Freie Universitat Berlin
Erasmus Program

10 ECTS

Telematics

Professor:
Prof. Dr. Ing. Jochen Schiller

Autor:
Filippo Ghirardini

Winter Semester 2024-2025

Contents

Telematics

Autor: Ghirardini Filippo

Winter Semester 2024-2025

1 Basics

1.1 Network composition

A network consists of three elements:

- **End systems**: can vary in size and usage
- **Intermediate systems**: these (e.g. routers) are the components that allow the network to work.
- **Links**: they connect the end systems and can be *optical*, *copper* or *wireless*. Even if wireless is becoming more and more important, cables are still fundamental (undersea cables, underground cables).

Question 1.1.1 (Why fiber optic?)**.** Because this medium has not reached it's maximum and still has a huge potential of **bandwidth**. Also, while copper cable start acting as an **antenna** (and a receiver), disturbing near copper cable, fiber optic doesn't have this problem. Furthermore, copper cables need amplifiers which increase **latency**.

Question 1.1.2 (Why copper?)**.** It's **cheaper** and **easier** to handle.

Question 1.1.3 (Why cables over wireless?)**.** Because of **stability** and **latency**. Usually the problem is tampered by buffers but obviously it doesn't work with interactive applications.

Question 1.1.4 (What are threats to cable?)**.**

1.2 Communication principles

There are two basics principles:

- **Synchronous**: joint action of sender and receiver. Requires **waiting** until all parties are ready (e.g. phone calls)
- **Asynchronous**: sender and receiver operate decoupled (e.g. SMS, email). Requires **buffering**.

Note 1.2.0.1. There is also **isochronous**, which means the messages are sent every predetermined amount of time.

1.2.1 Direction

Communication channels may allow traffic flow in different directions:

- **Simple duplex**: one direction
- **Half-duplex**: both directions in different moments
- **Full-duplex**: both directions at the same time

1.2.2 Distribution

The communication distribution can happen in different ways:

- **Unicast**: one to one
- **Broadcast**: one to all
- **Multicast**: one to a subset
- **Anycast**: one to the nearest, e.g. when requesting to a redundant database you don't care which one responds
- **Concast**: many to one, e.g. we collect sensor data and send it to one
- **Geocast**: one to a certain region

Note 1.2.2.1. Even if multicast would be easier and cheaper, companies usually go for unicast because they want to know who the clients are.

Note 1.2.2.2. Broadcast guarantees anonymity while multicast does not.

1.2.3 Topologies

The main topologies are:

- **Full mesh**: too expensive
- **Chain**: in cars and trains
- **Star**: ideal for switches
- **Partial mesh**: the best compromise
- **Tree**: not ideal for big networks since if you cut a side, you lose contact

1.3 Sharing

1.3.1 Cons

Sharing may create a lot of problems, like **bottlenecks**: links and intermediate nodes are shared between end systems. One solution may be to *reroute* or to start *dropping packets* (e.g. when streaming the resolution lowers down).

1.3.2 Pros

At the same time, sharing means more efficient (less expensive) mechanism to **exchange data** between different components of distributed systems and **minimize blocking** due to multiplexing.

1.3.3 How?

There are two possible ways of sharing:

- **Reservation**: you reserve in advance the resource so that it is guaranteed, e.g. remote surgery. When the peak demand and the flow duration varies, there are two options:

  1. *First Come First Served*
  2. Everyone gets $10Mbps$

  It is implemented with **circuit-switching**: establishes dynamically a dedicated communication channel. It has predictable performance and a simple and fast switching but it's inefficient for bursty traffic, complex to setup and not easily adaptive to failures.

- **On-demand**: when there is a resource available you take it (variable *delay*, **jitter**), e.g. email. It is implemented with **packet-switching**: splitting the resource in packets and multiplex them. Much more flexible but requires buffers, packets overhead and has unpredictable performances.

Osservazione 1.3.1. It all depends on the application. Each flow has a **peak rate** and an **average rate**.To decide if *reservation* works well for a specific case, we must look at the ratio $\frac{P}{A}$. If it's small then it works well, otherwise it's wasting resources.

1.4 Internet

Internet is a network of networks. It enables processes on different hosts to exchange data: it's a bit delivery system.
ISPs enable you to access and use Internet services: well defined and commonly required functions. There are two roles: **client** and **server** that can be on different machine (or not, like with P2P).

