Computer Networks

Chapter 01

prof. dr ir Maarten van Steen

Vrije Universiteit Amsterdam
Faculty of Science
Dept. Mathematics and Computer Science
Room R4.20. Tel: (020) 444 7784

steen@cs.vu.nl

Contents

Overview

Hardware: Talk just a bit about how you can configure a bunch of computers into a network:

- Local Area Networks (LAN)
- Metropolitan Area Networks (MAN)
- Wide Area Networks (WAN)
- Internetworks

Software: This is what actually makes computer networks – not the hardware!

- Protocols: describe *how* two communicating parties exchange information.
- Services: describe *what* a network offers to parties that want to communicate.
- Interfaces: describe *how* a client can make use of network services, i.e. how the services can be accessed.

Reference models: Describe how the OSI and Internet networks are organized.

Computer Networks
Hardware

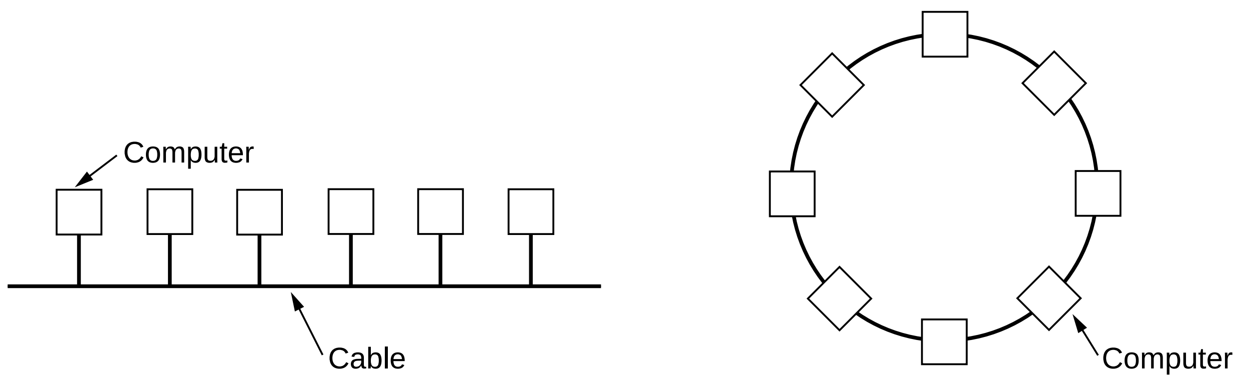
Broadcast network: a single communication channel is shared by all computers $\Rightarrow$ sending a packet implies that all others receive it.

Point-to-point network: Computers are connected in pairs $\Rightarrow$ sending a packet goes strictly from the sender to the receiver, possibly having to visit intermediate machines (*routing*).

Interprocessor distance	Processors located in same	Example
1 m	Square meter	Personal area network
10 m	Room	Local area network
100 m	Building	Local area network
1 km	Campus	Local area network
10 km	City	Metropolitan area network
100 km	Country	Wide area network
1000 km	Continent	Wide area network
10,000 km	Planet	The Internet

Local Area Networks (LAN) (1/2)

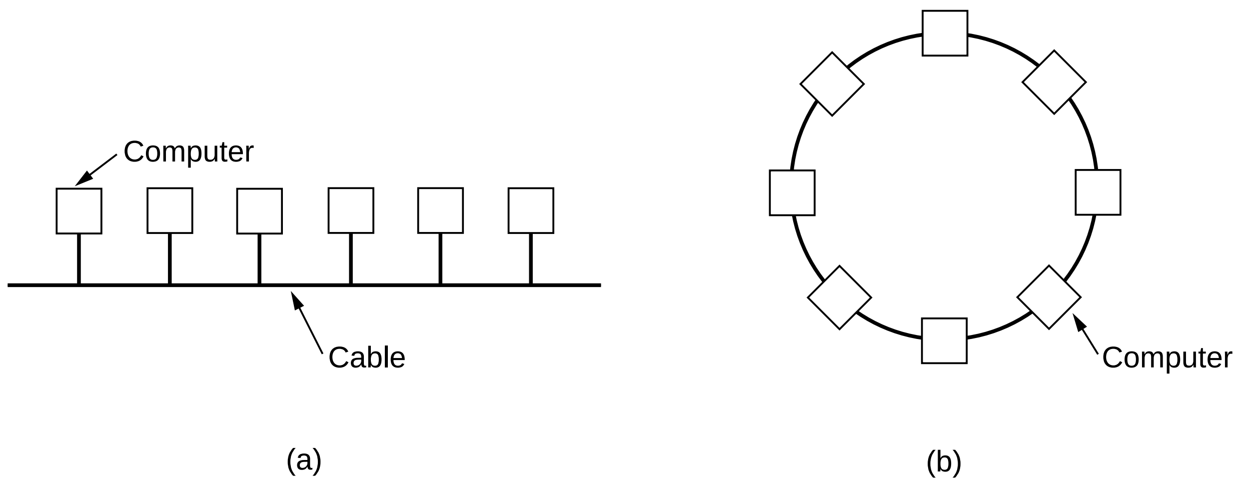
Apart from scale, LANs distinguish themselves from other networks by (generally) using **broadcast** technology, and having simple **topologies**:



(a) (b)

Type (a) (Bus-based): All computers are connected to the same wire. When one of them starts sending, the signal is propagated to all others. If two of them start sending at the same time, packets collide and rubbish is the result.

