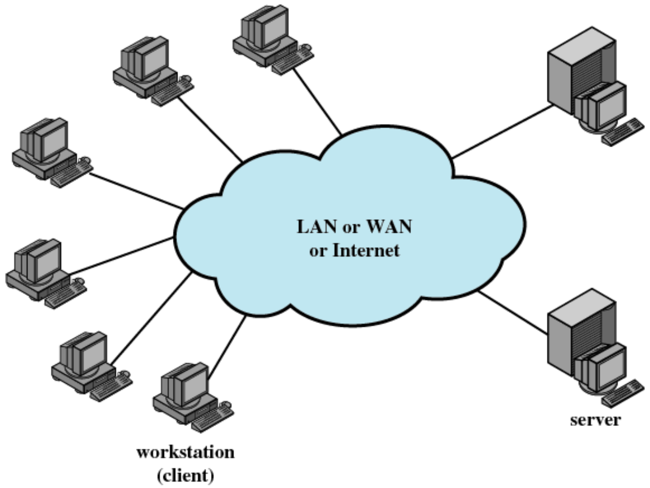
TI III: Operating Systems & Computer Networks
Networked Computer & Internet

Prof. Dr.-Ing. Jochen Schiller
Computer Systems & Telematics
Freie Universität Berlin, Germany



Content (2)

8. **Networked Computer & Internet**
   - Sockets
   - Internet
   - Layers
   - Protocols
9. Host-to-Network
   - Physical Layer, Signals, Modems
   - Data Link Layer, Framing, Flow Control, Error Detection/Correction
   - Topologies, Medium Access
   - Local Area Networks, Ethernet
10. Internetworking
    - Switches, Routers
    - Routing
    - Internet Protocol
    - Addressing
11. Transport Layer
    - Protocol Mechanisms
    - TCP, UDP
    - Addressing, Ports
12. Applications
    - Domain Name System
    - Email
    - World Wide Web
13. Network Security
    - Basic Concepts & Terms
    - Cryptology
    - Firewalls, VPNs, IP Security, Email Security
14. Example
    - Under the Hood of Surfing the Web

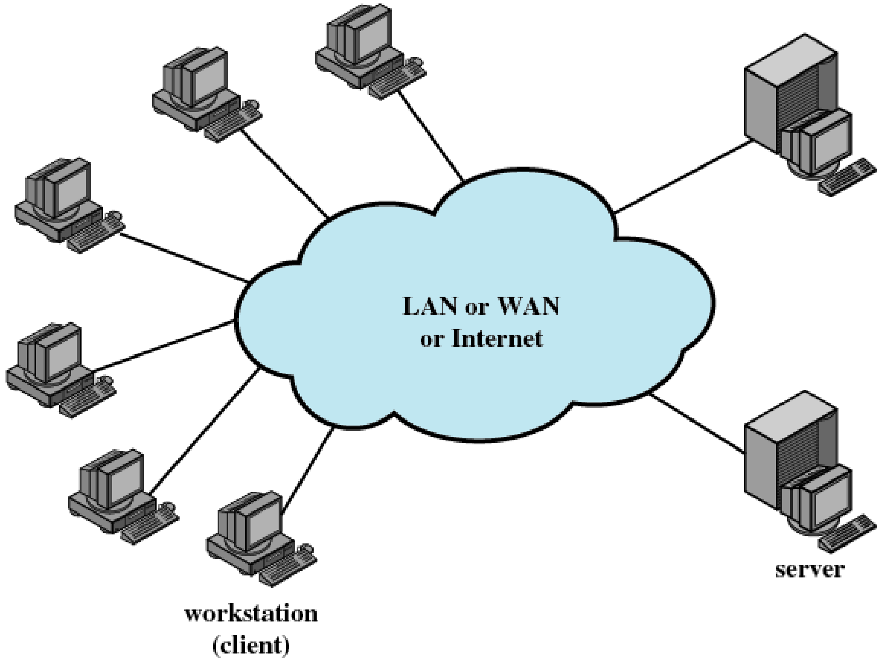
Motivation – Networked Computers

Questions:

How can a user/process communicate over the network?

How can (possibly distant) computers exchange data?

How does a computer know which other computer it should be talking to?



Motivation – Networked Computers

www.mi.fu-berlin.de

160.45.117.199

Socket

- Enable communication between a client and server
- Concatenation of a Port and an IP address form a socket, 160.45.117.199:80 (http://www.mi.fu-berlin.de)

OS Support for Networking

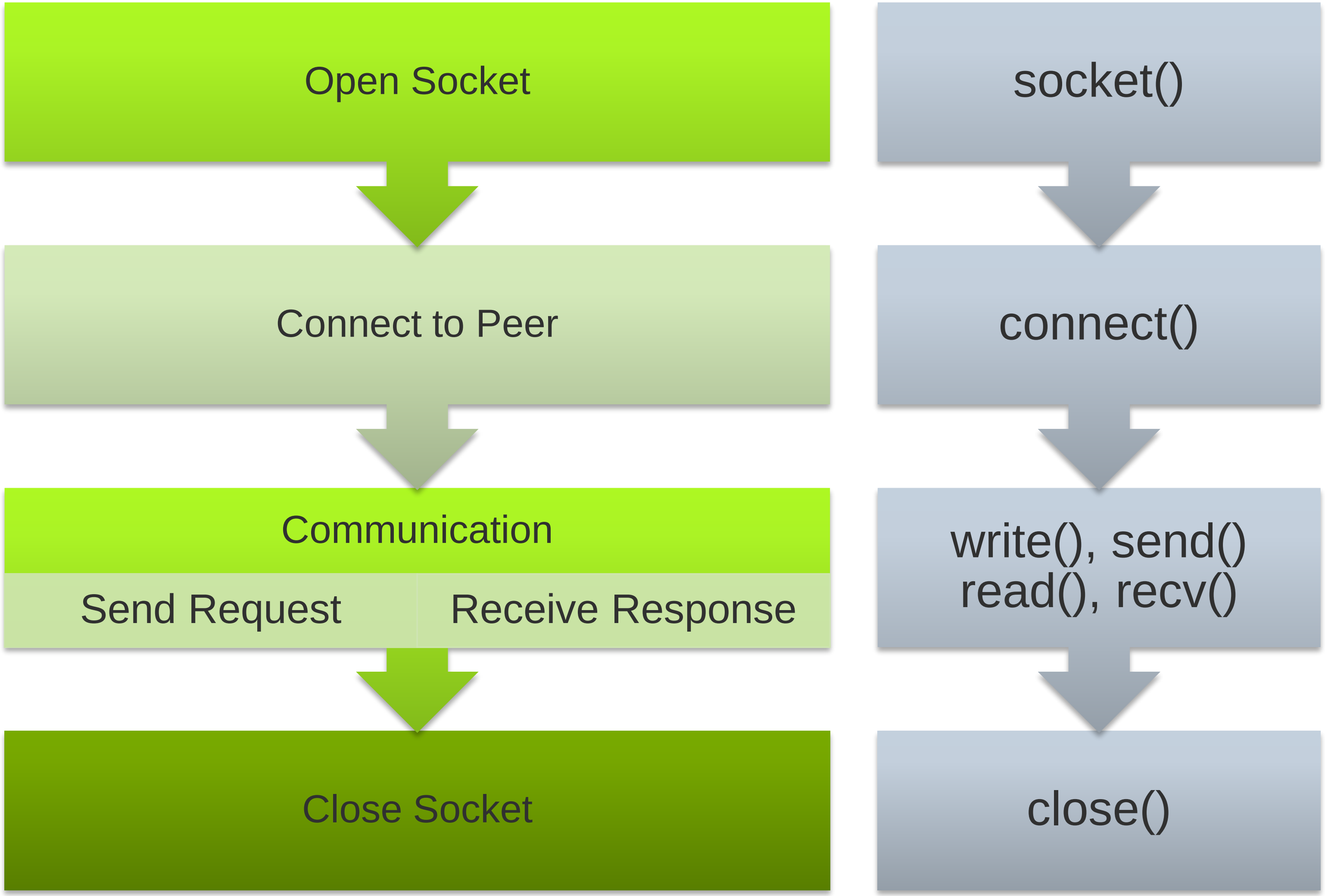
Types of Sockets (classical Internet)

Stream sockets
- Use Transmission Control Protocol (TCP)
- Reliable data transfer

Datagram sockets
- Use User Datagram Protocol (UDP)
- Delivery is not guaranteed

➢ Processes may open sockets to transparently communicate with processes on remote computers

