Redes de Comunicação em Ambientes Industriais
Aula 2

Luís Almeida lda@det.ua.pt

Electronic Systems Lab-IEETA / DET
Universidade de Aveiro
Aveiro, Portugal

Outline

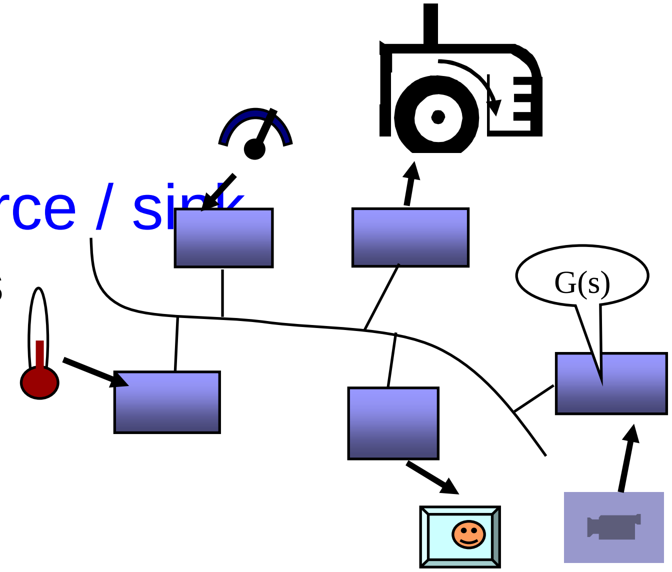
The network in a distributed system: general concepts

- ✓ **Application dommains**
- ✓ **Requirements**
- ✓ **Temporal behaviour merit figures**
- ✓ **Data exchange semantics**
- ✓ **Time and event triggering**

Background

Why distributed architectures

- Processing closer to data source / sink
  - Intelligent sensors and actuators
- Dependability
  - Error-containment within nodes
- Composability
  - System composition by integrating subsystems
- Scalability
  - Easy addition of new nodes with new or replicated functionality
- Maintainability
  - Modularity and easy node replacement
  - Simplification of the cabling



Background

Distributed architectures became pervasive in many real-time application fields, e.g.:

- ✓ **process control** and **factory automation** (manufacturing cells, robots)
- ✓ **transportation systems** (planes, trains, trucks, buses, cars)
- ✓ **multimedia systems** (remote surveillance, industrial monitoring, video on demand)

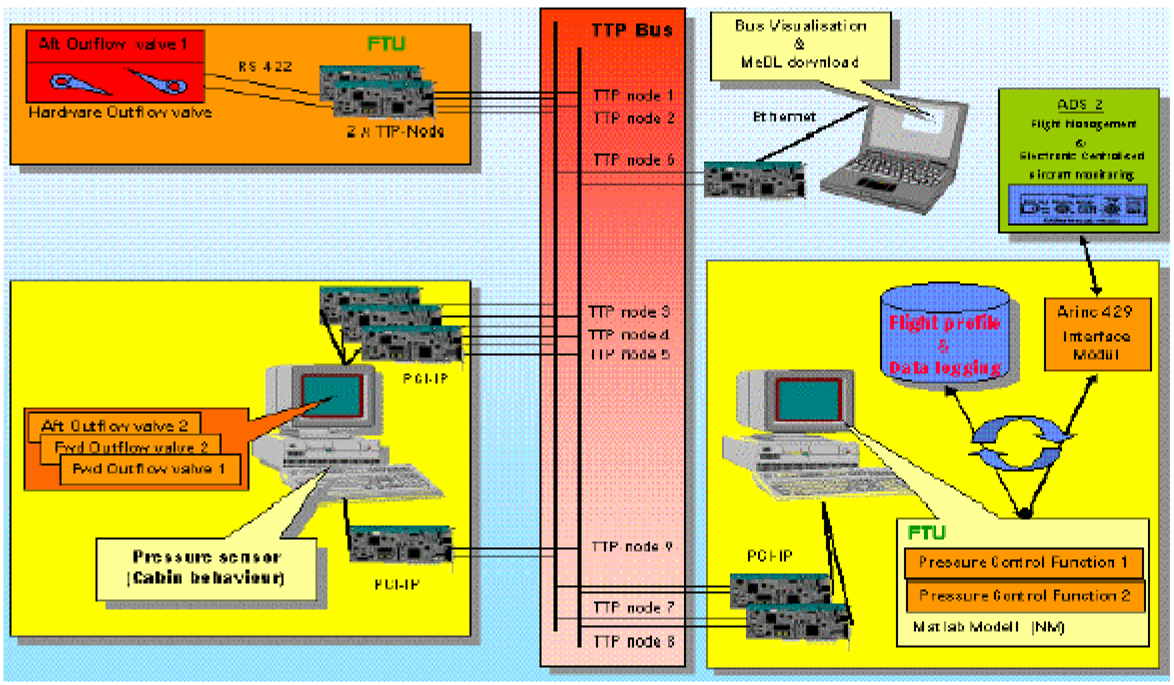
In many cases with **critical timeliness** and **safety** requirements

