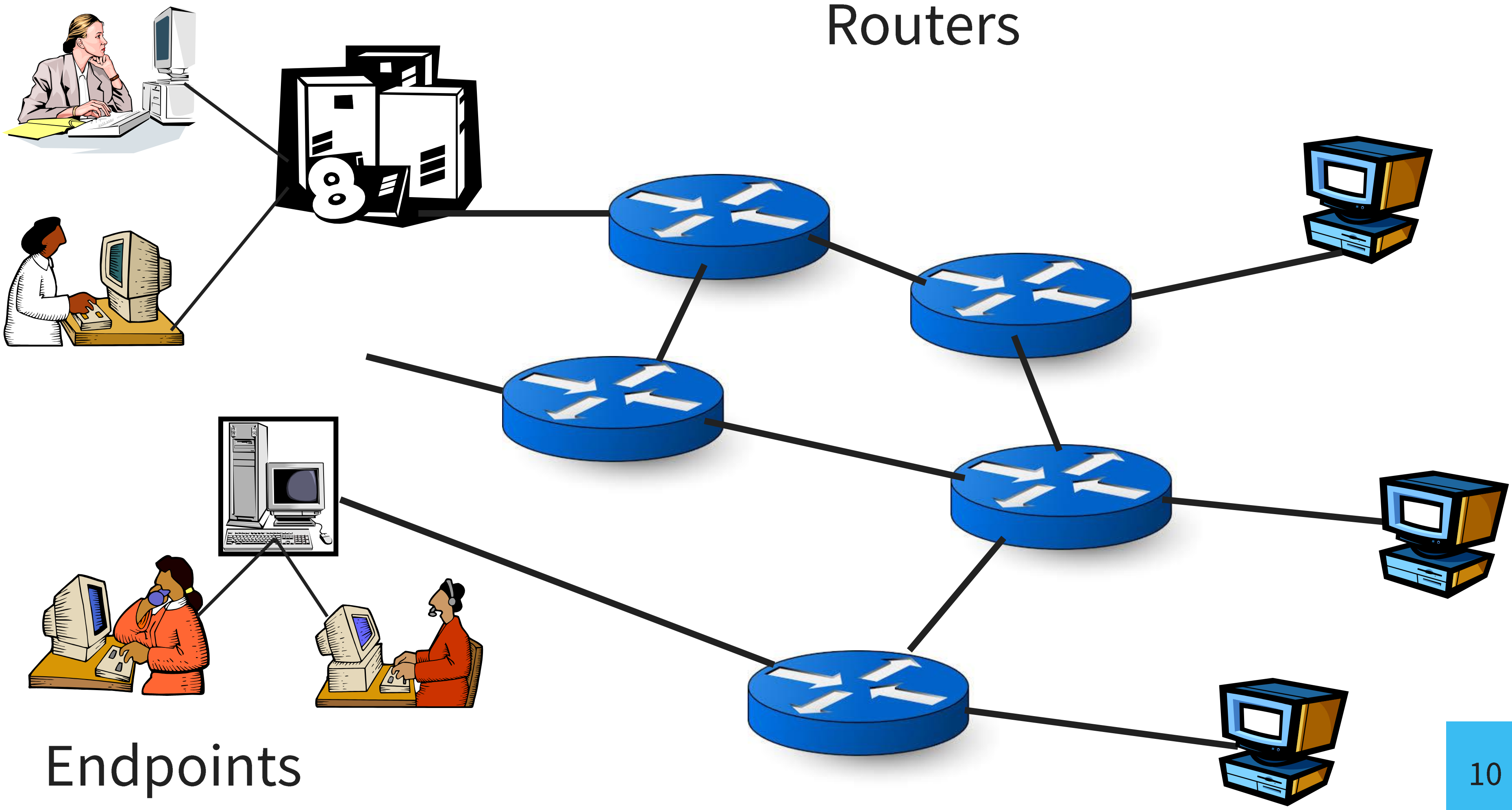
Can identify the relationship between distinct pieces of complex system.

Interfaces between layers:

- Hide implementation details
- Ease maintenance, updates
  - change of implementation of layer’s service transparent to rest of system

Internet, The Big Picture

How about an analogy?