OS Support for Networking



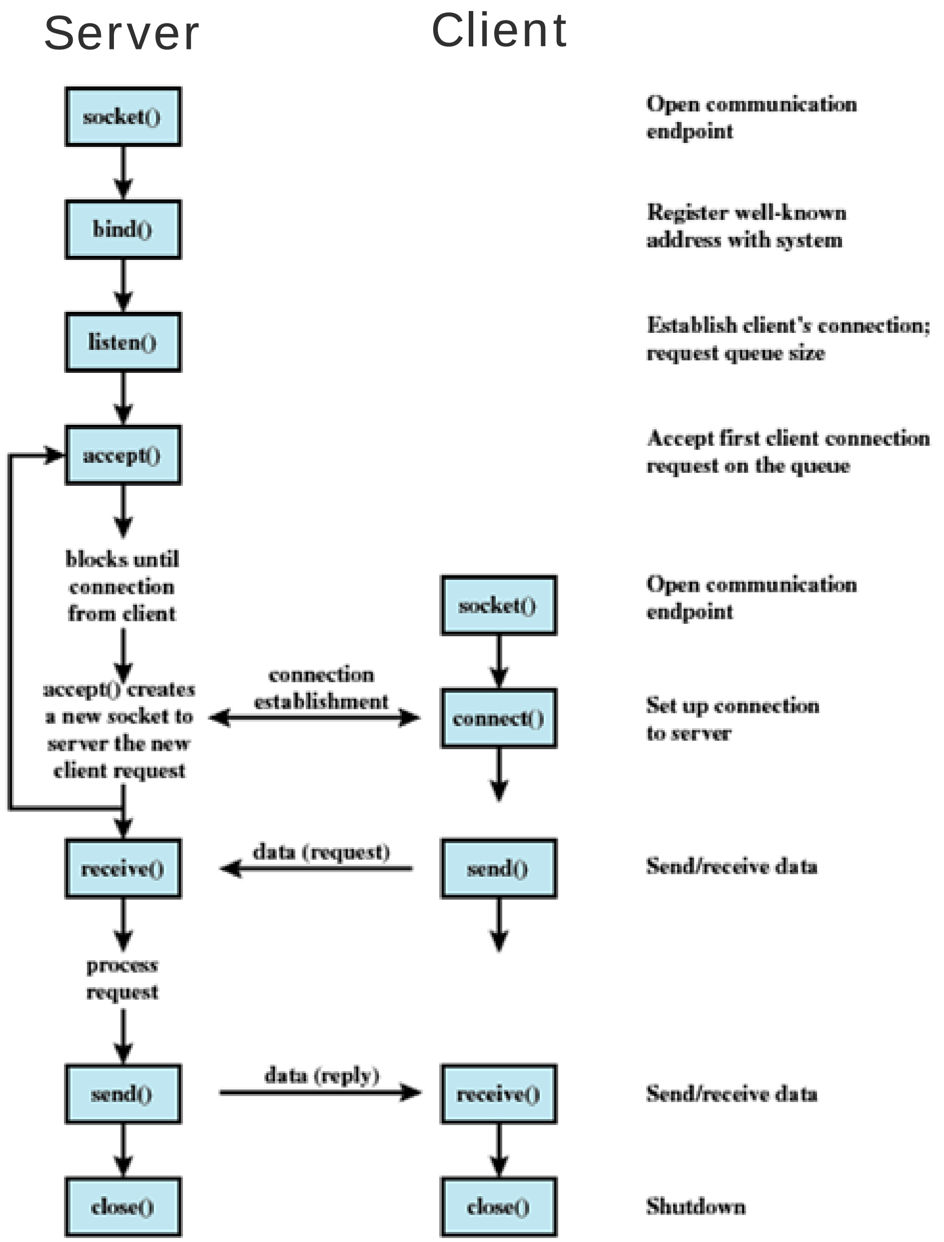
Socket Creation and Operation

System call

```
int socket(int domain, int type, int protocol)
```

Parameters

- **`domain`** Protocol family
  - ➢e.g. **`PF_INET`** for TCP/IP
- **`type`**
  - ➢Stream or datagram
- **`protocol`** (optional)
  - ➢e.g. TCP or UDP (for TCP/IP networking)



Datagram Communication

Simplest possible service: unreliable datagrams

Sender

1. `int s = socket(…);`
2. ```
   sendto(s,
       buffer,
       datasize,
       0,
       to_addr,
       addr_length);
   ```

- `to_addr` and `addr_length` specify destination

Receiver

1. `int s = socket(…);`
2. `bind(s, local_addr, …);`
3. ```
   recv(s,
       buffer,
       max_buff_length,
       0);
   ```

- Will wait until data is available on socket `s` and put the data into `buffer`

Byte Streams over Connection-Oriented Socket

For reliable byte streams, sockets have to be connected first

Receiver has to accept connection

Client

1. `int s = socket(…);`
2. `connect(s, destination_addr, addr_length);`
3. `send(s,buffer, datasize, 0);`
4. Arbitrary `recv()/send()`
5. `close (s);`

- Connected sockets use a `send` without address information

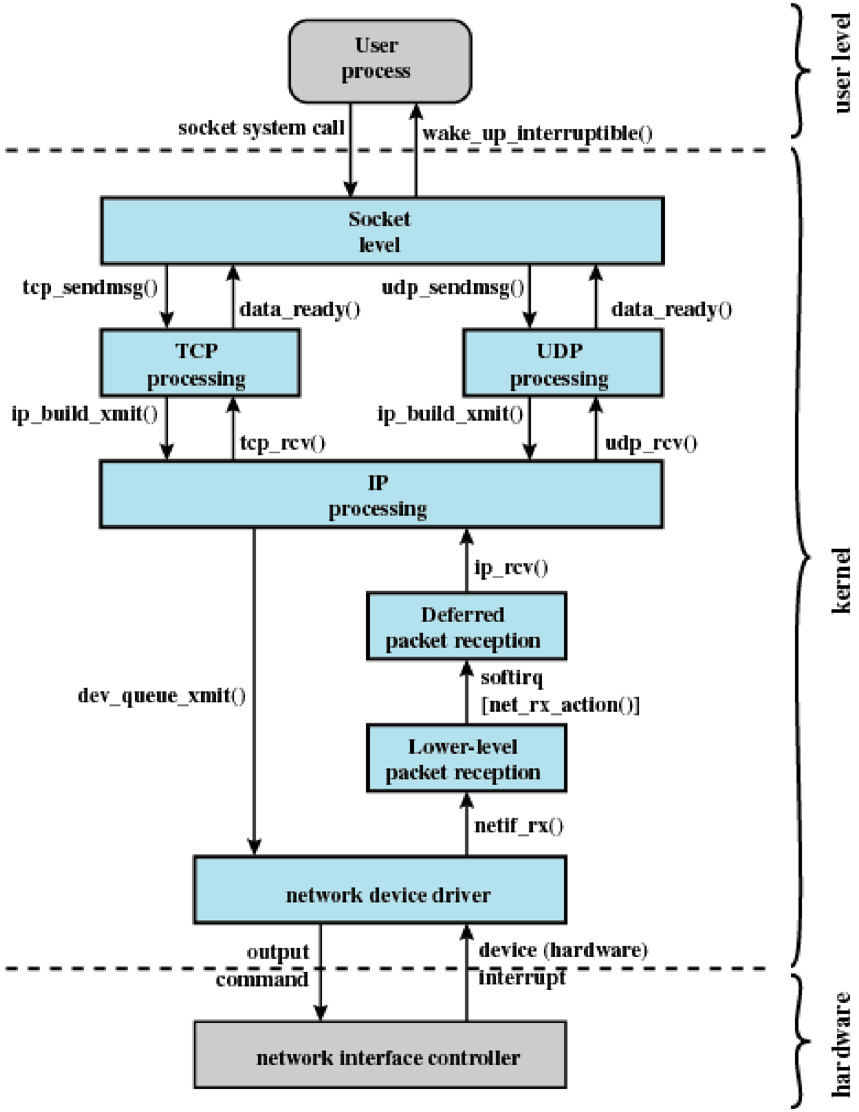
Server

1. `int s = socket(…);`
2. `bind(s, local_addr, …);`
3. `listen(s, …);`
4. `int newsock = accept(s, *remote_addr, …);`
5. `recv(newsock, buffer, max_buff_length, 0);`
6. Arbitrary `recv()/send()`
7. `close (newsock);`

   `...`

8. `close(s);`

Kernel-level Socket Support



API to user space

TCP / UDP implementation

Internet Protocol (IP) implementation

Device driver (includes method to access communication medium)

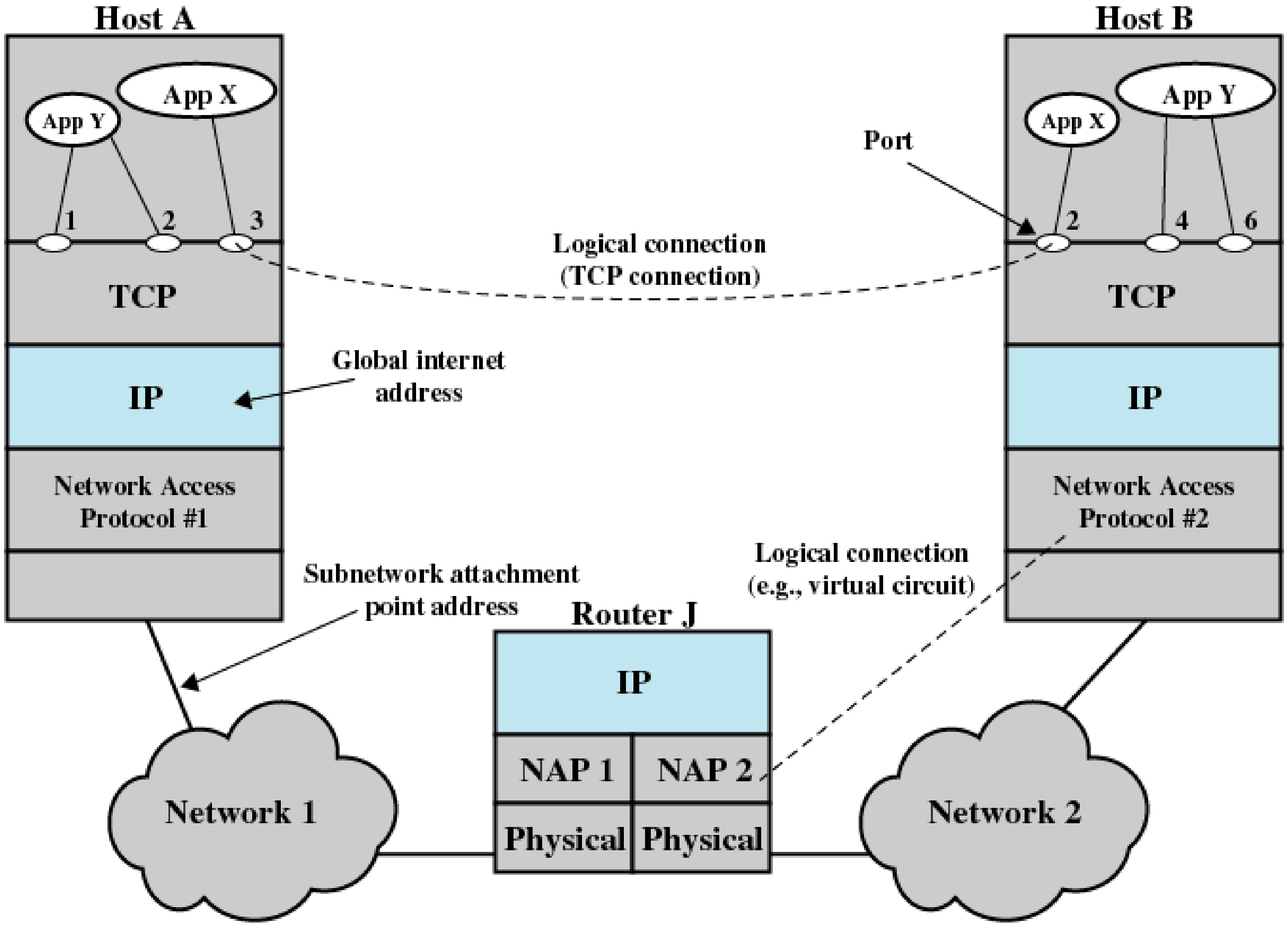
Layered communication architecture

Questions & Tasks

- What is the difference between connection oriented (streams) and connection-less (datagrams) services?
- Which protocols support these services?
- Where is the distance from sender to receiver (local, same city/country, global) reflected?
- How to address a computer and a process?