Background

Today, there are **many different networks** with real-time capabilities, aiming at different application domains e.g.:

- ✓ ARINC629, SwiftNet, SAFEbus, TTP – **avionics**
- ✓ WorldFIP, TCN – **trains**
- ✓ CAN, TT-CAN, TTP, FlexRay – **cars**
- ✓ PROFIBUS, WorldFIP, P-Net, DeviceNet, FF-HSE/H1, Ethernet/IP, PROFINET, Ethernet Powerlink – **automation**
- ✓ Firewire, USB – **multimedia**

Background



Avionics

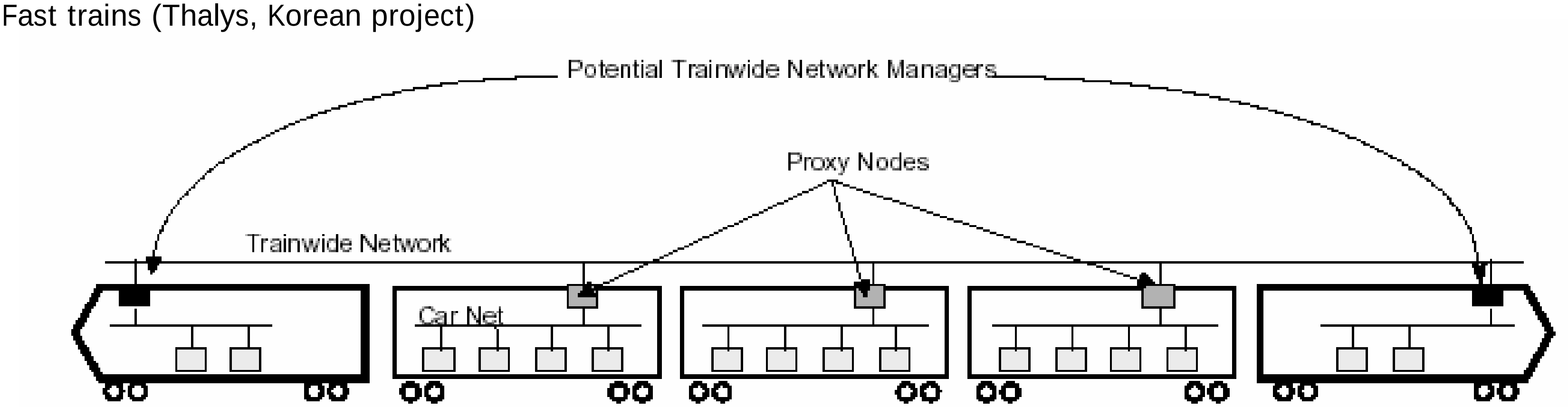
from the SETTA project

Background

Fast trains (Thalys, Korean project)

Automatic trains (Frankfurt Airport)

Train control systems



(LONworks applications)