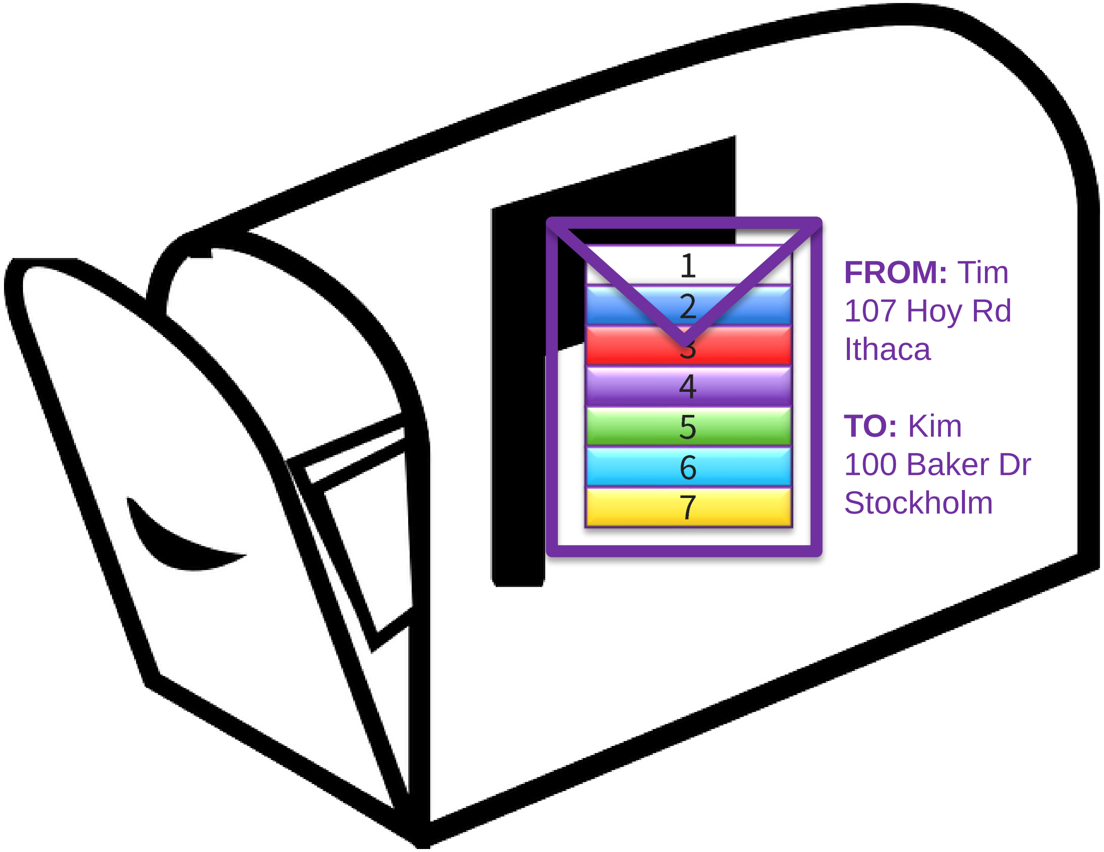


Routers

Endpoints

Application Layer

Tim → Kim



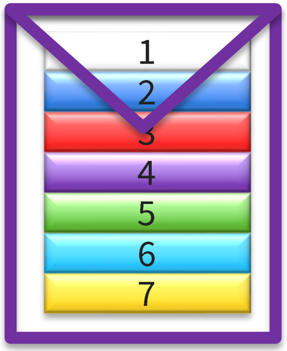
FROM: Tim
107 Hoy Rd
Ithaca

TO: Kim
100 Baker Dr
Stockholm

Analogy by Shubham Dubey on Quora

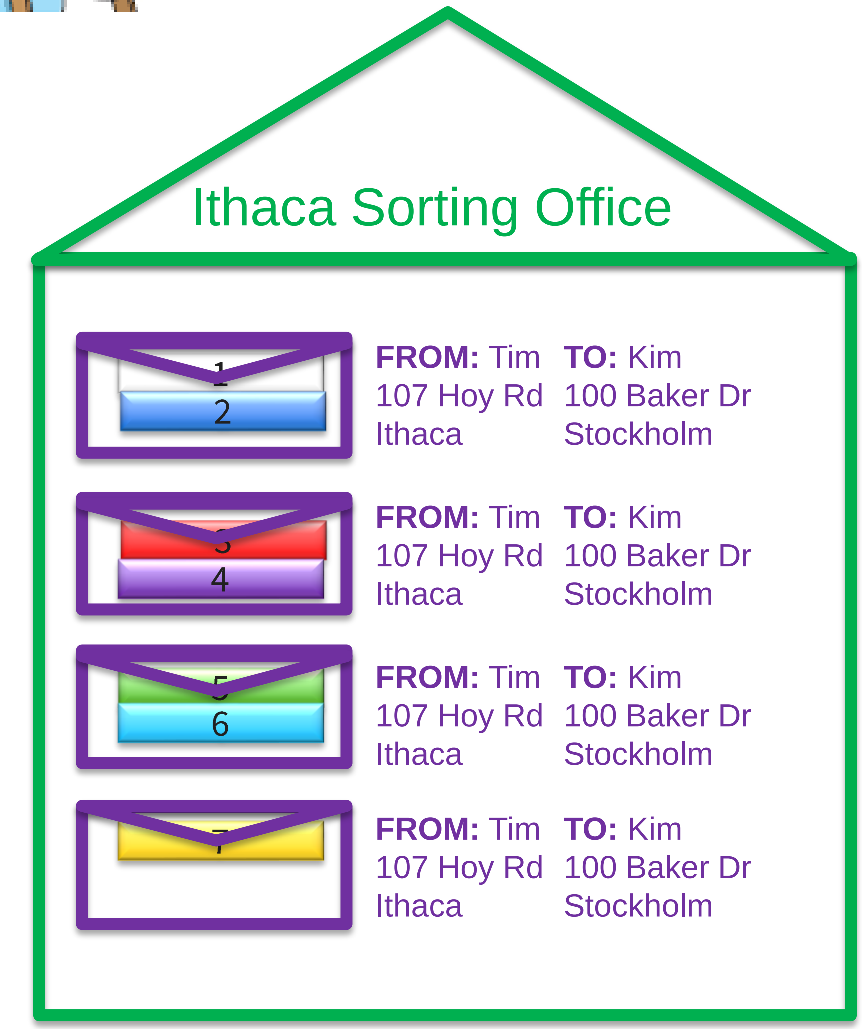
Transport Layer

Ithaca Postman → Stockholm Postman



FROM: Tim
107 Hoy Rd
Ithaca