Definition 1.4.1 (Internet)**.** *The set of all reachable parties (IP addresses).*

1.5 Protocols, layer and standards

Definition 1.5.1 (Digital Data Communication)**.** *Processing and transport of digital data between interconnected computers.*

Definition 1.5.2 (Data)**.** ***Representation*** *of facts, concepts and statement in a formal way which is suitable for communication, interpretation and processing by human beings or technical means.*

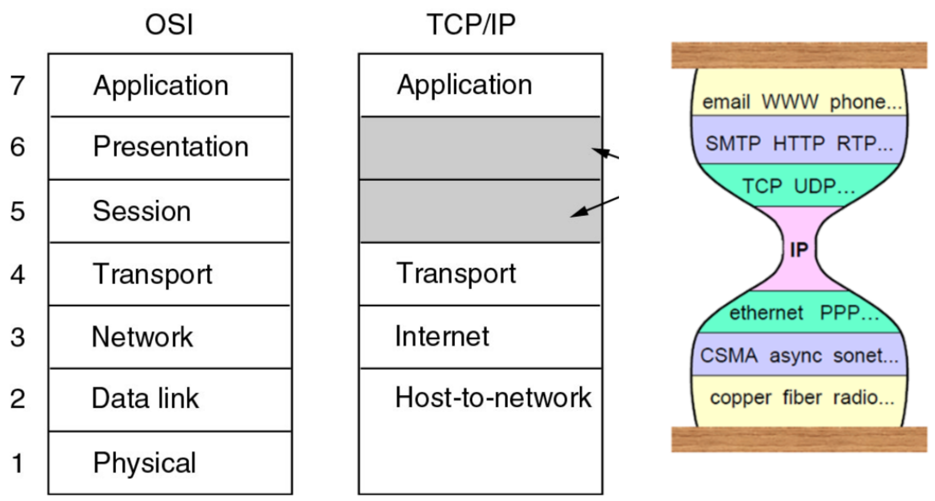
Note 1.5.0.1. **Information** is different from the data.

Definition 1.5.3 (Signal)**.** *A signal is the physical representation of data by spatial or timely variation of physical characteristics.*

In this context we need rules of communication for the different devices to communicate: we have heterogeneous computer architectures, network infrastructure and distributed application. A **protocol** is a conversational convention, consisting of syntax and semantic.

Definition 1.5.4 (Protocol)**.** *Protocols define format, order of messages sent and received among network entities, and actions taken on message transmission and receipt.*
A protocol is a set of ***unambiguous*** *specifications defining how processes communicate with one another through a connection (wire, radio etc.).*

To provide structure to the design of networks protocols, network designers organize protocols in **layers**. We use two models, either the ISO/OSI or the TCP/IP.



All the different layers need additional information, which is added via **headers** to the data payload via **encapsulation**. This could cause a lot of **overhead**.
All the protocols rely on the Internet Protocol. This maximizes **interoperability** and does not ensure anything, therefore no one has expectations.

1.6 Quality of service

To define the quality of communication we check:

- **Technical performance**: delay, jitter, throughput, data rate, etc.
- **Costs**
- **Reliability**: fault tolerance, system stability, immunity, availability
- **Security and Protection**: eavesdropping, authentication, denial of service, etc.

1.6.1 Latency

The main parameters we check are:

- **One-way delay**: measured in seconds

$$d_1 = t'_1 - t_1 \tag{1}$$

- **Round-trip-time**: measured in seconds

$$r_1 = t_2 - t_1 \tag{2}$$

  It should also integrate the processing time of the other device.

1.6.2 Stability

The main parameter that measures stability is the **Jitter**. It's measured in seconds and calculated using the delay:

$$d_i = t'_i - t_i \qquad j_i = d_{i+1} - d_i \tag{3}$$

1.6.3 Capacity

From a capacity perspective, we measure the **throughput** in $\frac{bit}{s}$ as follows:

$$T = \frac{\sum \text{data}_i}{\Delta t} \tag{4}$$

The **goodput** instead, is the amount of **useful** throughput from a user perspective.

Note 1.6.3.1. **Bandwidth** is used for the description of the channel characteristics.

There is also the **Delay-Throughput-Produc** which measures how much data can be on the medium itself while traveling. E.g. with a connection of $1Mbps$ that has $200ms$ of delay we have

$$1Mbps \times 0.2s = 200kbit$$