Background

<u>VW Phaeton:</u>

→ **11.136 electrical** parts in total

→ **61 ECUs** in total

→ external diagnosis for 31 ECUs via serial communication

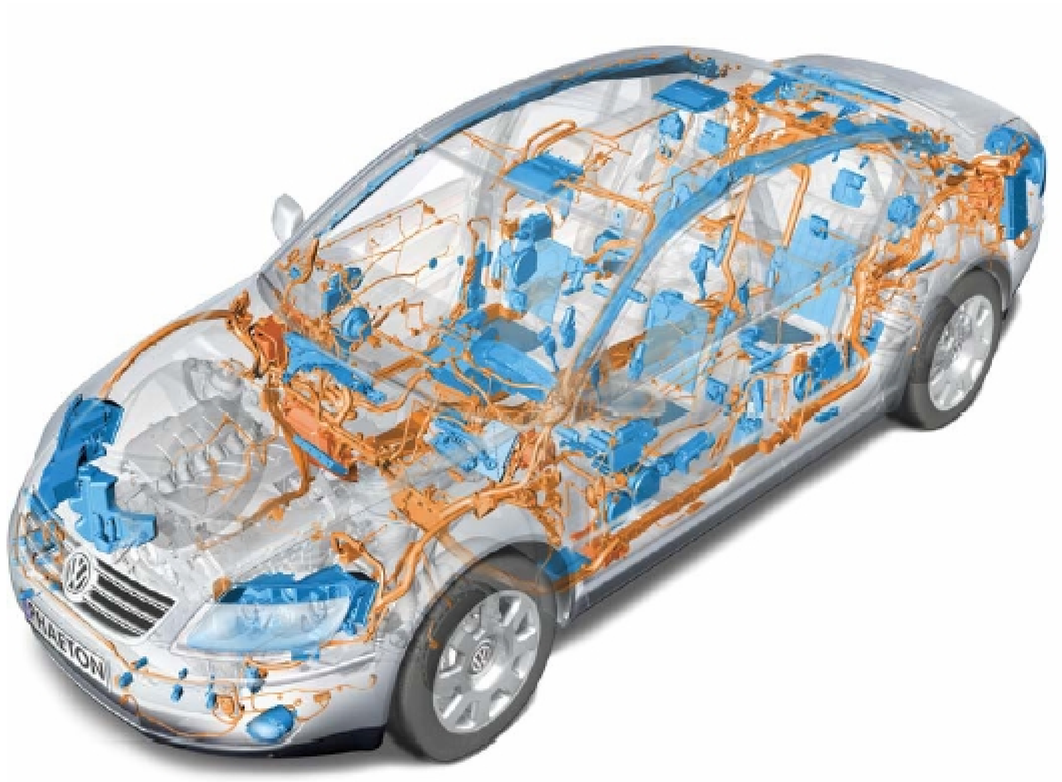
→ **optical bus** for high bandwidth Infotainment-data

→ sub-networks based on proprietary serial bus

→ **35 ECUs** connected by **3 CAN-busses** sharing:

→ appr. **2500 signals**

→ in **250 CAN messages**

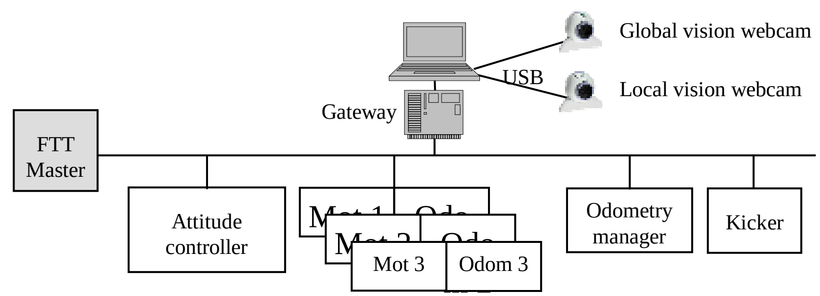


The VW Phaeton

Adapted from (Loehold, WFCS2004)

Background

The CAMBADA robotic soccer team



→ **9 ECUs** connected by **1 CAN-bus** conveying:

→ **9 periodic** messages

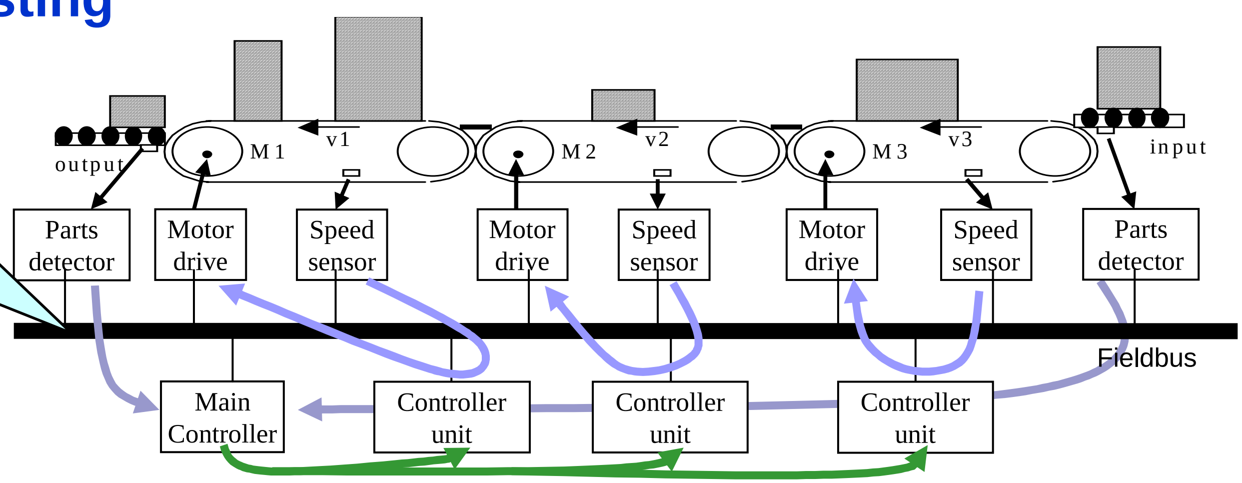
→ **12 aperiodic** messages

HW control architectures for mobile robots

Typical service requirements

- ✓ In these applications the network must support
  - ✓ Efficient transmission of **short data** (few bytes)
  - ✓ **Periodic** transmission (*monitoring, feedback control*) with **short periods** (ms) and **low jitter**
  - ✓ **Fast** transmission (ms) of **aperiodic** requests (*alarms*)
  - ✓ Transmission of **non-real-time data** (*configuration, logs*)
  - ✓ **Multicasting**

•Short data
•Periodic & aperiodic,
•Single broadcast domain



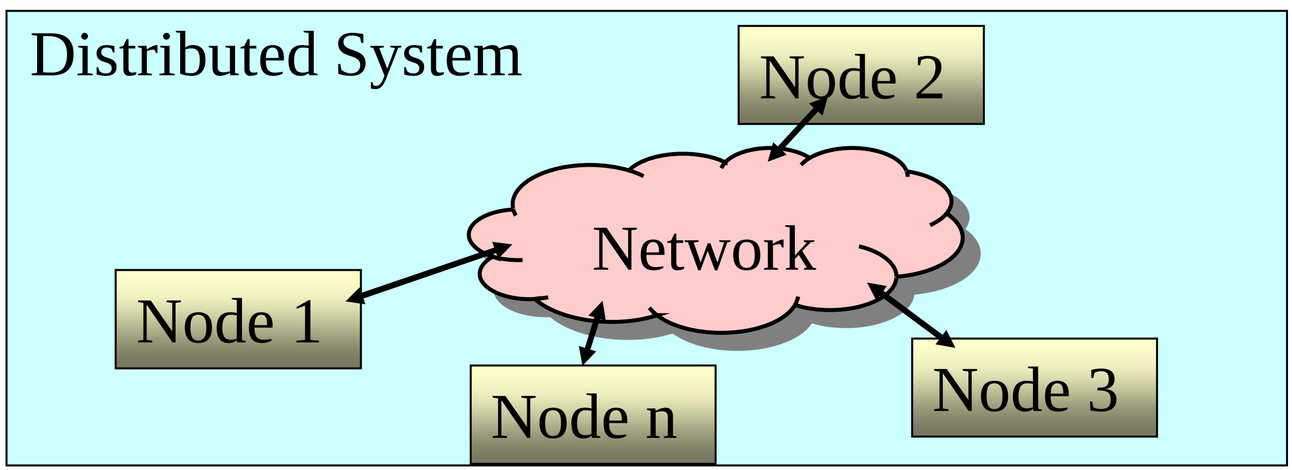
How difficult is it
meeting these requirements?

How to build networks that meet them?

How to verify that
the requirements are met?

The network in a DS

- The network is a **fundamental component** in a distributed system (DS) **supporting all** the **interactions** among nodes
- Thus, it is also a **critical resource** since loss of communication, results in the loss of all global system services



But what is a network?

- **Physical infrastructure** ?
  - Wiring
  - Connectors
  - Networking equipment

OR

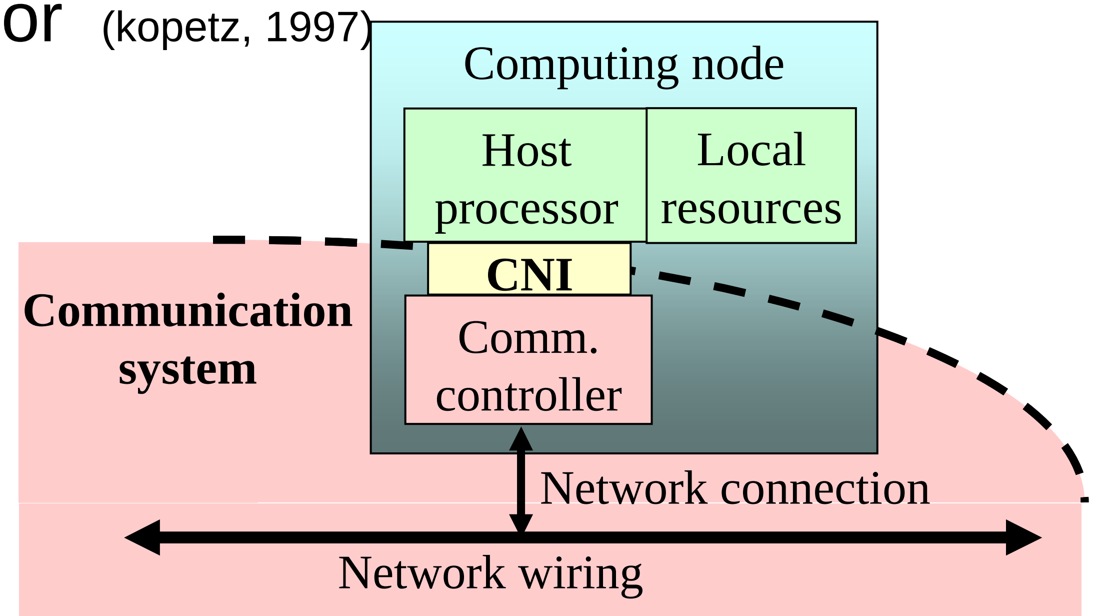
Preferred definition for "networkers"