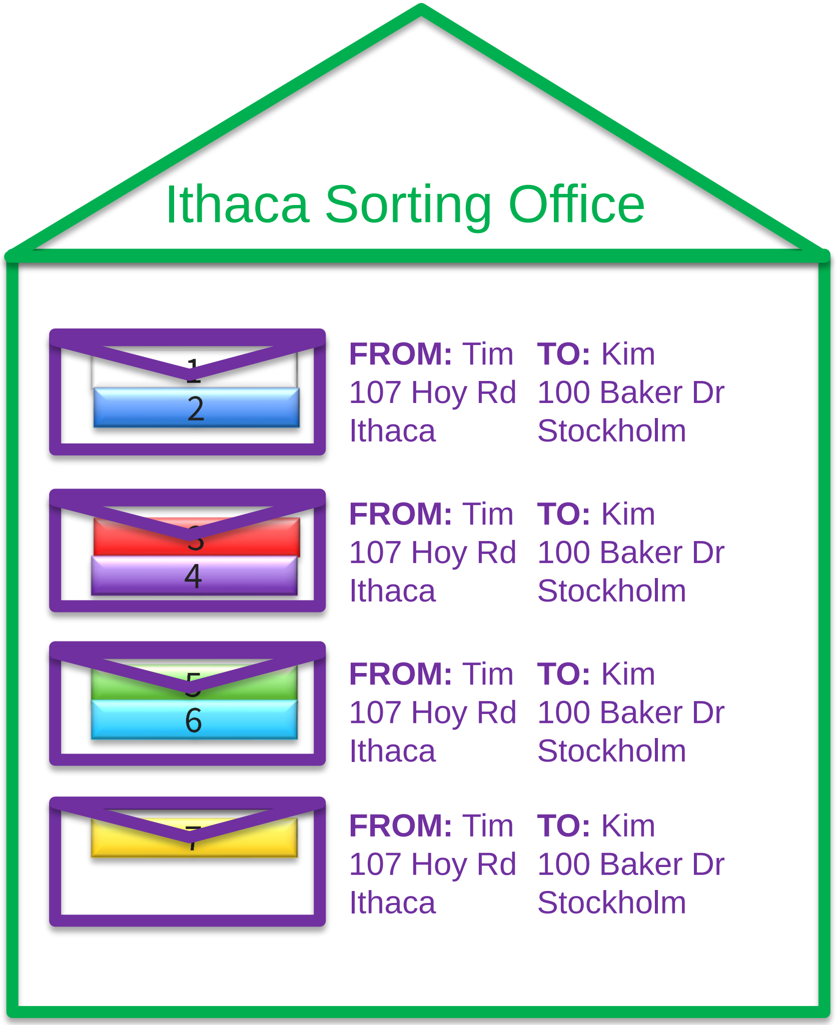
TO: Kim
100 Baker Dr
Stockholm



FROM: Tim 107 Hoy Rd Ithaca	**TO:** Kim 100 Baker Dr Stockholm
FROM: Tim 107 Hoy Rd Ithaca	**TO:** Kim 100 Baker Dr Stockholm
FROM: Tim 107 Hoy Rd Ithaca	**TO:** Kim 100 Baker Dr Stockholm
FROM: Tim 107 Hoy Rd Ithaca	**TO:** Kim 100 Baker Dr Stockholm

Network Layer

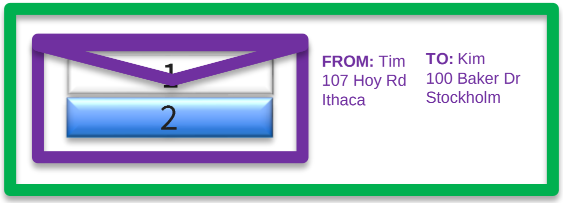
Ithaca Sorting Office → Stockholm Sorting Office



Ithaca Sorting Office

FROM: Tim 107 Hoy Rd Ithaca	**TO:** Kim 100 Baker Dr Stockholm
FROM: Tim 107 Hoy Rd Ithaca	**TO:** Kim 100 Baker Dr Stockholm
FROM: Tim 107 Hoy Rd Ithaca	**TO:** Kim 100 Baker Dr Stockholm
FROM: Tim 107 Hoy Rd Ithaca	**TO:** Kim 100 Baker Dr Stockholm