THE INTERNET

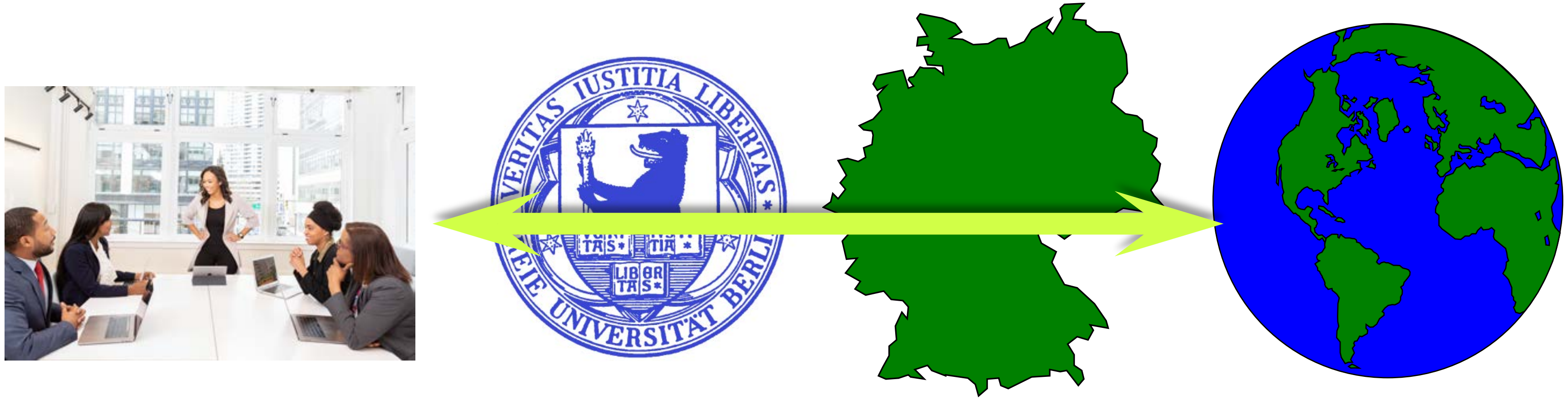
Internet / TCP/IP Network Stack



The Internet

The Internet consists of

- many computers
  - using same network protocol family TCP/IP
    - IP on top of lower-level protocol (Ethernet, WLAN, Bluetooth, ...)
  - that are (directly or indirectly) connected to each other
  - that offer or use certain services
- many users that have direct access to the services
- many networks interconnected via gateways



Computer Science Dept. FU Berlin Germany World

Structure of the Internet (Concept)

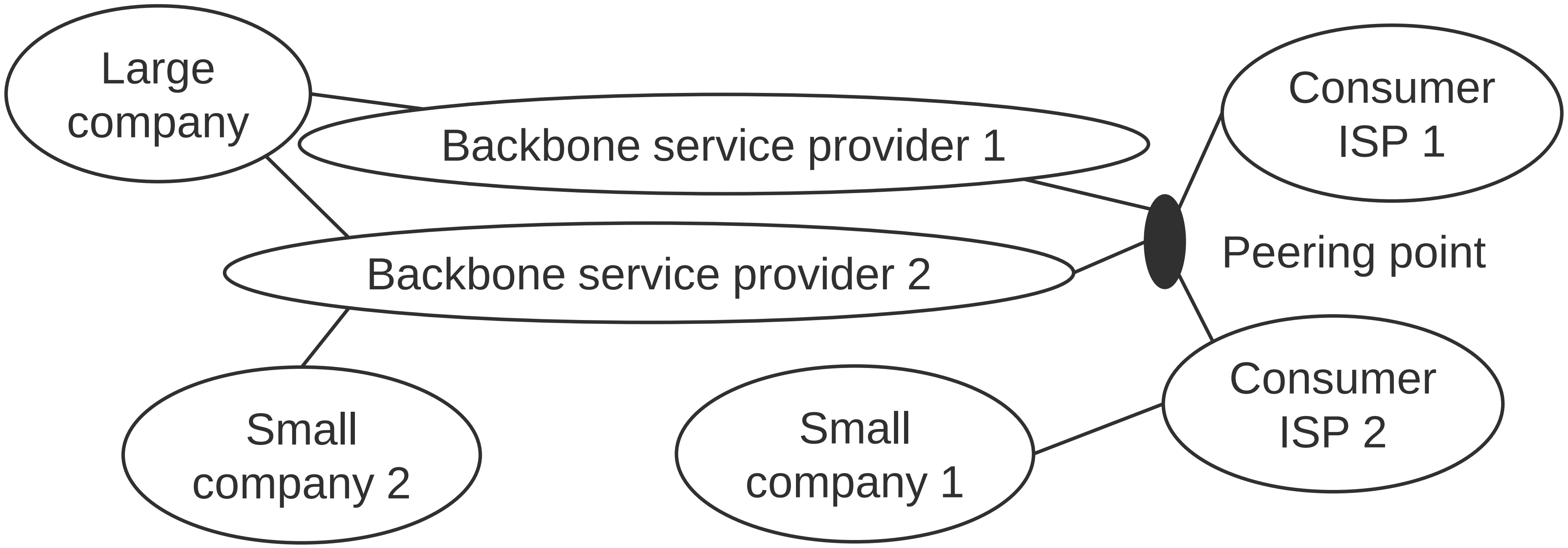
Backbone service providers

Consumer Internet Service Provider (ISP)

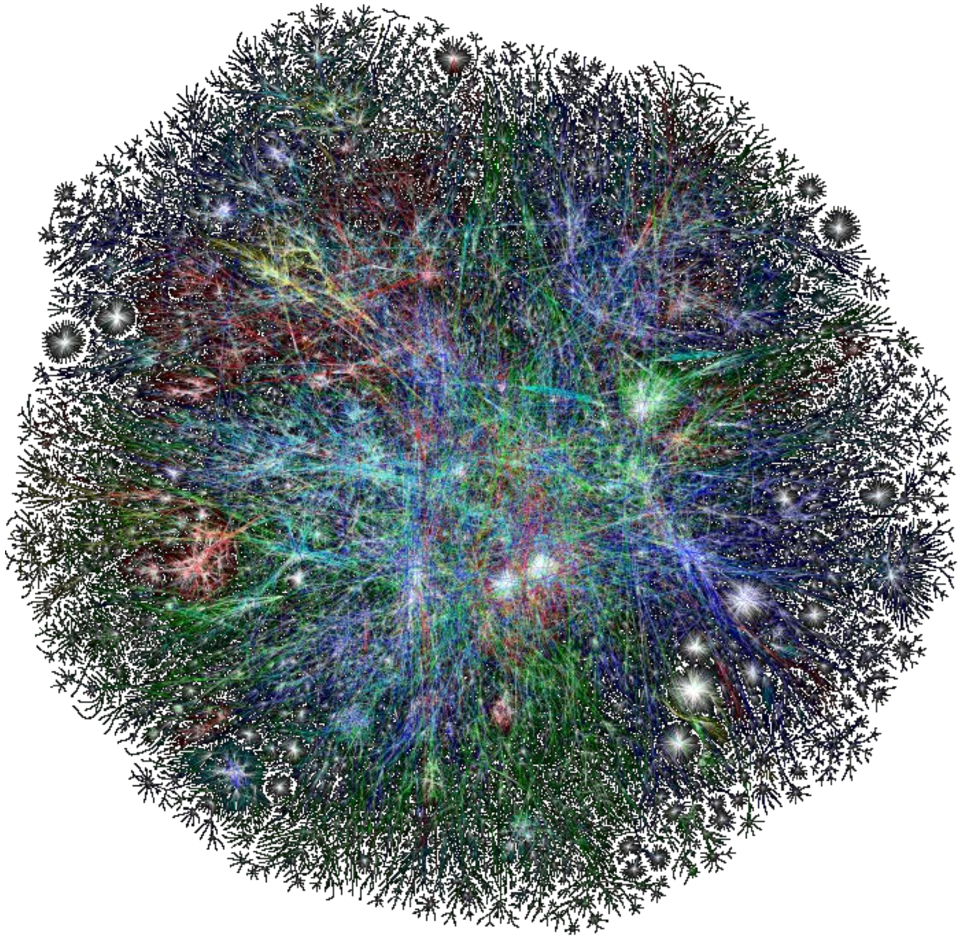
Peering Points – shortcuts between operators

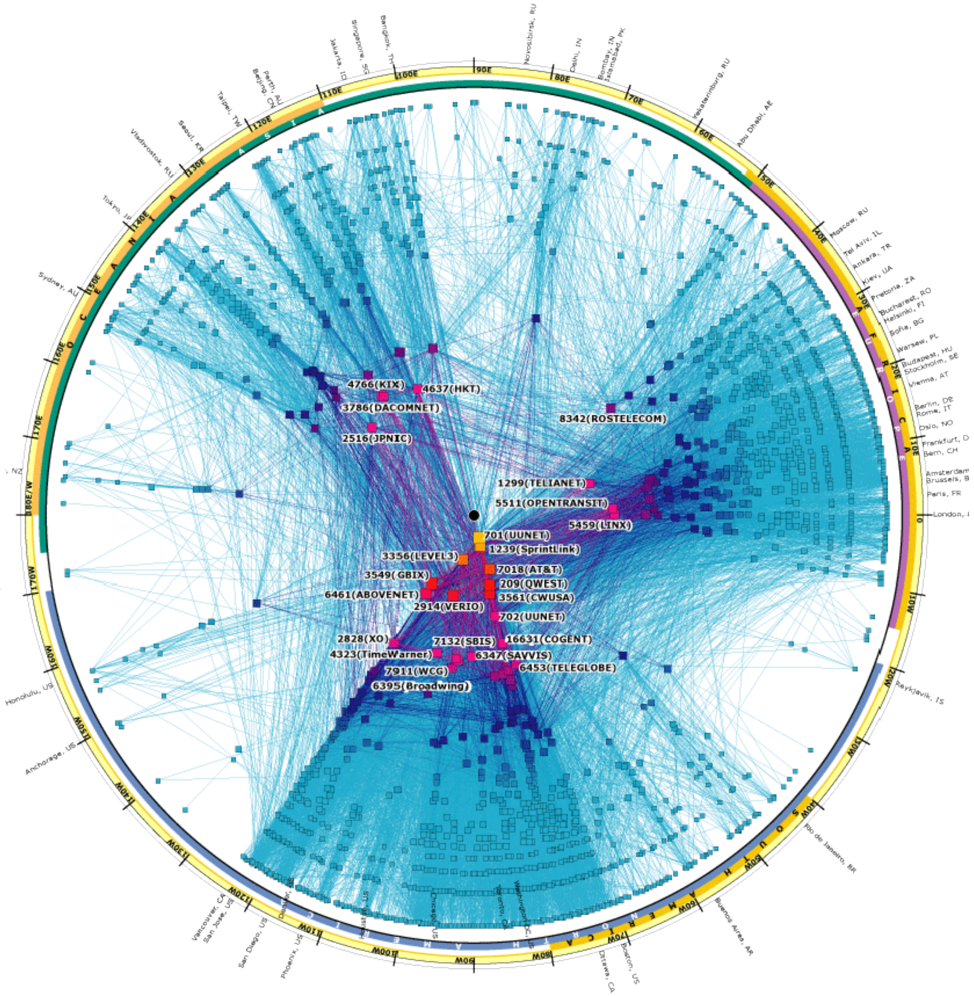
Consumers

- Direct backbone connectivity (companies) or ISP (private)



Structure of the Internet (“Reality”)





Source: www.caida.org

Exemplary Services in the Internet

World Wide Web (WWW)
- World-wide interlinked resources
- Based on “Hypertext Transfer Protocol” (HTTP)

Electronic mail (email)
- Exchange of digital multimedia messages
- Based on “Simple Mail Transfer Protocol” (SMTP)

File transfer
- Exchange of files
- Based on “File Transfer Protocol” (FTP)

Network management
- Monitoring and control of networked systems
- Based on “Simple Network Management Protocol” (SNMP)

P2P, VoIP, IPTV, CDN, ...