- **Communication system** ?
  - Physical infrastructure **plus**
  - Network access interfaces
  - Protocol stack

Supports a system-wide vision that facilitates reasoning about the interplay between all communicating nodes.

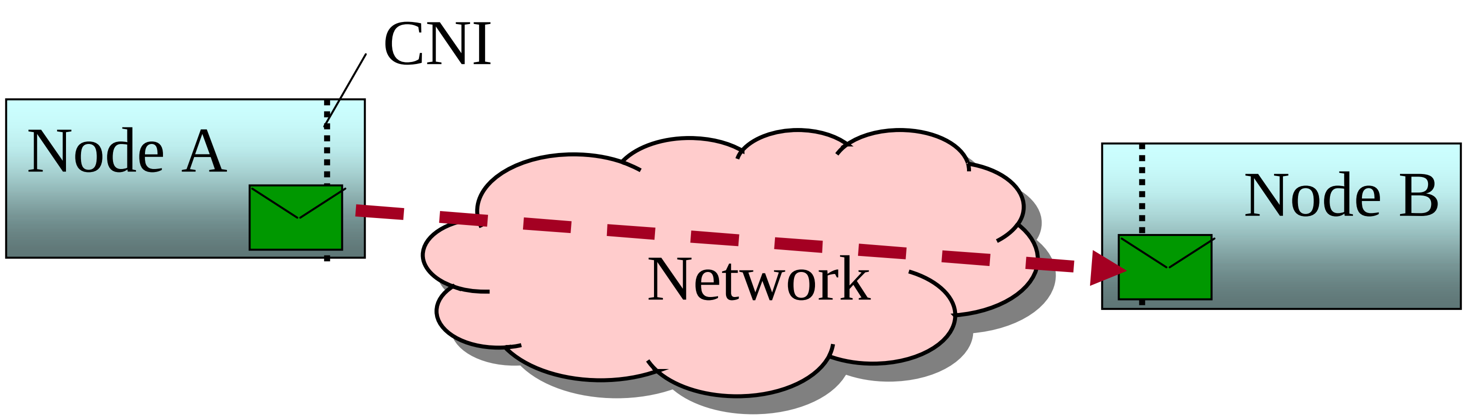
Network interfaces

- ✓ The network extends up to the **Communication Network Interface (CNI)** that establishes the frontier between communication system and the node host processor (kopetz, 1997)



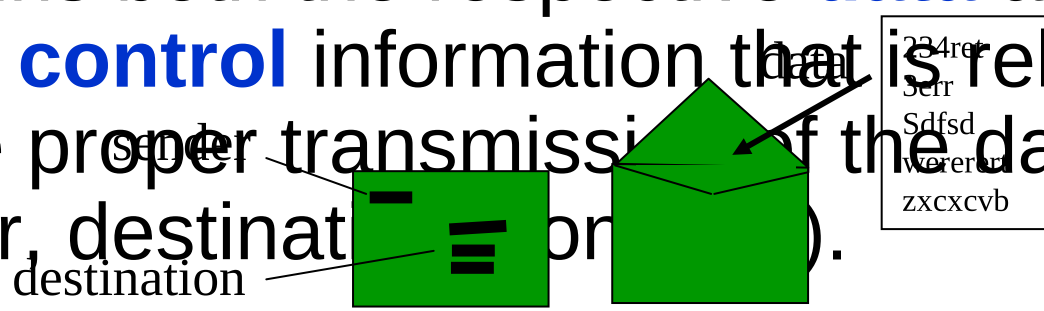
Purpose of the network

- ✓ To deliver **communication services** to the requesting nodes (i.e., to transport messages from the CNI of a sender node to the CNI of a receiver node) reliably, securely, efficiently and (for a real-time network) **timely**