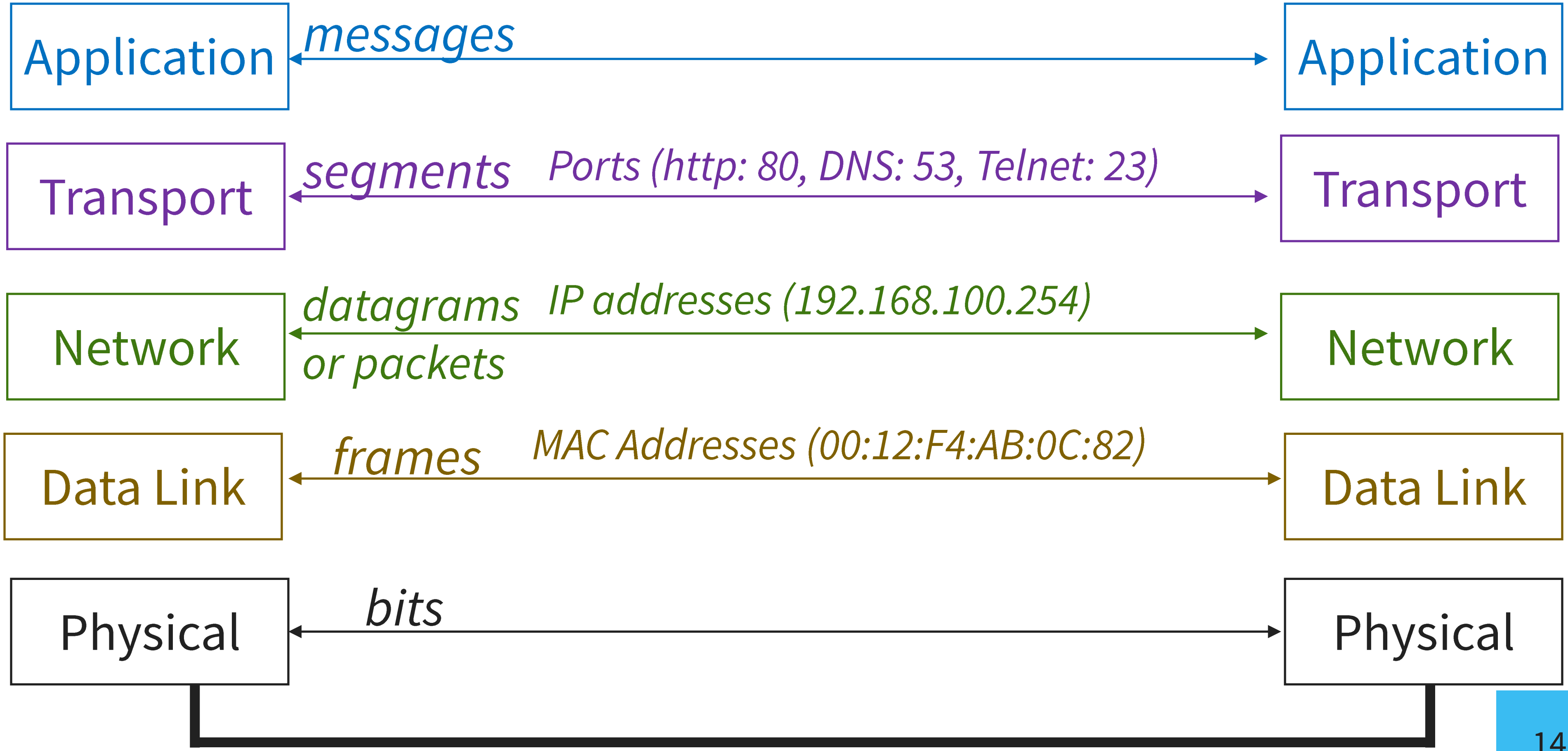
TO: Stockholm Sorting Office



FROM: Tim
107 Hoy Rd
Ithaca

TO: Kim
100 Baker Dr
Stockholm

The Big Picture