Local Area Networks (LAN) (2/2)

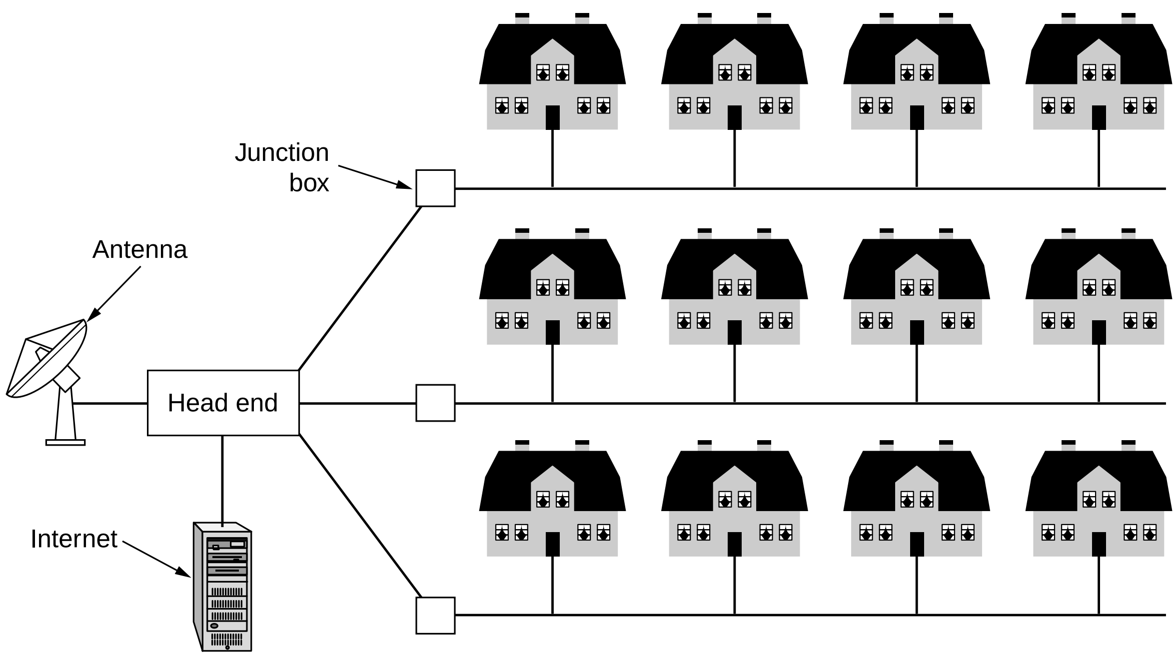


Type (b) (Token-based): a token (which is just a small packet) *continuously* circulates along the ring. A sending computer

(1) waits until the token passes and removes it
(2) sends its packet along the ring,
(3) waits until the packet returns
(4) reinserts the token.

Metropolitan Area Networks (MAN)

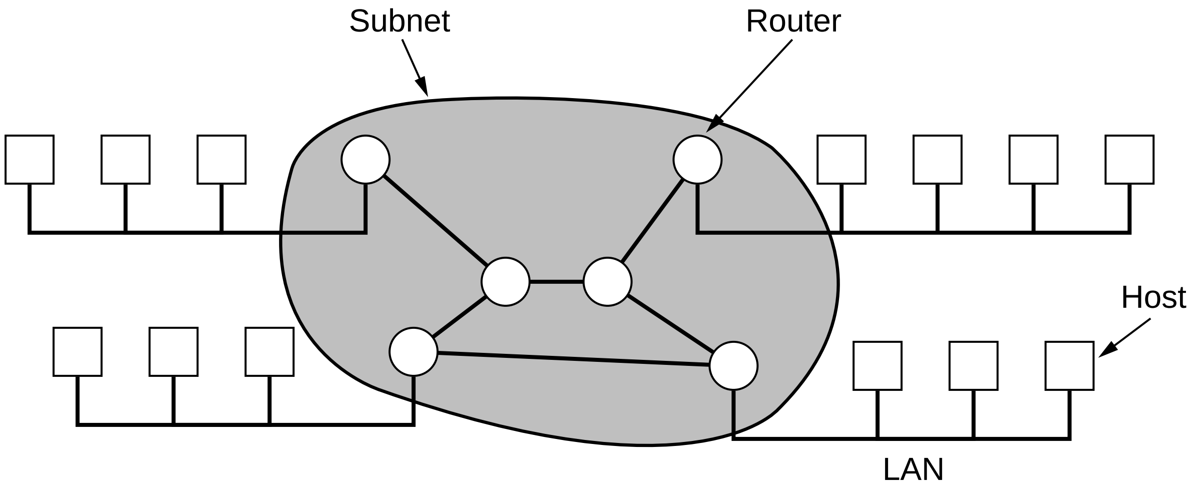
A MAN is intended to cover an entire city. A well-known example of a MAN is the **cable television network**:



In essence, the original network was used only for broadcasting radio and TV, but it soon became clear that the infrastructure could be used for two-way data communication as well.

Wide Area Networks (WAN)

Note: LANs and MANs generally don't have any **switching elements**: the wire does all the work. This makes them extremely efficient, although harder to scale. Here's where WANs come in.