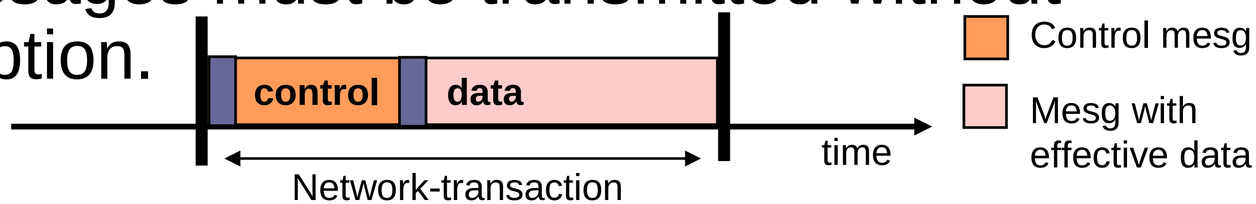
Messages

- **Interactions** are supported by **message passing**
- A **message** is a **unit of information** that is to be transferred, at a given time, from a sender process to one or more receiver processes
- Contains both the respective **data** as well as the **control** information that is relevant for the proper transmissi of the data (e.g. sender, destinat or).



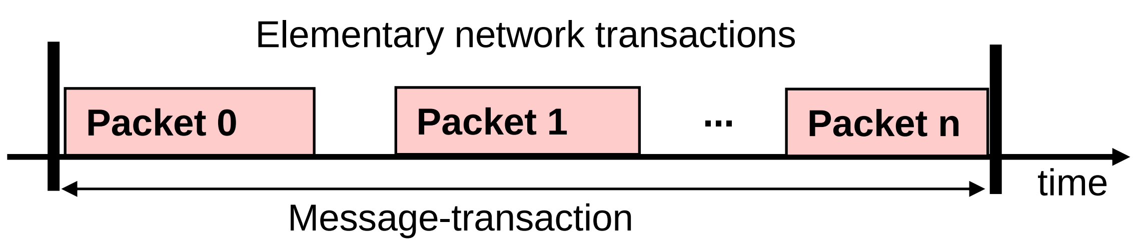
Transactions

- A **network transaction** is the **sequence of actions** within the communication system required to accomplish the **effective transfer** of a message's data.
- A transaction might include the transfer of messages carrying protocol control information, only. These are referred to as **control messages**.
- A multi-message transaction is **atomic** when all its messages must be transmitted without interruption.



Transactions

- Many networks automatically break large messages in smaller packets (fragmentation/reassembly).
- In that case, a **message transaction** includes several **elementary network transactions** that correspond to the transfer of the respective packets.
- A **packet** is the smallest unit of information that is transmitted without interruption (when there is no risk of confusion we will use **message** and **packet** interchangeably).



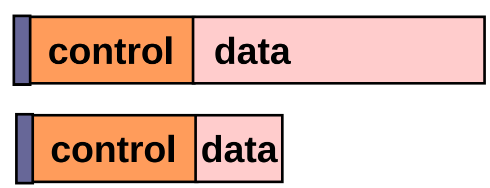
Transactions

- The **data efficiency** of the network protocol can be defined as the **ratio** between the time to transmit **effective data** bits and the **total duration** of the respective transaction.

$$\text{Data\_eff} = \frac{\text{data tx\_time}}{\text{transaction duration}}$$

- In general:

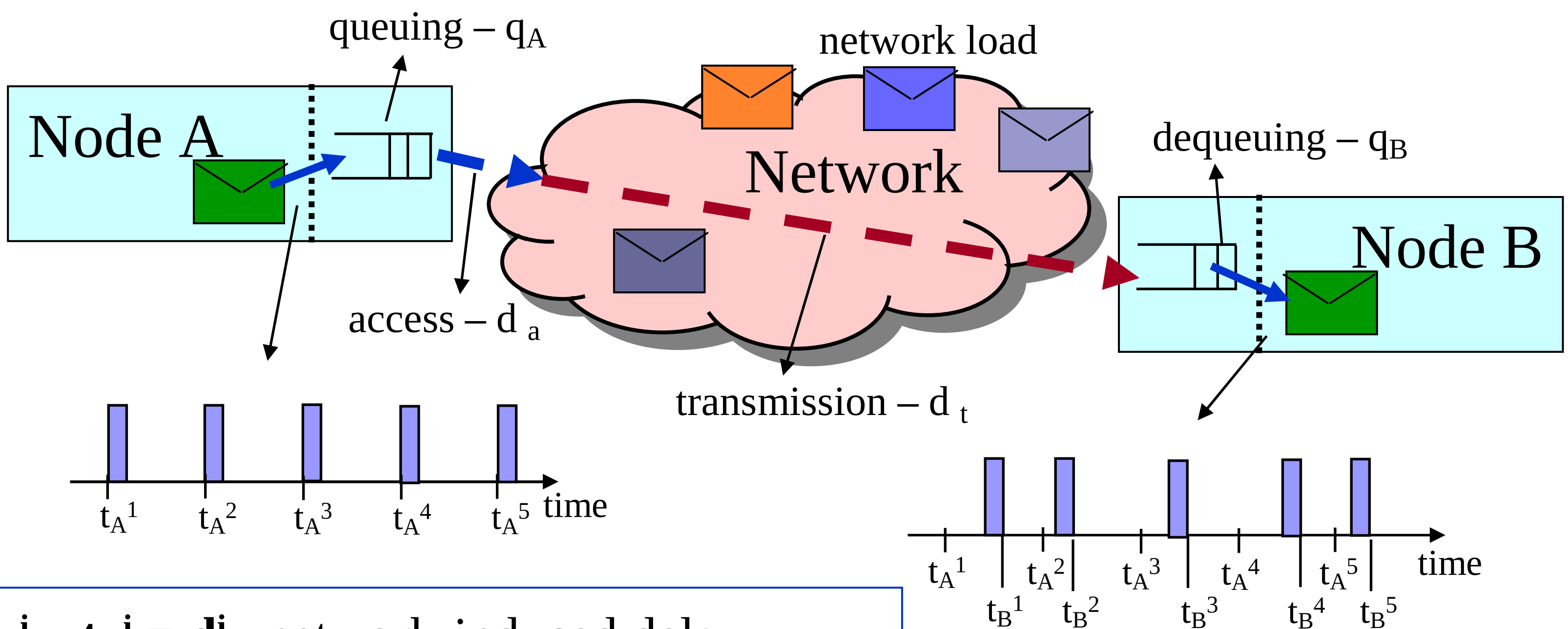
  The **shorter** the data per transaction,
  the **lower** the efficiency is



Timing figures

- Typical figures of merit about the temporal behavior of the network:
  - **Network induced delay** – extra delay caused by the transmission of data over the network. Some applications (e.g. control) are particularly sensitive to this delay
  - **Delay jitter** – variation affecting the network induced delay. Some applications (e.g. streaming) are not much affected by the network delay but are highly affected by delay jitter
  - **Buffer requirements** – when the instantaneous transmission from a node is larger than the capacity of the network to dispatch it, the traffic must be held in buffers. Too few buffers lead to packet losses