Many company-specific services: Skype, Gaming, ...

Classical Internet Design Principles

Minimalism and autonomy

- Independent operation of the network, no internal changes necessary if connected to other networks

“Best-Effort” services

- Network tries as best as possible to transmit data end-to-end
- Reliable communication is feasible through retransmission
    - Today several extensions towards quality-of-service (QoS) support exist

Stateless intermediate systems

- No intermediate system (routers) should keep state related to any end-to-end communication
    - Big difference to classical telephone networks (circuit vs. packet switched)
    - Alternatives necessary for quality-of-service support

Decentralized control

- No global, centralized control of all interconnected networks

Do we still have this situation today with >60% traffic handled by Google, Amazon, Facebook, Apple …?

Some (Historical) IP Design Principles

RFC 1958, based on papers from mid-80s

Make sure it works – before writing the standard

Keep it simple

Make clear choices

Exploit modularity

Expect heterogeneity

Avoid *static* options and parameters

Look for a good design; it need not be perfect

- 80-20 rule: 80% of effects comes from 20% of causes

Be strict when sending and tolerant when receiving

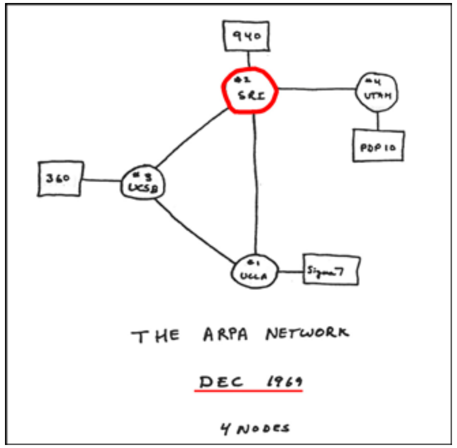
Think about scalability (with regard to nodes and traffic)

Consider performance and cost

➢ Looking back, some choices are not optimal anymore.

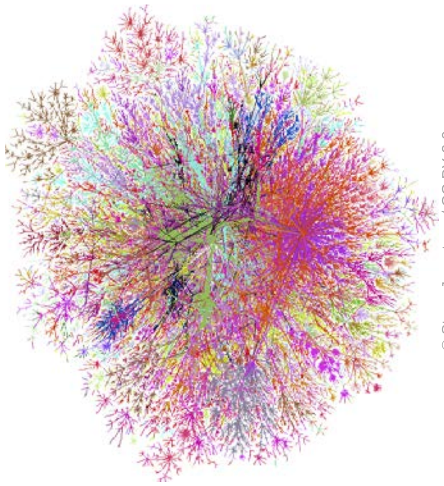
Development of the Internet

1962	DoD (Department of Defense): “Defense depends on communication.”
1967	ARPA (Advanced Research Project Agency) of the DoD: Project reliable packet network at SRI
1969	First “Internet” (4 hosts)
1971	Start of ARPAnet, the first Internet backbone
1974	New protocol suite: TCP/IP (Transmission Control Protocol/Internet Protocol)
1980	Integration of TCP/IP protocols into UNIX (BSD)
1988	IP connection to the Internet from Germany via EUnet - IRB Dortmund and XLink Karlsruhe
1991	EBONE: European backbone
1995	Internet becomes visible due to WWW
1996	University Corporation for Advanced Internet Development - Internet2
1999	Second Internet2-Backbone: Abilene
~2000	Rise and fall of dotcoms
2006	VoIP, Web 2.0 hype (and history repeats…)
2009	Clouds, more clouds
2010+	Everything is mobile (> 4.5bn subscribers), apps rule…
20xy	Internet of Things with > 30bn devices, IPv6 finally everywhere

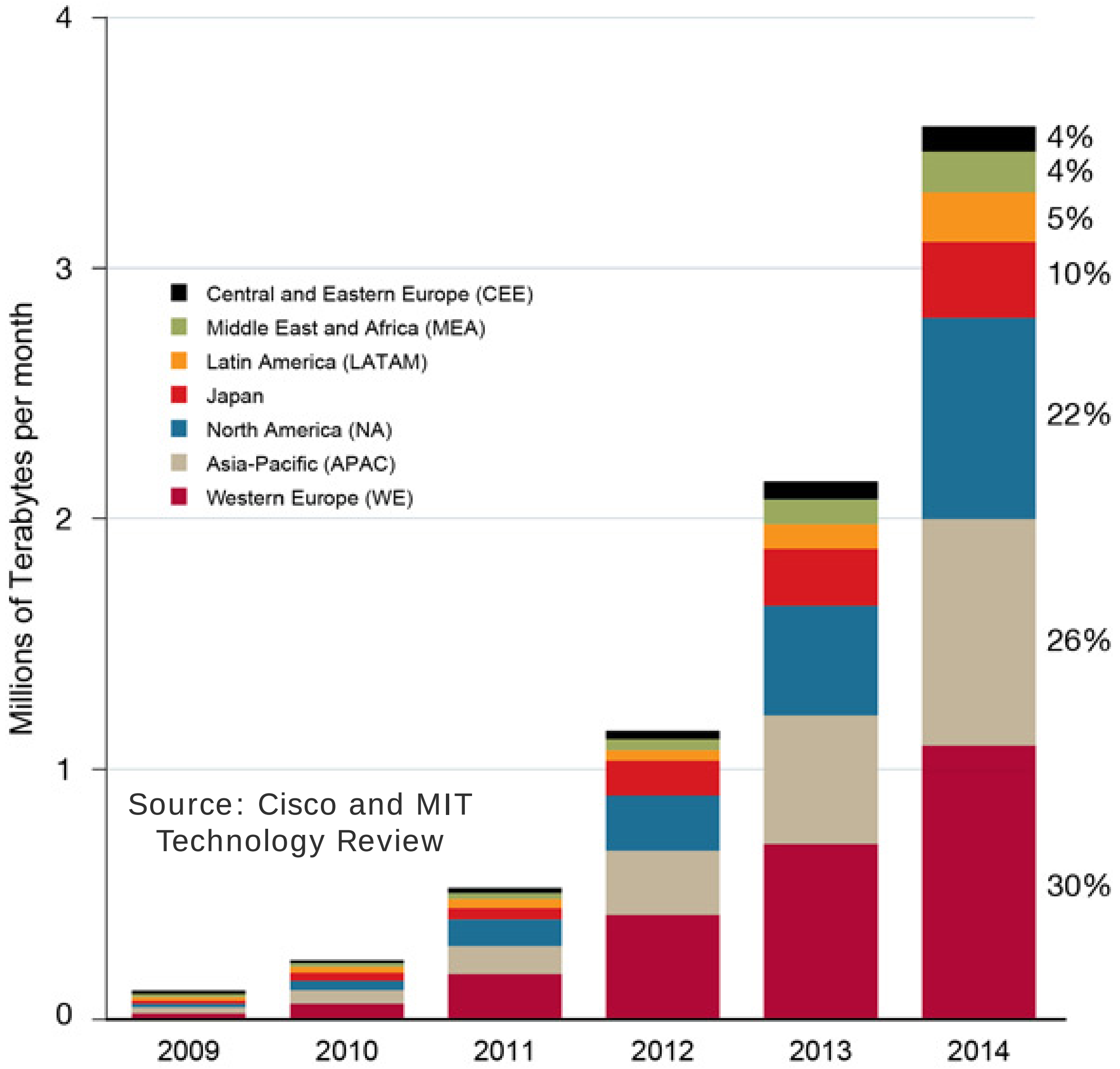


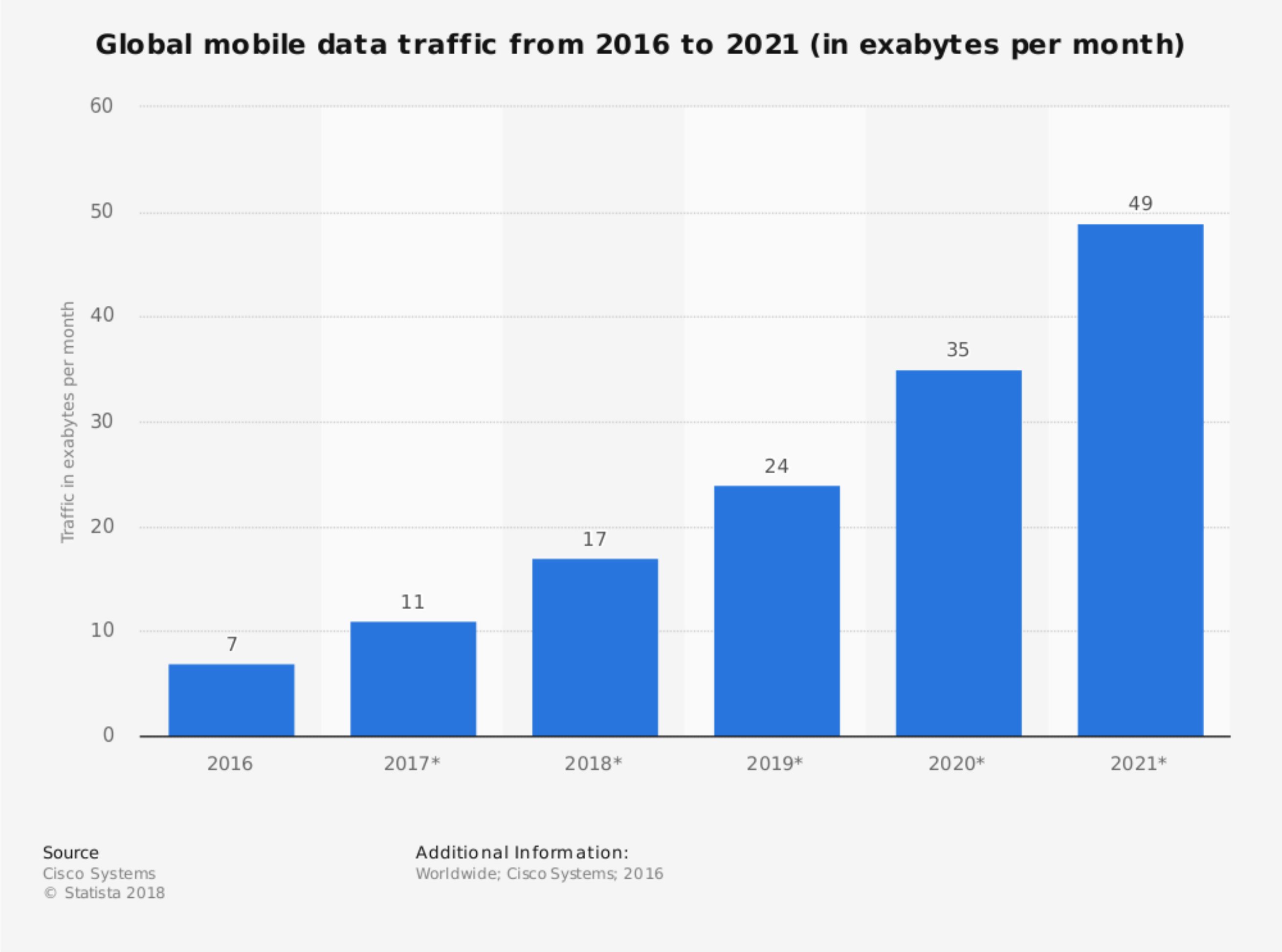
Scan of ARPANET logic map, circa 1969, © SRI International