In a WAN, **hosts** are connected to a **subnet**, which in turn consists of **routers** (switching elements) and **transmission lines**.

Routers generally adhere to a **store-and-forward** principle: incoming packets are first buffered (stored), the router takes a decision on where the packet has to go, and forwards the packet across the selected output line.

Wireless Networks (1/2)

In wireless networks, the transmission occurs through *unguided media*. Distinguish three different categories:

System interconnection: Basically for replacing the traditional cables that connect local devices to each other (monitor–keyboard–mouse–..., cordless telephone, stereo components).

Wireless LANs: Computers communicate through a (wired) base station that is placed on the ceiling, or directly in a peer-to-peer fashion (e.g., home networks).

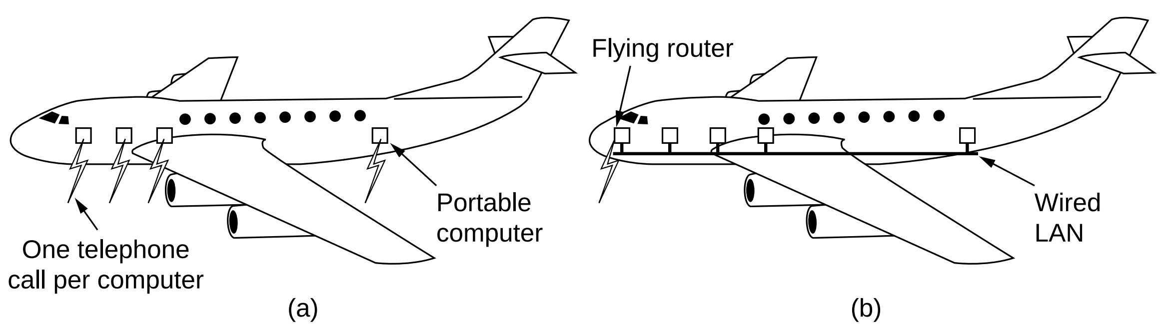
Wireless WANs: Typically the digital cellular phone networks fall into this category. New technology is now pushing bandwidth to 50 Mbps over several kilometers.

Wireless Networks (2/2)

Important: Make distinction between wireless and mobile:

Application	W	M
Office workstations	N	N
Notebooks	N	Y
LAN in the jungle	Y	N
LAN on a ship	N	Y/N
PDA	Y	Y

Note: Distinction is sometimes hard to make, especially when wireless and wired go together:



Home Networks

Essence: Many devices in the same home will be connected to a single network and communicate with each other:

- Computers
- Entertainment (audio equipment, game centers)
- Telecommunications (mobile phones, fax devices, intercom)
- Appliances (refrigerator, microwave, central heating, lights)
- Telemetry (alarms, cameras, thermostat)

Different models for home networking exist (completely decentralized versus centralized), but no single one can be pointed to as the winner. Note the specific requirements:

- 100% robust
- *very* easy to use
- self-managing
- high bandwidth

Internetworks

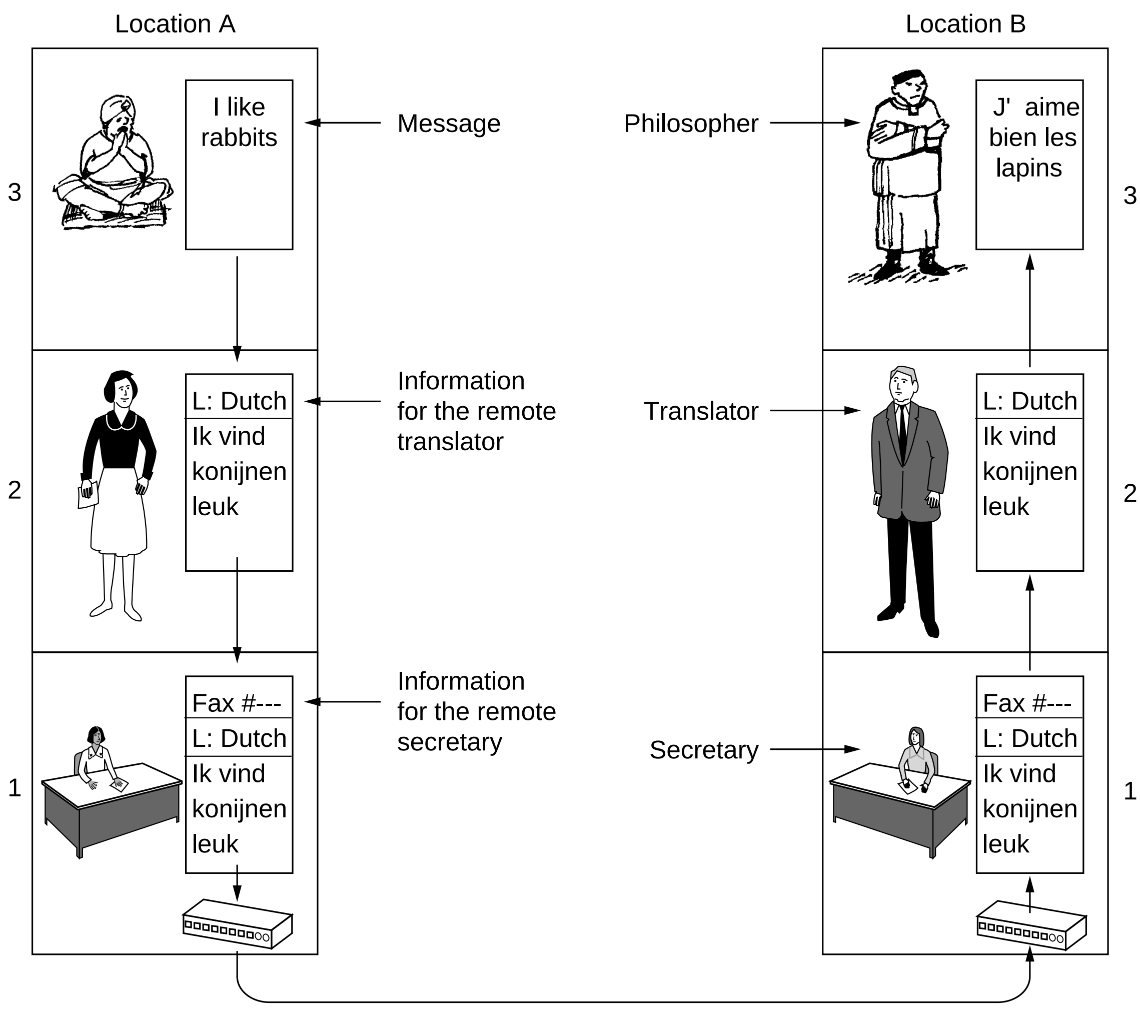
The assumption so far is that a network is **homogeneous**: there is hardly any variation in hardware and software. In practice, large networks can only be constructed by **interconnecting** different kinds of networks $\Rightarrow$ **internet(work)**.