Application *messages* Application

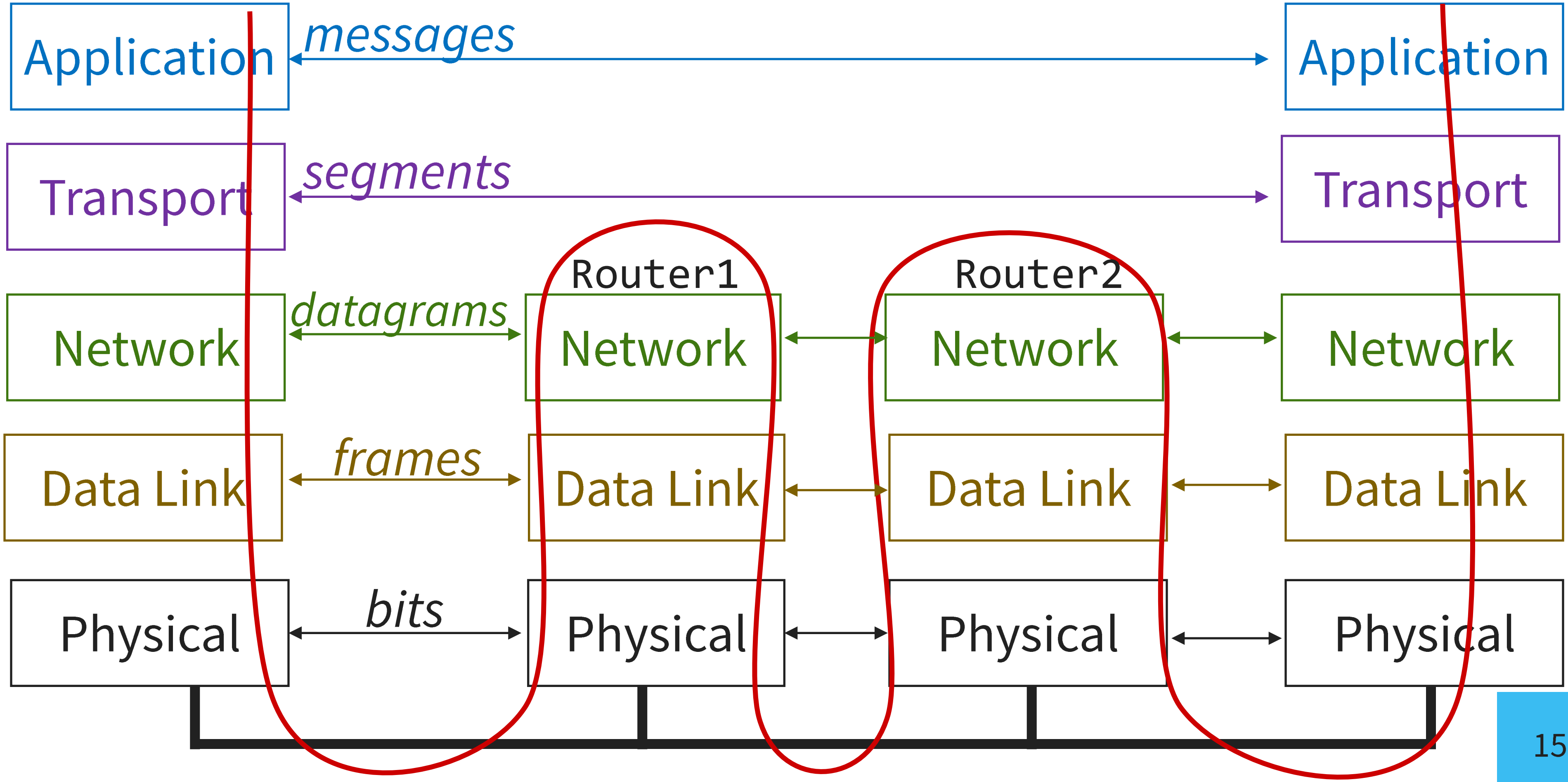
Transport *segments* *Ports (http: 80, DNS: 53, Telnet: 23)* Transport