1969
…
2020



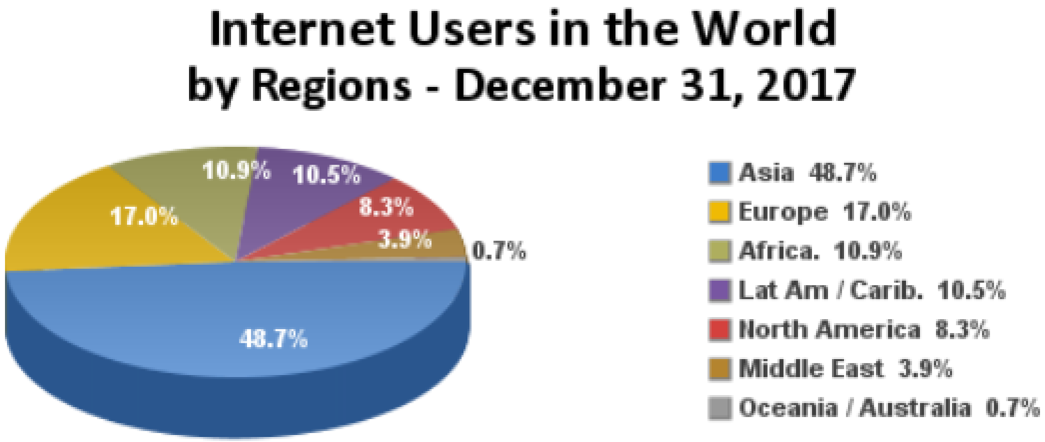
Global Mobile Data Traffic Forecast by Region



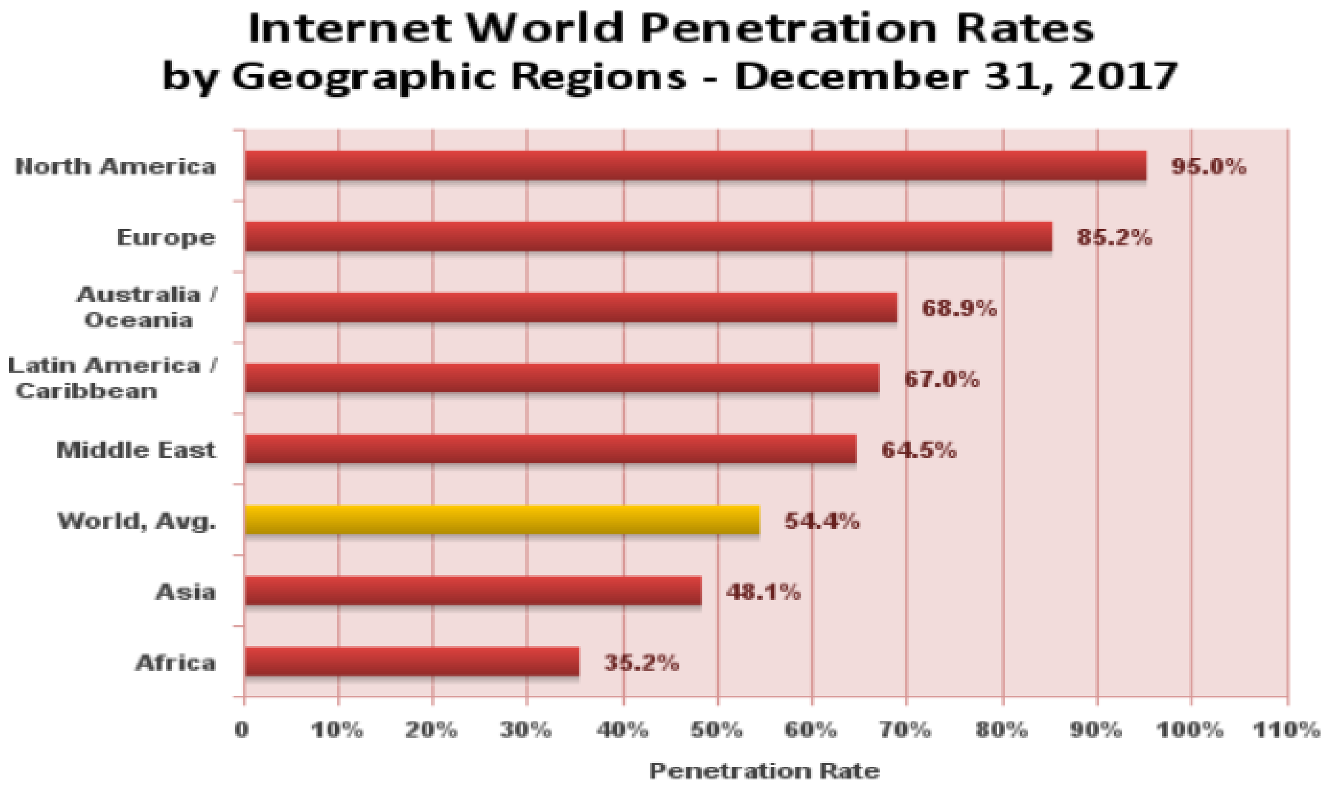


More than 50% is video!

Internet Users World-Wide



Source: Internet World Stats - www.internetworldstats.com/stats.htm
Basis: 4,156,932,140 Internet users in December 31, 2017
Copyright © 2018, Miniwatts Marketing Group



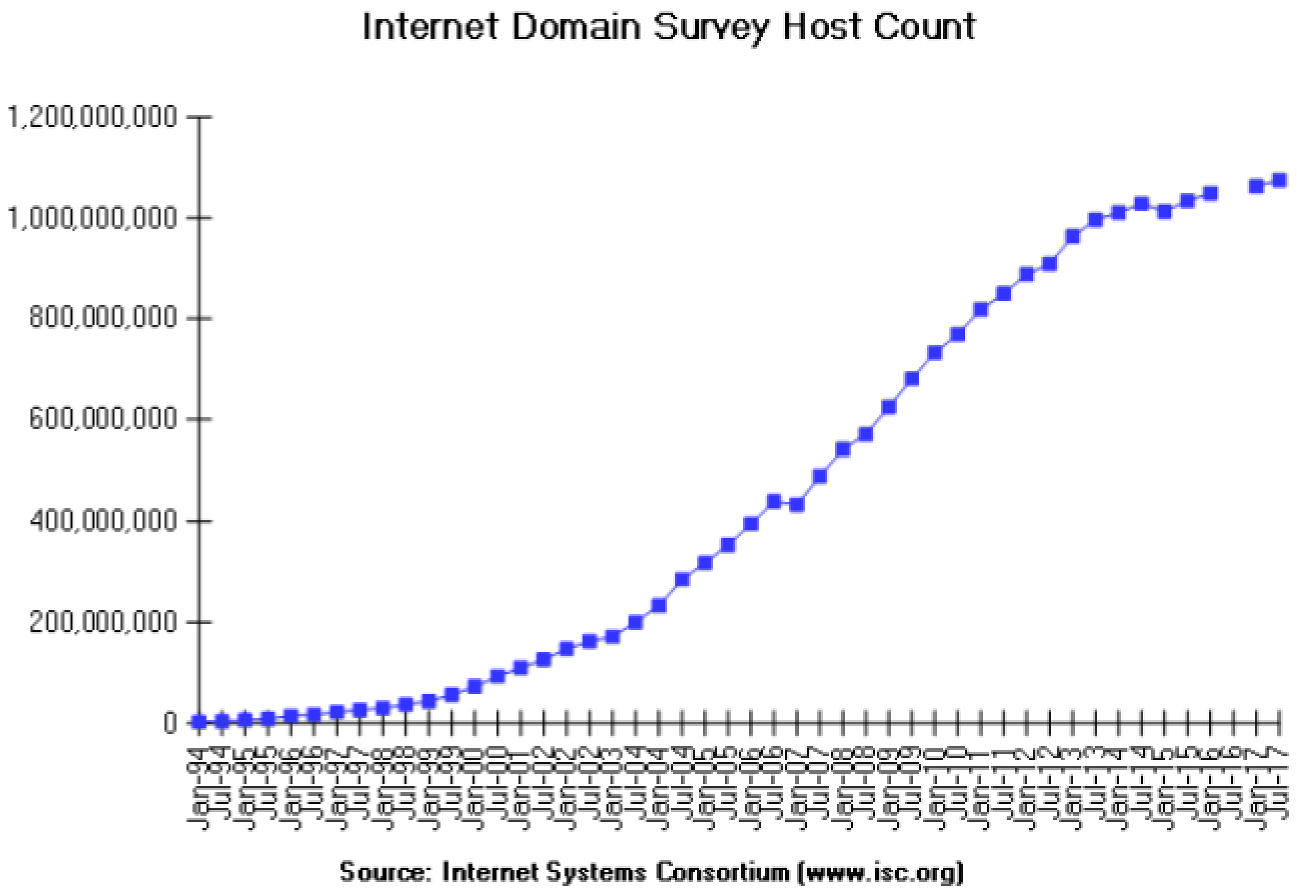
Source: Internet World Stats - www.internetworldstats.com/stats.htm
Penetration Rates are based on a world population of 7,634,758,428
and 4,156,932,140 estimated Internet users in December 31, 2017.
Copyright © 2018, Miniwatts Marketing Group

Hosts and Internet Domains

15 million .de – domains
> 1 billion hosts world-wide (but > 7 billion mobile devices, > 30 billion IoT predicted…)

Date: Start of May 2016 (* or latest available data)

Top Level Domain	Domains Worldwide
.com	125,973,142
.cn	18,787,105
.de	16,134,584
.net	15,665,055
.org	10,904,338
.uk	10,798,153
.nl	5,623,775
.info	5,489,556
.ru	5,206,897
.eu	3,827,745



Sources: DENIC (www.denic.de), Internet Systems Consortium, Inc. (http://www.isc.org/)

Questions & Tasks

- Check the latest numbers regarding domains, hosts, users, penetration!
- What is the job of a router (from a high-level perspective)?
- What do applications see from the network stack?
- Who owns the Internet?
- Why having peering points?
- What do many services in the Internet have in common?
- Compare the classical Internet design principles with today’s applications and their requirements. What challenges do arise?