Examples:

- Connecting a collection of different kinds of LANs (bus-based to token-based) within a department.
- Connecting LANs to each other through a WAN (think of enterprise networks for multinationals). The WAN acts as a subnet.
- Connecting WANs to each other (the Internet).

Protocol Hierarchy

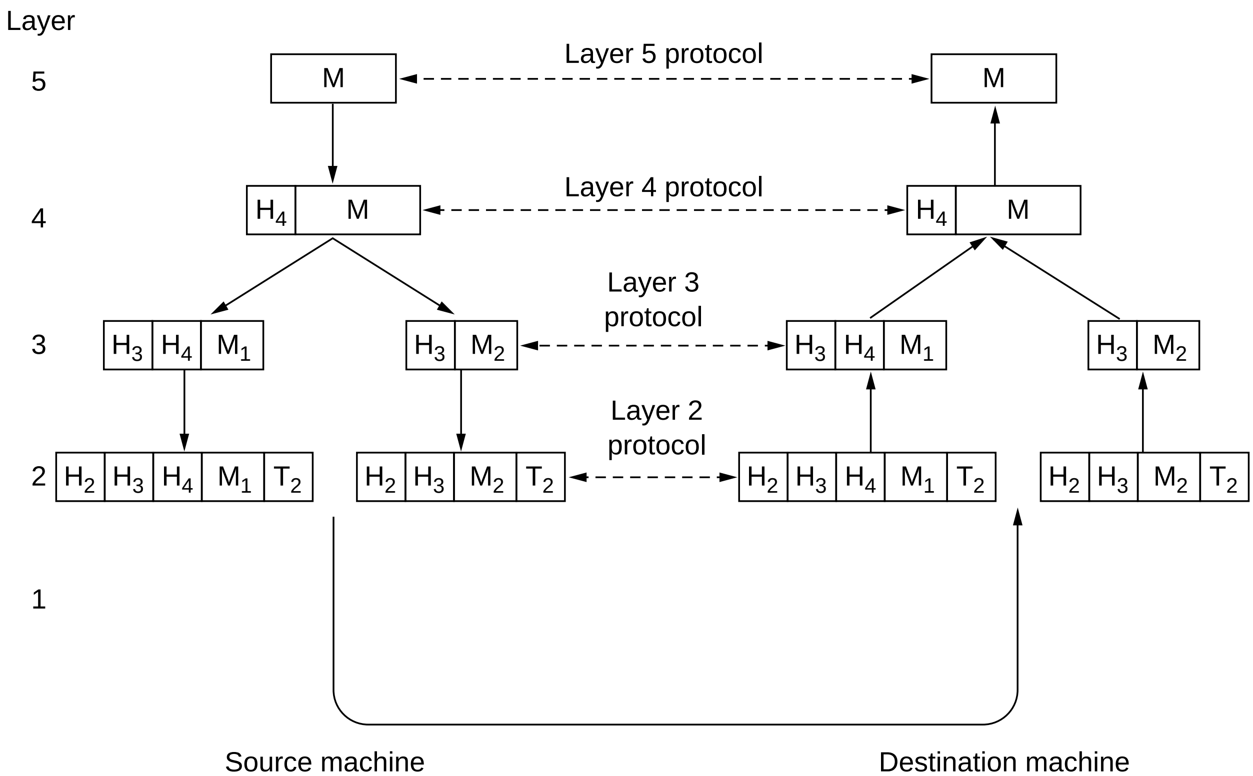
Fundamental to *all* software that makes a computer network run, is the notion of **protocol hierarchies**: structuring the services that a network must offer in terms of **layers**.