Network *datagrams* *or packets* *IP addresses (192.168.100.254)* Network

Data Link *frames* *MAC Addresses (00:12:F4:AB:0C:82)* Data Link

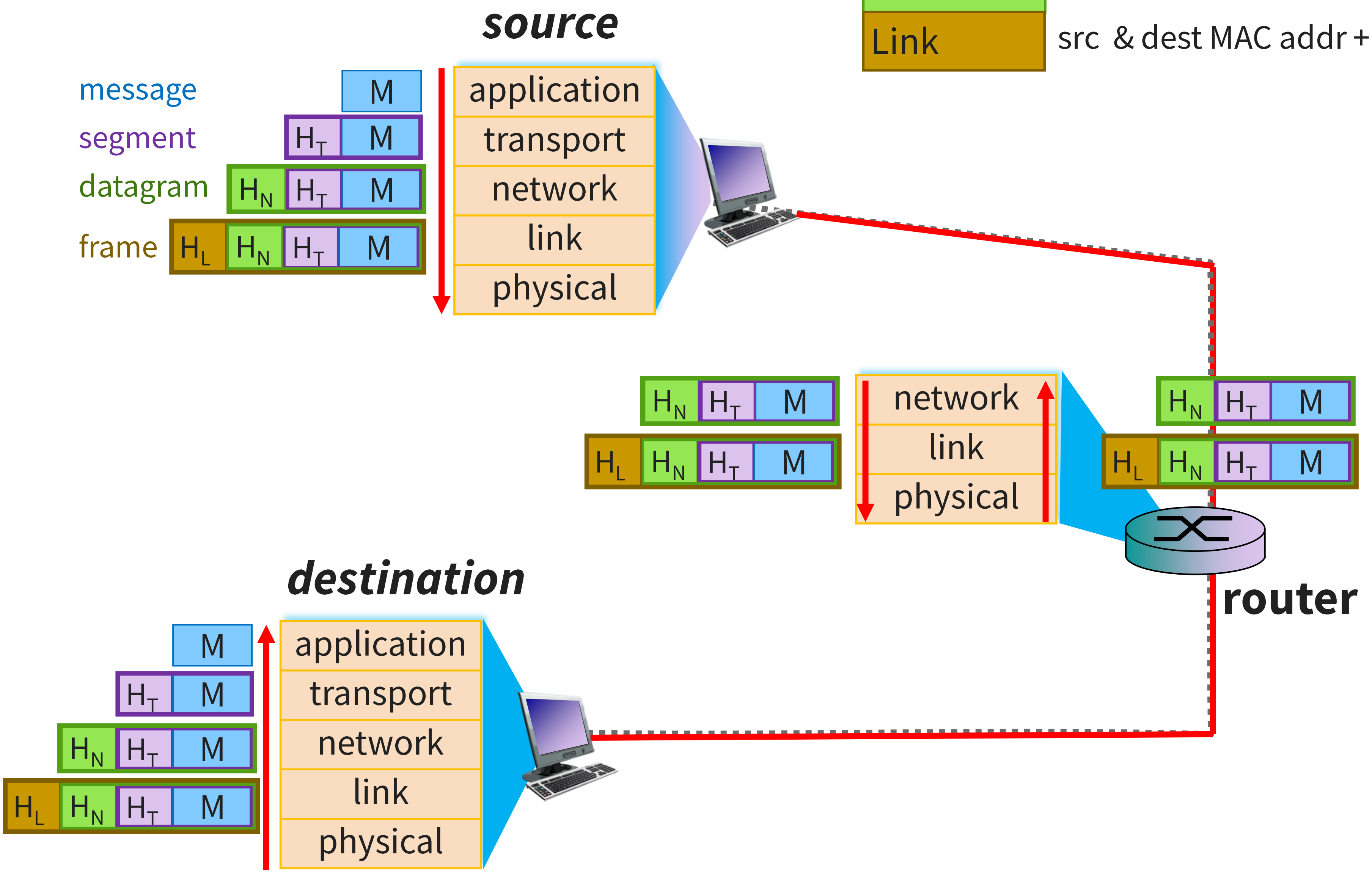
Physical *bits* Physical

The Big Picture



Encapsulation

Headers	
Transport	src & dst ports + …
Network	src & dest IP addr + …
Link	src & dest MAC addr + …



End-to-End Argument

- Occam's Razor for Internet architecture
- Application-specific properties are best provided by the applications, **not the network**
  - Guaranteed, or ordered, packet delivery, duplicate suppression, security, *etc.*

- Internet performs the simplest packet routing and delivery service it can
  - Packets are sent on a best-effort basis
  - Higher-level applications do the rest

End-to-End Example

Should the network guarantee packet delivery?

Consider: a file transfer program (read file from disk, send it, receiver reads packets & writes them to disk)

- **Q:** If network guarantees delivery, wouldn't applications be simpler? (no retransmissions!)
- **A:** no, still need to check that file was written to remote disk intact

A check is necessary if nodes can fail.

→ Applications need to be written to perform their own retransmits

Why burden the network with properties that can, and must, be implemented at the periphery?

The Missing Layers

Presentation

translation between network & application formats (e.g., RPC packages, sockets). Allows communicating applications to interpret the meaning of data exchanged:

- data conversion
- character code translation
- compression
- encryption

Session

synchronization of data exchange:

- supports checkpointing and recovery schemes
- establish, manage, and tear down connections

Need these services?
Put them in your application.