PROTOCOLS

The Classical Internet Protocol Suite

TCP (Transmission Control Protocol)

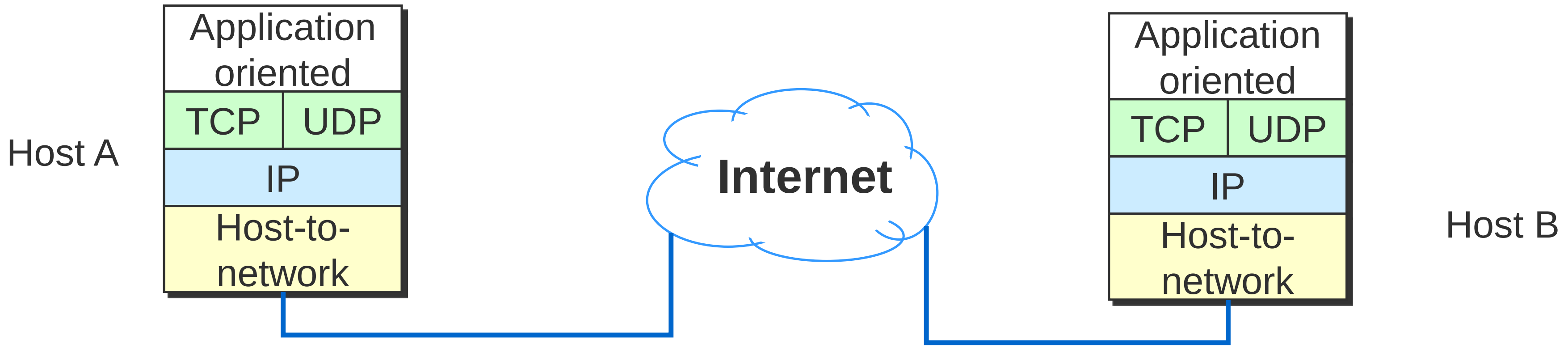
- Reliable, connection oriented transport protocol over unreliable IP (Internet Protocol)

UDP (User Datagram Protocol)

- Connectionless transport protocol, offers application interface to IP plus multiplexing

Examples for application oriented protocols

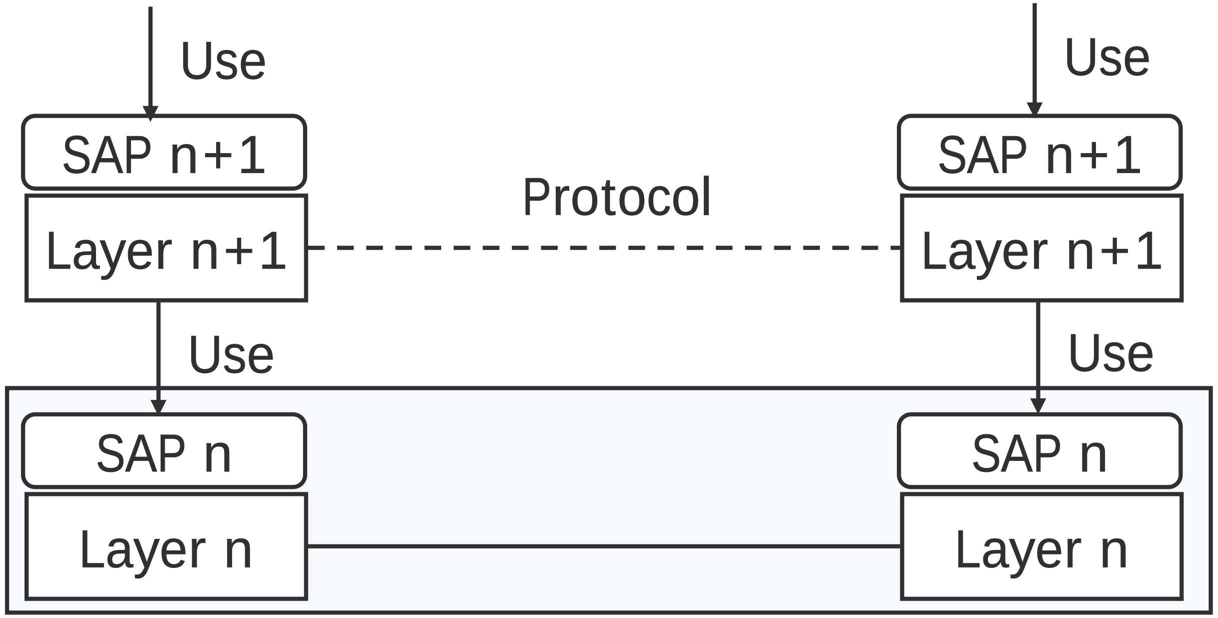
- HTTP: HyperText Transfer Protocol
- FTP: File Transfer Protocol
- Telnet: Simple terminal protocol



Protocols

Protocols are a set of rules

- Describe how two (or more) remote parts of a layer cooperate to *implement the service* of the given layer
    - Behavior, packet formats
- These remote parts are called *peer protocol entities* or simply *peers*
- Use the service of underlying layer to exchange data with peer

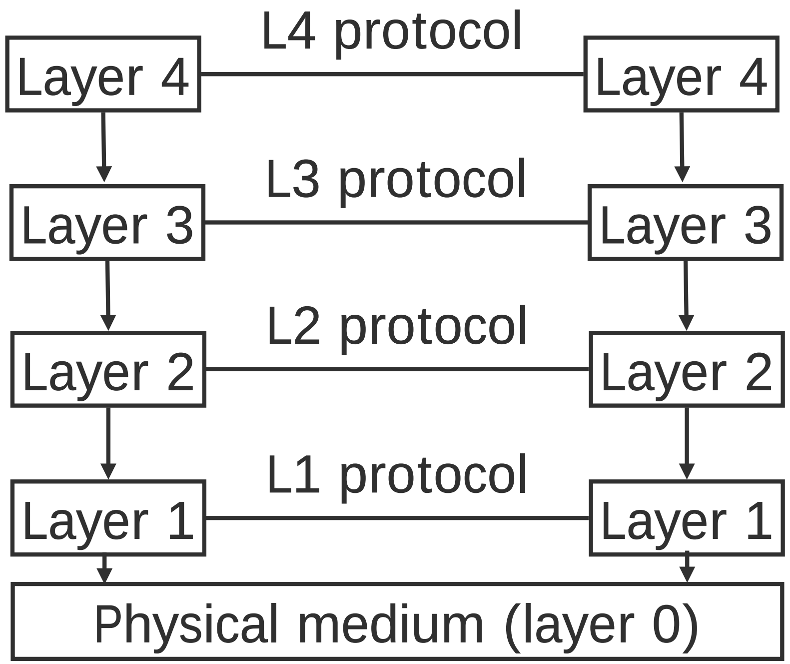


Protocol Stacks

Typically, several layers and thus several protocols in real system

Layers/protocols are arranged as *(protocol) stack*

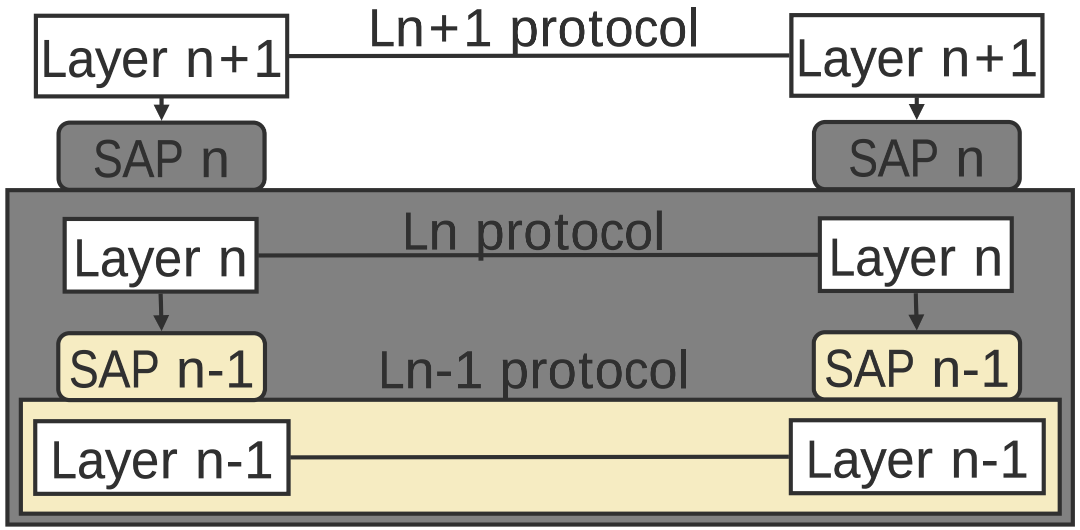
- One atop the other, *only* using services from directly beneath (so-called *strict layering*)