Layering: The Concepts

- Two parties at different sites, but at the same level, always agree on how they will exchange information $\Rightarrow$ specified in a **protocol**. Example: we all agree to speak Dutch (but use English on the sheets).
- In order for one party to send and receive information, it can only make use of the **communication services** offered by the layer directly underneath it. Example: The use of interpreters in negotiations between countries.
- Services offered by a layer are always fully specified in terms of an **interface** that makes those services accessible. Example: phones have buttons that allow you to “dial” a number.

Layering: An Example



Observation: In a protocol stack, Layer k puts its entire packet as **data** into a Layer $k-1$ packet; the latter may add a header and/or a trailer.

Note: It may even occur that Layer k data has to be split across several Layer $k-1$ packets $\Rightarrow$ **fragmentation**

Services: Connections or Not (1/2)

(Most) Network layers offer one or both of the following types of services:

Connection-oriented: This is the telephone model: you first **establish** a connection, then do a lot communication, and finally **release** the connection.

Connectionless: The postal model: your data is put into some kind of envelope on which the destination address has been written. The envelope + contents is sent to the destination, and that's it.

Services: Connections or Not (2/2)

	Service	Example
Connection-oriented	Reliable message stream	Sequence of pages
	Reliable byte stream	Remote login
	Unreliable connection	Digitized voice
Connection-less	Unreliable datagram	Electronic junk mail
	Acknowledged datagram	Registered mail
	Request-reply	Database query

Each service can provide some quality:

- *Is data delivered in the order it was sent?* With connections, this is generally the case.
- *Is data transmission **reliable**?* Generally offered with connections, but not always with connection-less services. Reliability requires sending acknowledgements $\Rightarrow$ performance may degrade.

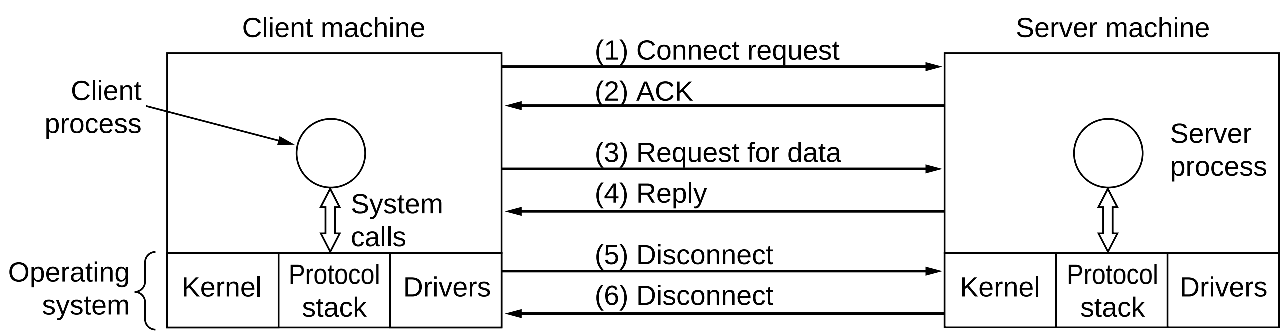
Service Primitives (1/2)

A service is specified by means of a set of **primitives**, also called **operations**, available to clients. Example of primitives for a simple connection-oriented service:

Primitive	Meaning
LISTEN	Block waiting for an incoming connection
CONNECT	Establish a connection with a waiting peer
RECEIVE	Block waiting for an incoming message
SEND	Send a message to the peer
DISCONNECT	Terminate a connection

Service Primitives (2/2)

Server	Client
LISTEN	
	CONNECT
RECEIVE	SEND
SEND	RECEIVE
DISCONNECT	DISCONNECT

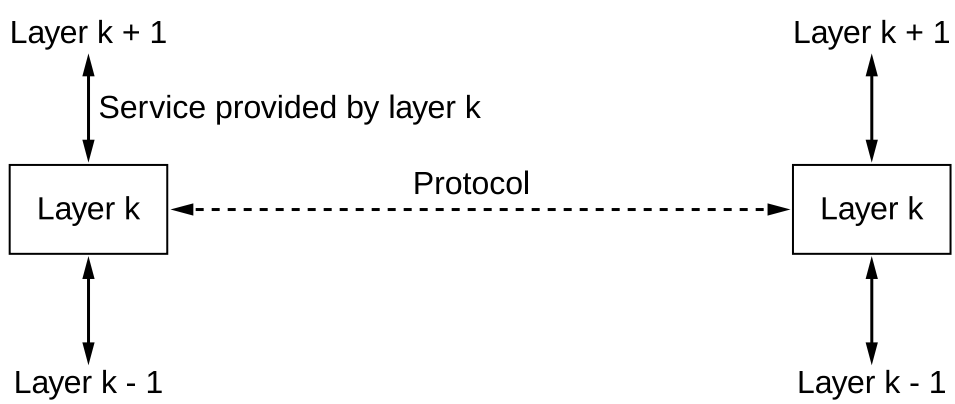


Services vs. Protocols

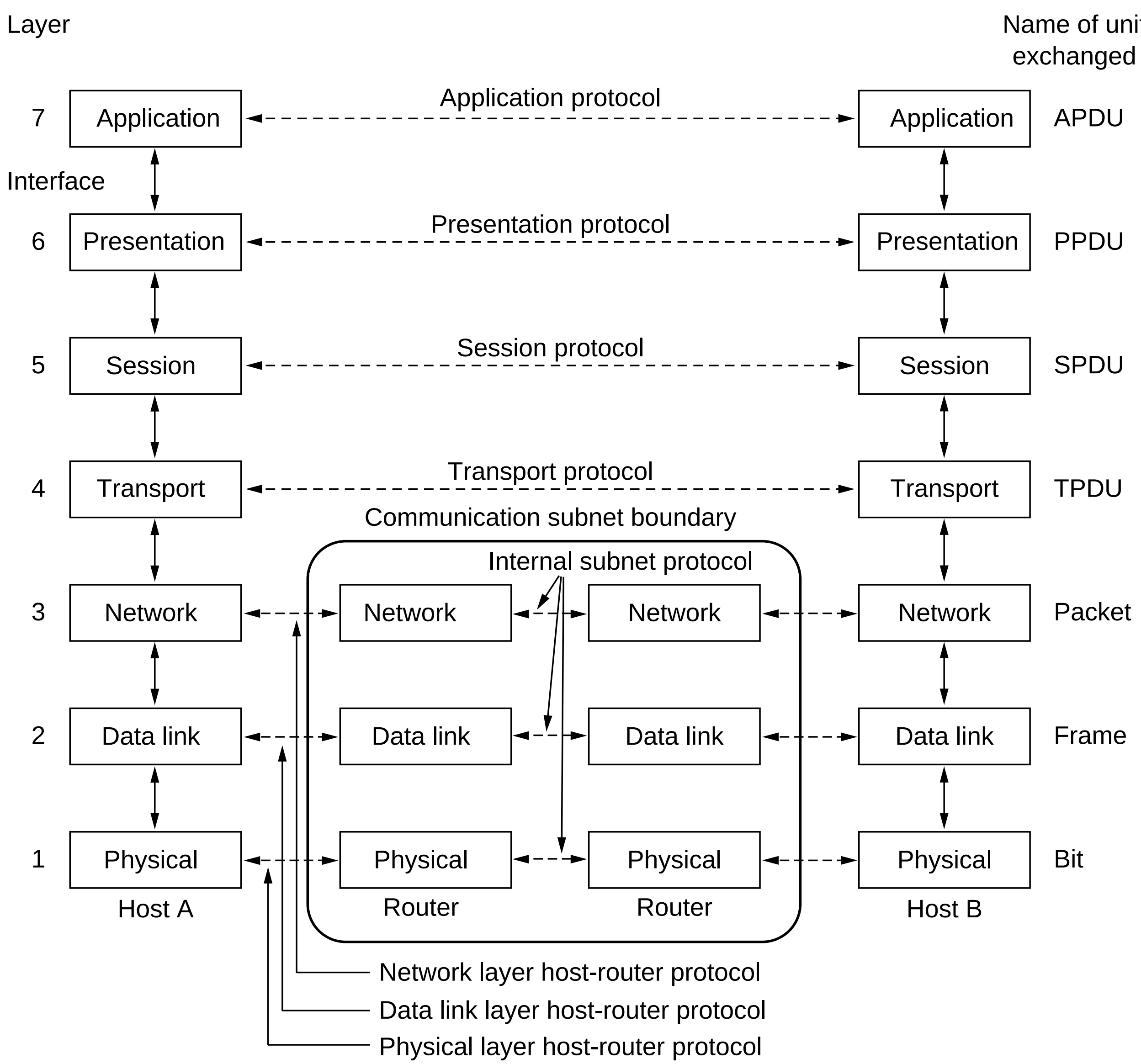
Service: set of primitives that a layer provides to the layer above it.

Protocol: set of rules that prescribe the layout and meaning of packets, and often the order in which specific packets should be sent.

Note: The same service can be realized by different protocols; each protocol can be implemented differently (using different data structures, programming languages, etc.)



The OSI Model



Note: Knowing the layering by heart is important

OSI Physical Layer

Essence: Describes the transmission of raw bits in terms of mechanical and electrical issues:

Example: Connect two computers by means of a wire:

- Setting -3V on the wire corresponds to a binary 1; +4V is a binary 0
- The wire is not to be longer than 15 meters
- You may change the voltage at most 20,000 times per second (**Question:** what's the transfer rate?)

OSI Data Link Layer (1/2)

Observation: We need to at least **detect bit transmission errors** $\Rightarrow$ send bits in **frames** that add redundancy to detect something went wrong

Bits	8	8	8	$\geqslant 0$	16	8
	0 1 1 1 1 1 1 0	Address	Control	Data	Checksum	0 1 1 1 1 1 1 0

Examples:

- Add a parity bit to every 7 transmitted bits: 1 says there were odd number of 1's; 0 says there were an even number of 1's
- Add a checksum (cyclic redundancy check) that should match the bits before it

Also: Provide the mechanisms so that fast senders don't overwhelm slow receivers (flow control)

OSI Data Link Layer (2/2)

Observation: We also need to specify how a number of computers can **share** a common channel (i.e. wire) $\Rightarrow$ **medium access control sublayer** (MAC):

- Specifies how one out of several competing senders, is eventually allowed exclusive access to the wire
- Common approach 1: listen to each other; retreat when you hear someone else, and try again later
- Common approach 2: wait your turn by passing a **token** between all stations

Well-known protocols: Ethernet, token ring, token bus, FDDI

OSI Network Layer

Essence: Describes how routing (and congestion) is to be done. Mostly needed in subnets.