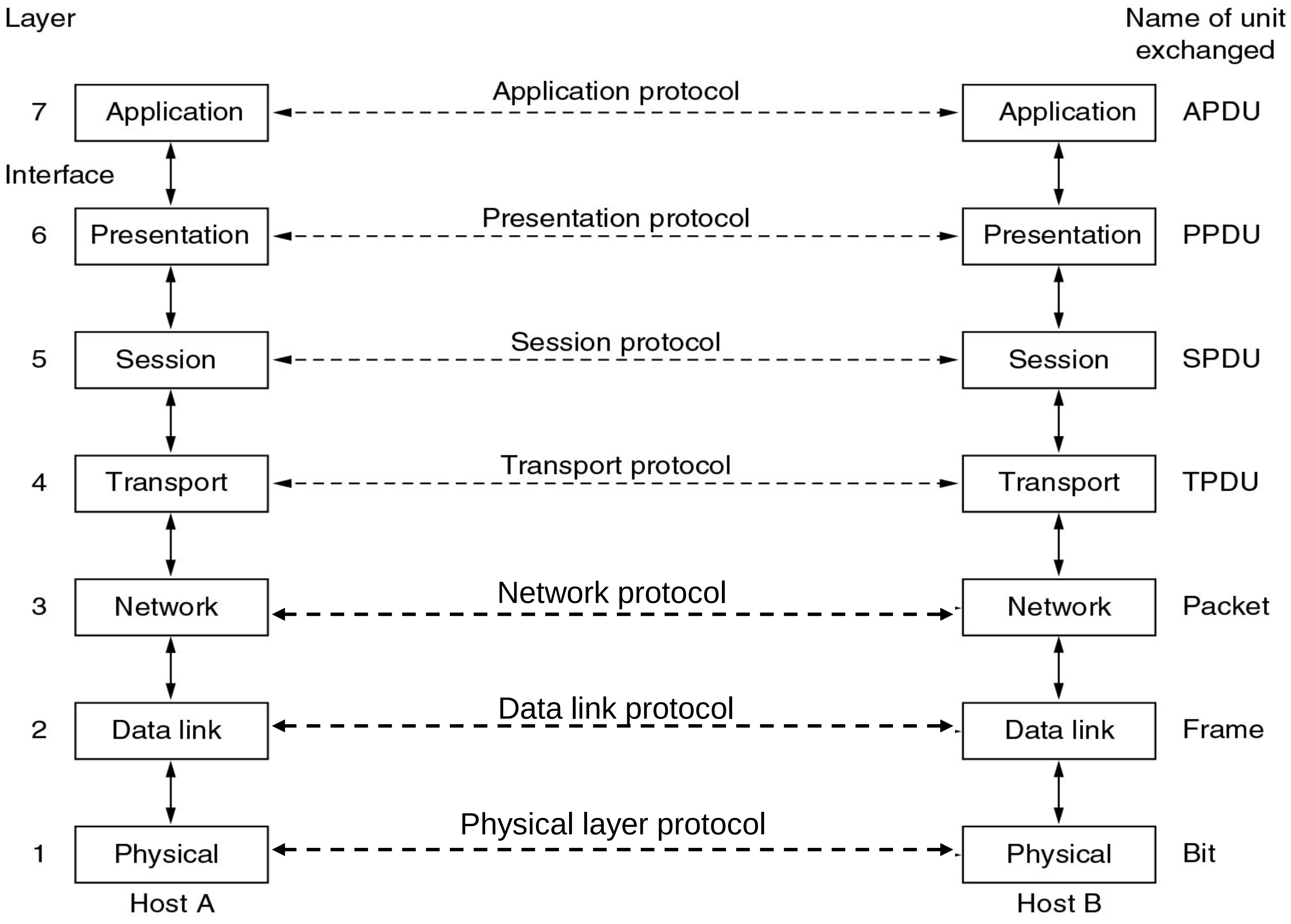
Layers Do Not Care About Distributed Lower Layers

A given layer n+1 does not care about the fact that its lower layer is actually distributed …

- Layer n+1 imagines layer n as something that “just works”, has service access points where they are necessary
- In reality, layer n of course is distributed in turn, relying on yet lower layers
- At the end, the physical medium (layer 0) is transporting signals (as physical representation of data)



ISO/OSI 7-layer Reference Model

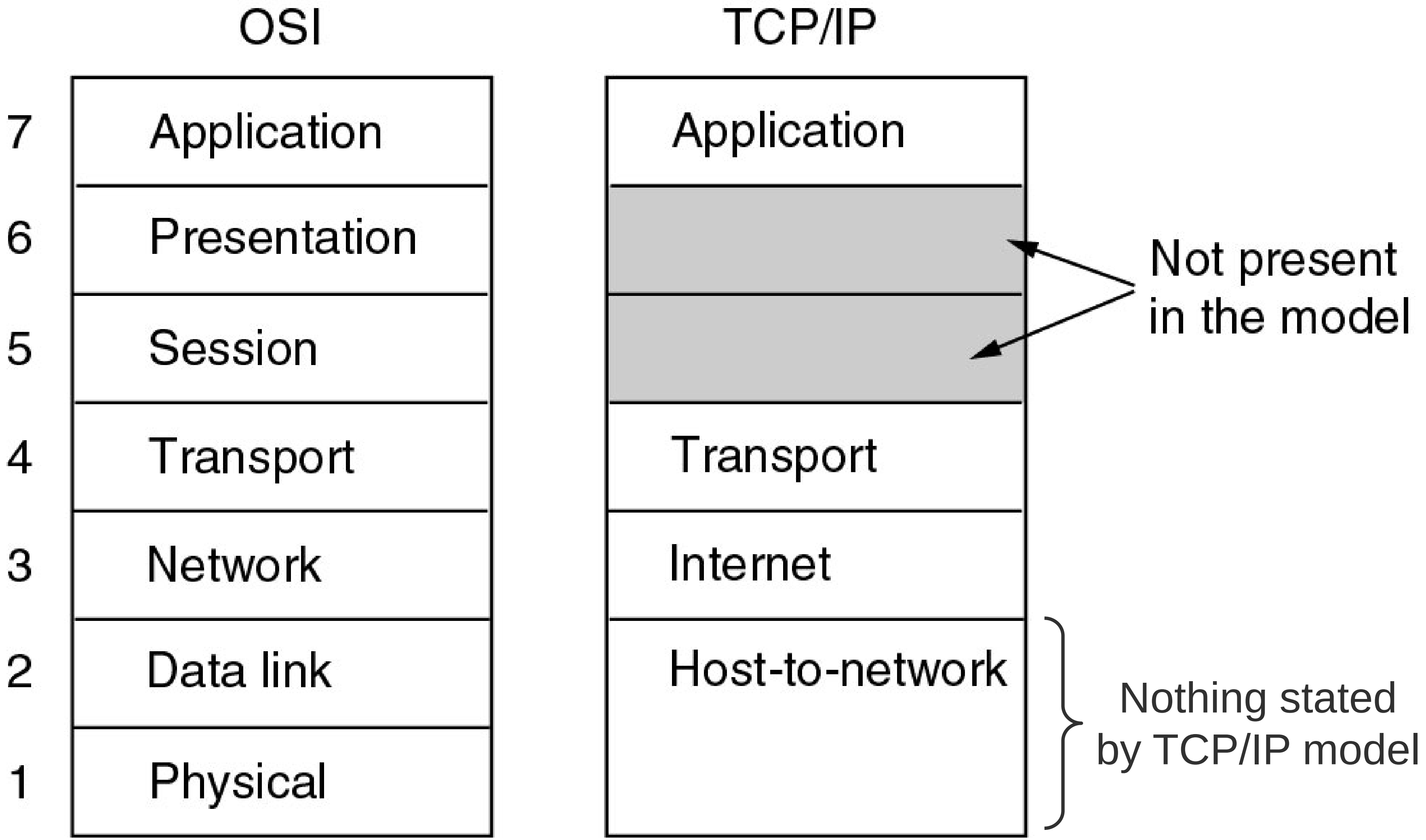


PDU: Protocol Data Unit

Seven Layers (in brief)

1. **Physical layer**: Transmit raw bits over a physical medium
2. **Data Link layer**: Provide a (more or less) error-free transmission service for data frames over a shared medium
3. **Network layer**: Solve the forwarding and routing problem for a network
4. **Transport layer**: Provide (possibly reliable, in order) end-to-end communication, overload protection, fragmentation
5. **Session layer**: Group communication into *sessions* which can be synchronized, checkpointed, …
6. **Presentation layer**: Ensure that syntax and semantic of data is uniform between all types of terminals
7. **Application layer**: Actual application, e.g., protocols to transport web pages