- How do we find out which computers/routers are in the network?
- How do we calculate the best route from A to B?
- What happens when a computer/router goes down?
- Should multicasting/broadcasting be supported?
- What happens if a router becomes overloaded and starts dropping packets?
- Can we detect and avoid “hot spots?”

OSI Transport Layer

Observation: Generally offers connection-oriented as well as connectionless services, and varying degrees of reliability. This layer provides the actual network interface to *applications*

- Often provides network interface through **sockets** (UNIX, Windows)
- Allows to set up a connection to another application, and subsequently deliver data **reliably**, and in the order that it was **sent**
- Often also support for **secure connections**
- Also support for **datagrams**: unreliable message passing on a per-message basis

OSI Session and Presentation Layers

Session: tells how applications can set up "long-lasting" communications, for example, allowing a connection to be re-established when suddenly broken (think of downloading large files).

Presentation: Describes everything that is needed to exchange data in a platform-independent way. Example: think of byte ordering in different computers, or passing "binary" data through e-mail

OSI Application Layer

Essence: Contains the rest

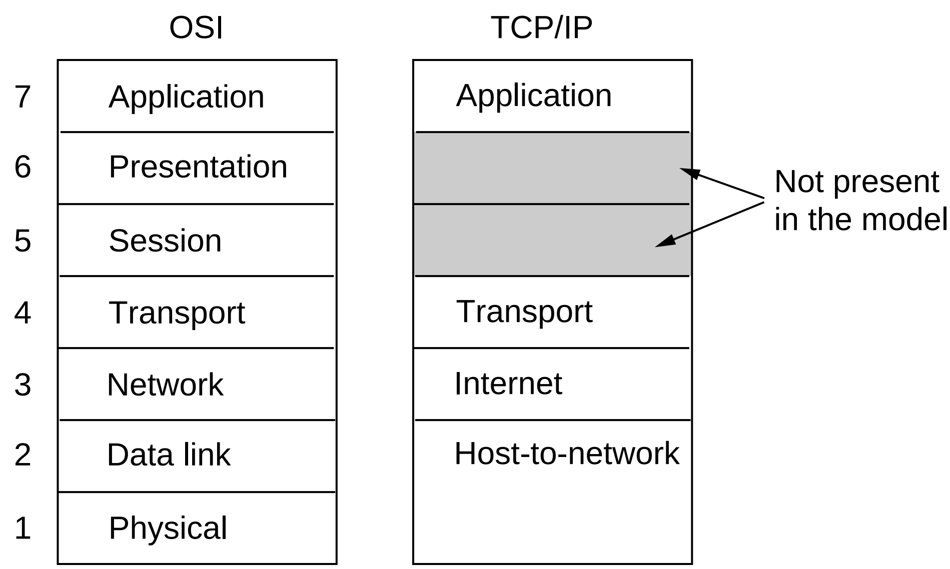
Traditional: Name services (DNS), security, e-mail (SMTP), News (NNTP), Web (HTTP)

Modern: All types of **middleware protocols** to support **distributed systems**:

- New transfer protocols for object systems like Java (RMI), CORBA (IIOP), DCOM (propriety)
- Special protocols to handle replication, fault tolerance, caching, data persistence, etc.

High-level protocols: Special application-level protocols for e-commerce, banking, EDI, etc.

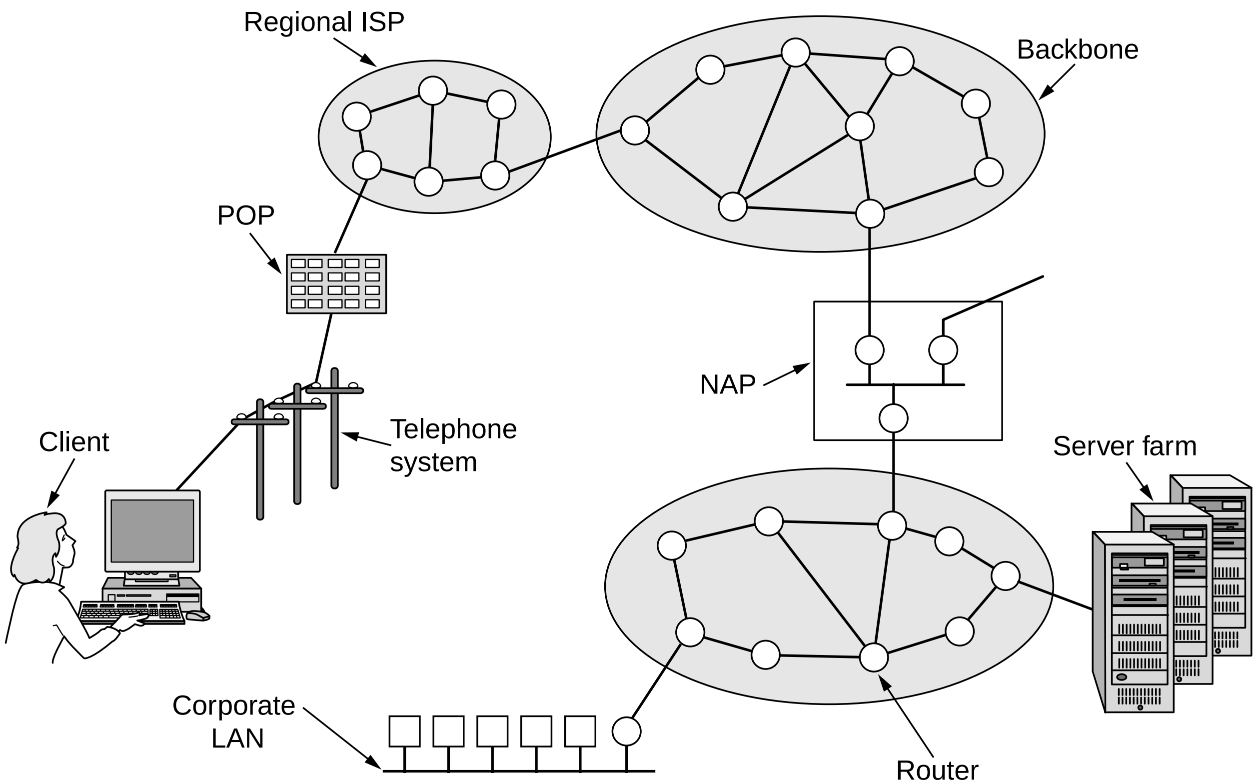
The TCP/IP Model



The bad thing is that TCP/IP did not make a clear distinction between services, interfaces, and protocols. That makes it much harder to re-implement certain layers.

TCP/IP protocol suite is successful because (1) it was *there* when needed (OSI implementations were terrible), (2) freely distributed with the UNIX operating system.

Example: The Internet



ISP	Internet Service Provider
POP	Point Of Presence – interface between telephone system and ISP's network
NAP	Network Access Point – connection point to different backbones

Connection-Oriented Networks

Observation: As long as carriers rule the world (i.e. they have the cables), and competition increases, carriers will increase and improve their data communication services for the public:

SMDS: Switched Multimegabit Data Service primarily intended to connect a number of LANs through long-haul networks (owned by the carrier)

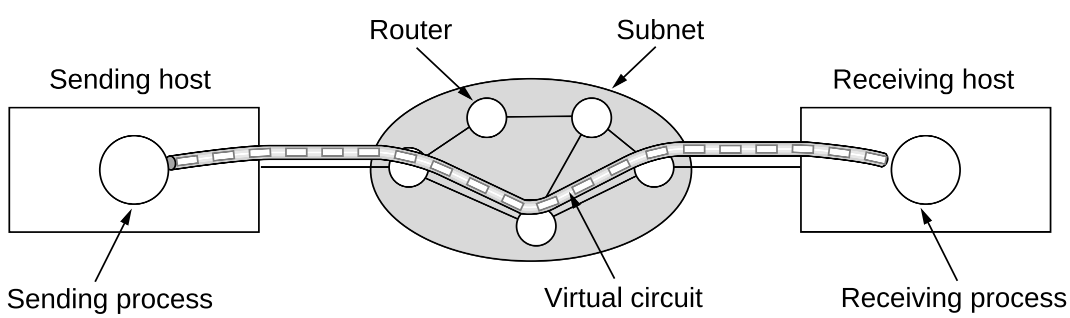
X.25: The OSI network protocol (also covering datalink and physical layer), intended to offer a *data* network on top of an (existing) cable infrastructure. Pretty old.

Frame relay: Extremely simple facility that allows a customer to hire a single high-bandwidth link. There is hardly any support for error detection, routing, flow control, etc.

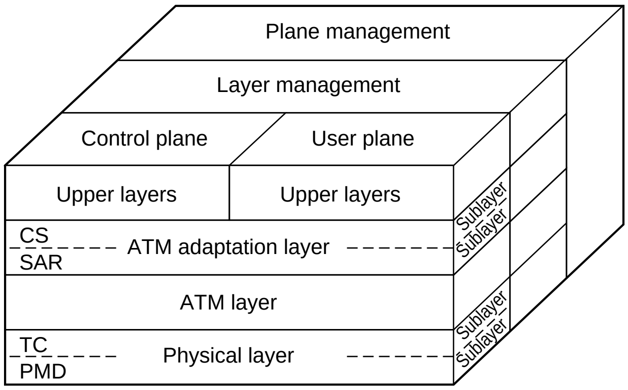
ATM (1/2)

Problem: How can telcos really offer high-speed connections?

Solution (they thought): Asynchronous Transfer Mode (ATM), by which cells of 53 bytes (!) are sent along **virtual circuits** from sender to destination.



ATM (2/2)



CS: Convergence sublayer
SAR: Segmentation and reassembly sublayer
TC: Transmission convergence sublayer
PMD: Physical medium dependent sublayer

OSI layer	ATM layer	ATM sublayer	Functionality
3/4	AAL	CS	Providing the standard interface (convergence)
		SAR	Segmentation and reassembly
2/3	ATM		Flow control Cell header generation/extraction Virtual circuit/path management Cell multiplexing/demultiplexing
2	Physical	TC	Cell rate decoupling Header checksum generation and verification Cell generation Packing/unpacking cells from the enclosing envelope Frame generation
1		PMD	Bit timing Physical network access

Observation: This model does not fit into OSI's layered approach: ATM assumes there's a separate signaling network to setup a connection