TCP/IP Protocol Stack



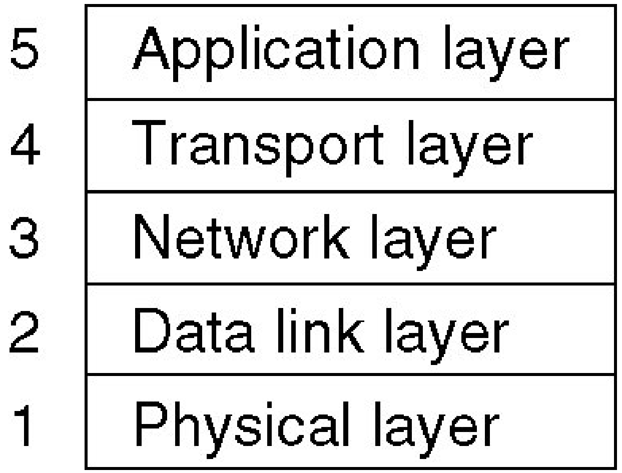
ISO/OSI versus TCP/IP

ISO/OSI: Very useful model, almost non-existing protocols

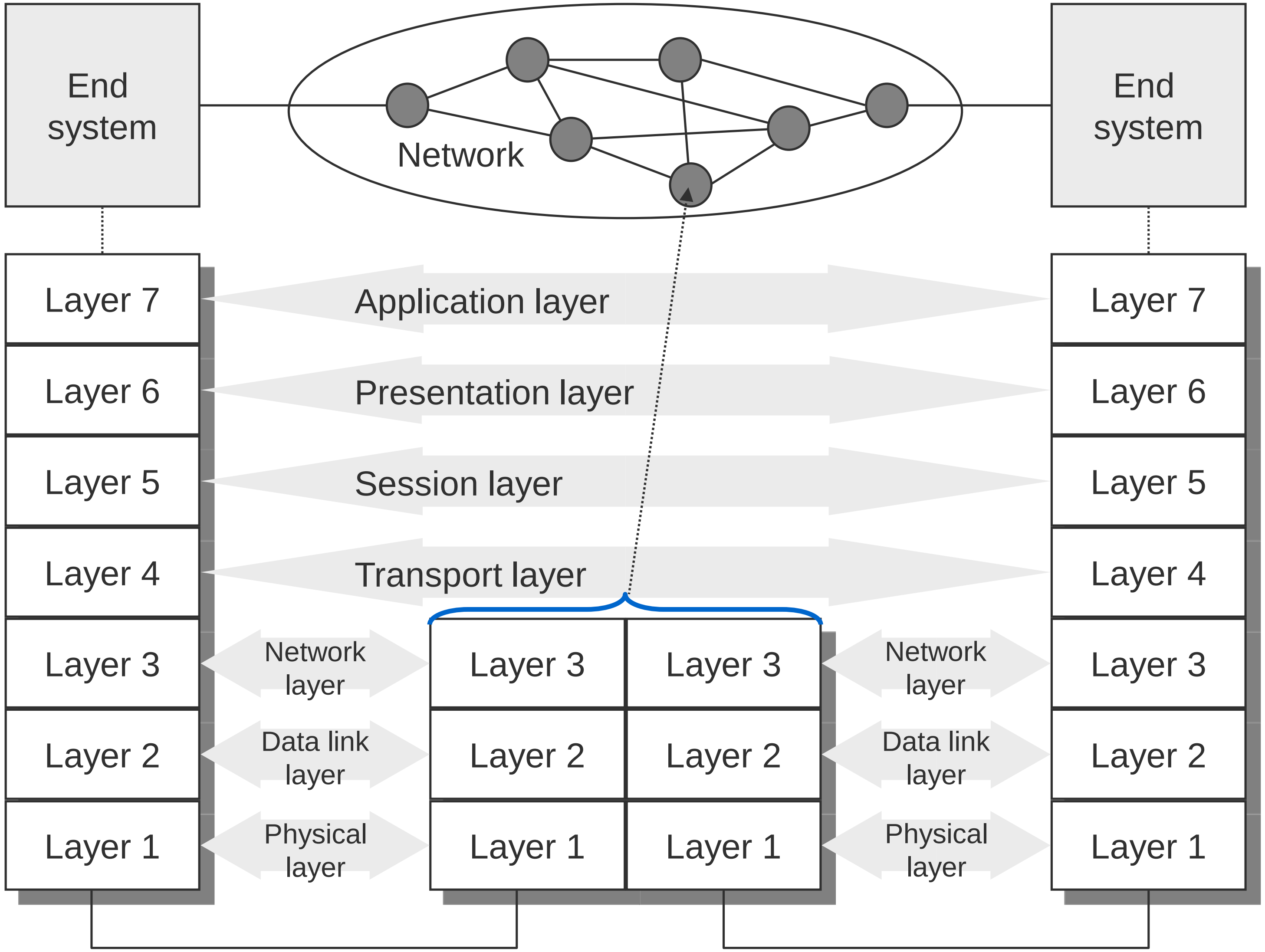
TCP/IP: Non-existing model, very useful protocols

➢Use simplified ISO/OSI model, but treat TCP/IP protocol stack in context of this model

- With suitable add-ons especially for the lower layers



7 Layers with Intermediate System

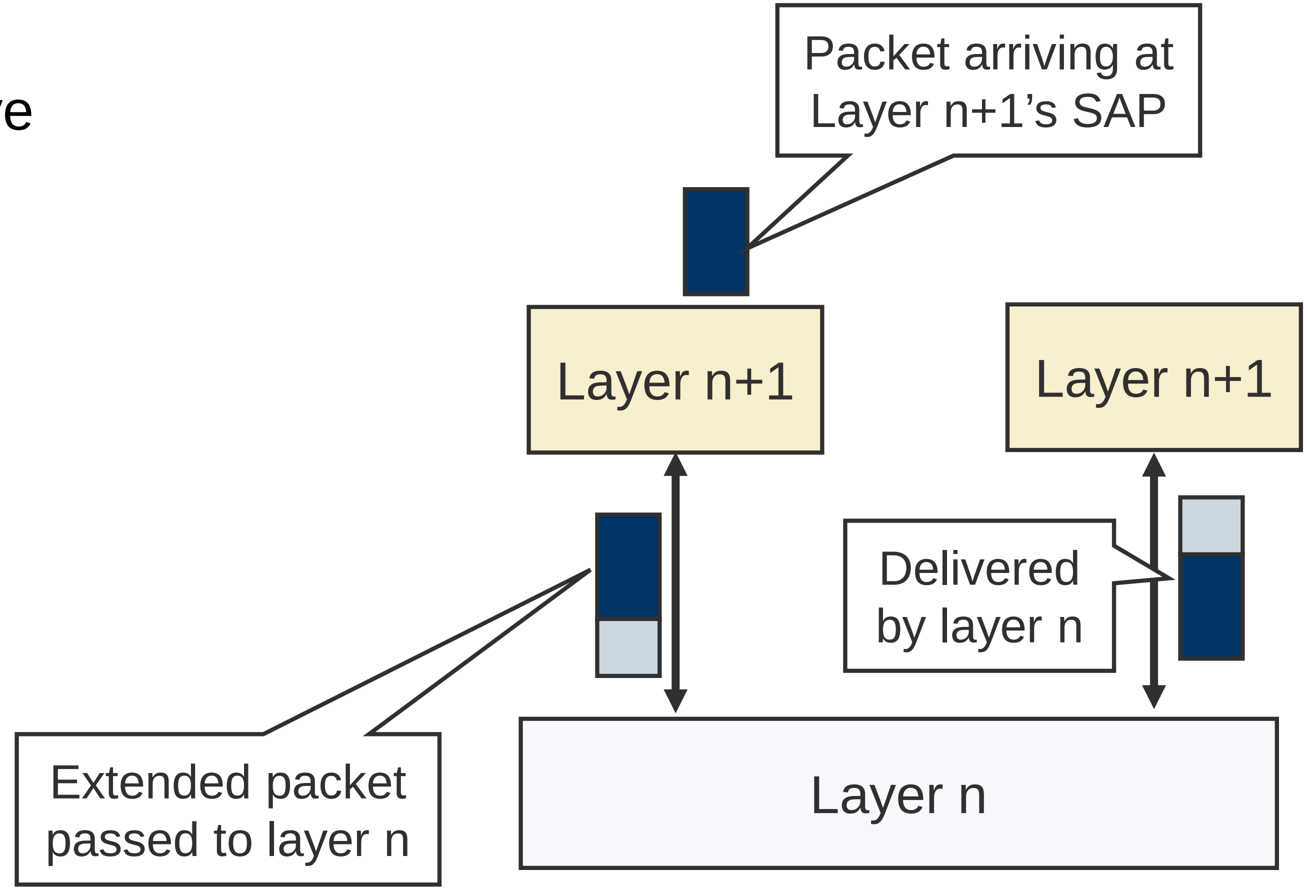


Protocols and Messages

When using lower-layer services to communicate with remote peer, administrative data is usually included in those messages

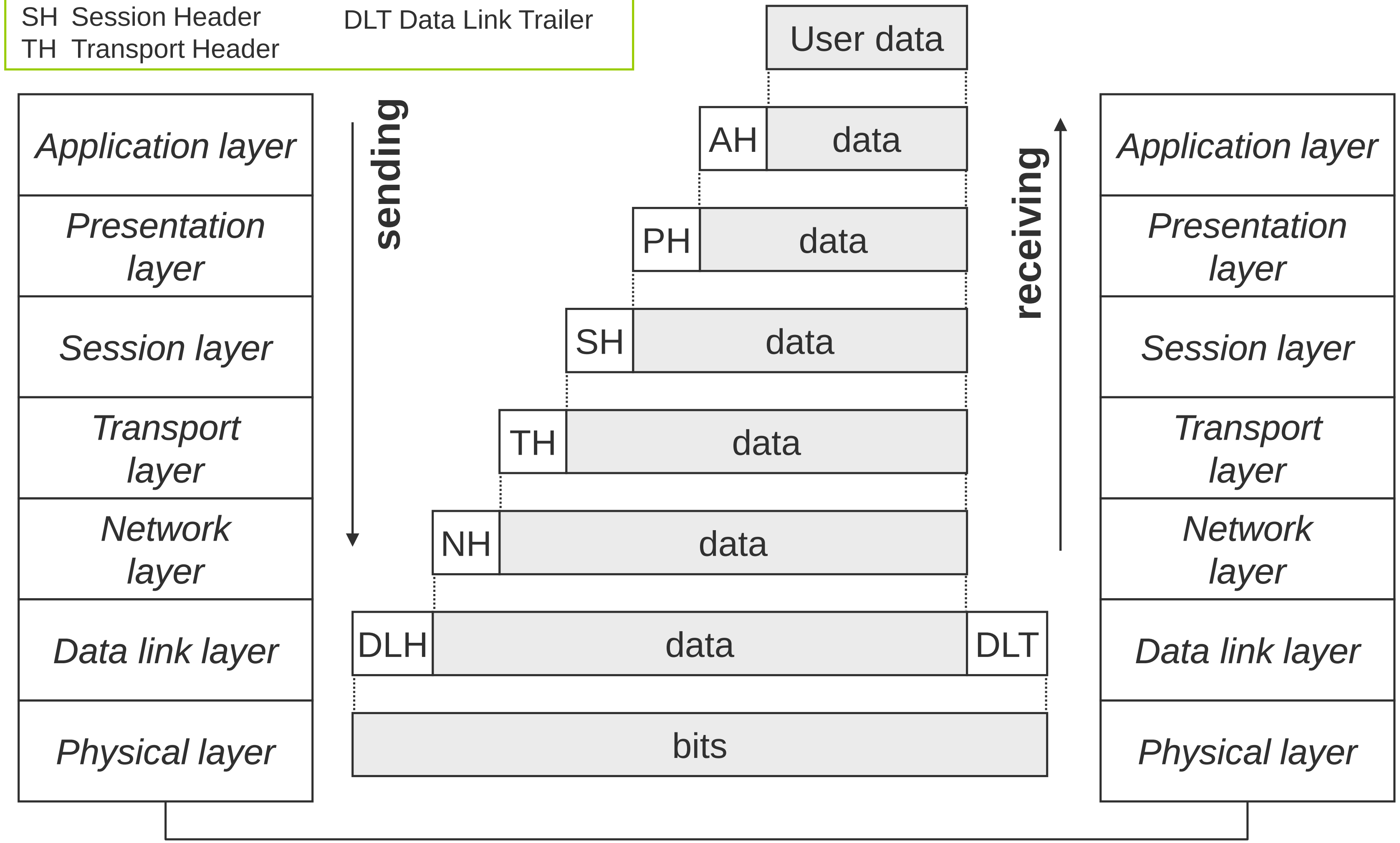
Typical example:

- Protocol receivers data from higher layer
- Adds own administrative data (header/trailer)
- Passes the extended message down to the lower layer
- Receiver will receive original message plus administrative data



Encapsulation of Data

AH Application Header
PH Presentation Header
SH Session Header
TH Transport Header
NH Network Header
DLH Data Link Header
DLT Data Link Trailer



Questions & Tasks

- Layering – again! Where do you already know this from?
- What is a protocol? What is a peer?
- What is the idea of strict layering? Advantages/disadvantages?
- What are the differences between TCP and UDP? Advantages/disadvantages?
- Which layers just virtually transport data, which one does this in real?
- Encapsulation comes with the layering – what are advantages and disadvantages?
- Do you know encapsulation from systems outside computer networks?

Content