Timing figures



$\mathbf{t_B^i - t_A^i = d^i}$, network-induced delay

$\mathbf{d^i = q_A^i + d_a^i + d_t^i + q_B^i}$, delay components

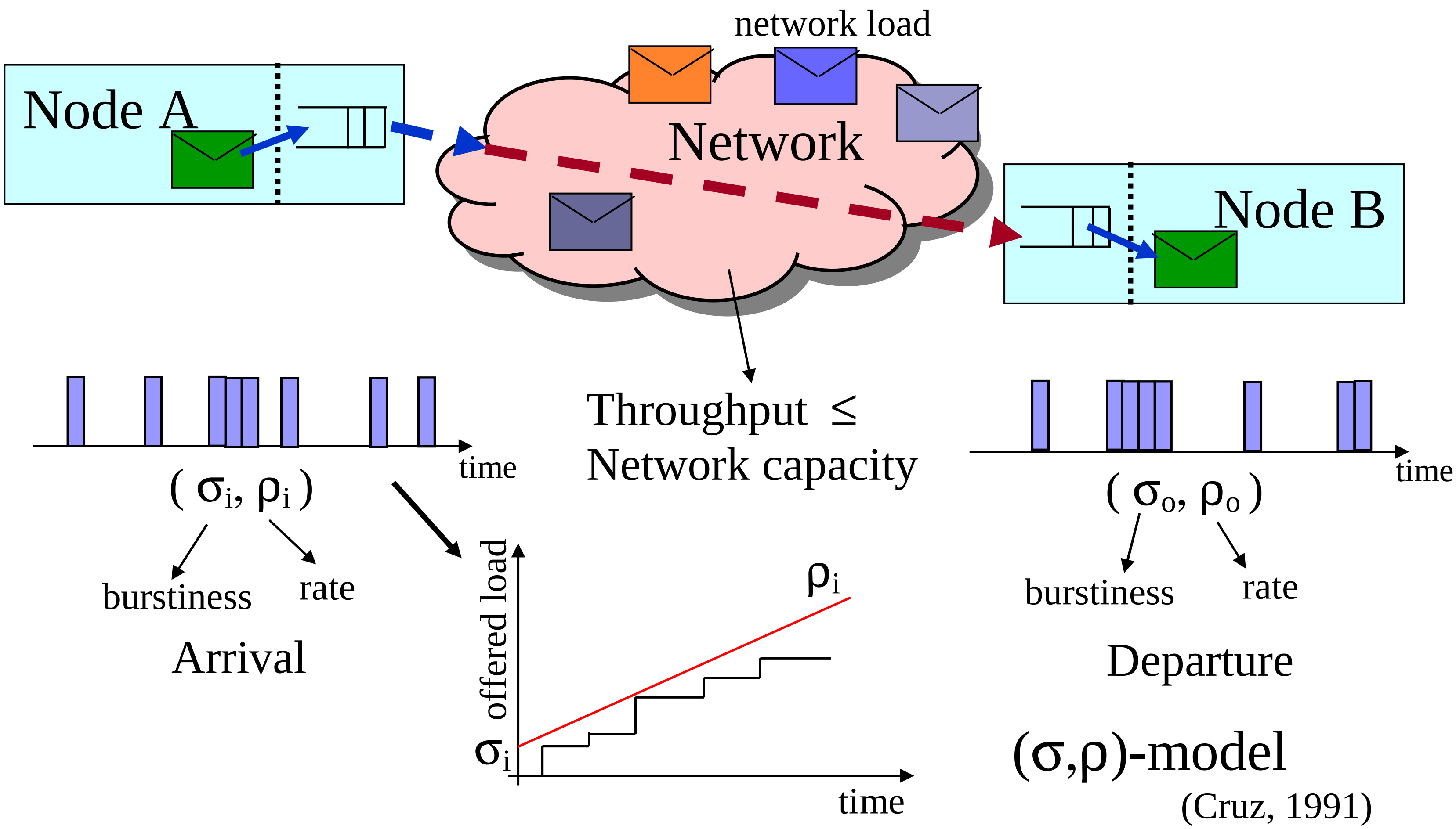
$\mathbf{d^i - d^{i-1} = j^i}$, delay jitter

Reception instants may suffer irregular delays due to interferences from the network load, queuing policies and processor load

Timing figures

- Other relevant figures of merit:
  - **Throughput** – amount of data, or packets, that the network dispatches per unit of time (bit/s and packets/s).
  - **Arrival / departure rate** – rate at which data arrives at/from the network (bit/s and packets/s).
  - **Burstiness** – measure of the load submitted to the network in a short interval of time. Bursts have a profound impact on the real-time performance of the network and impose high buffering requirements.
    File transfers are a frequent cause of bursts.

Timing figures



(σ,ρ)-model

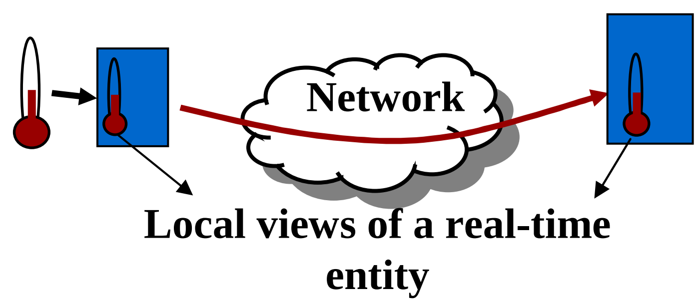
(Cruz, 1991)

Real-time messages

- ✓ A message related to a *real-time entity* (e.g. a sensor) is a **real-time message**.

- ✓ Real-time messages must be transmitted within precise **time-bounds** to assure **coherence** between senders and receivers concerning their **local views** of the respective real-time entities

- ✓ Real-time messages can have **event** or **state semantics**



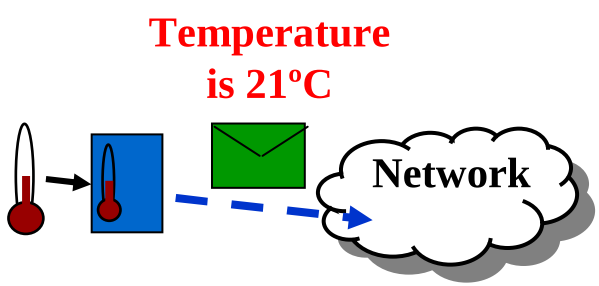
Real-time messages

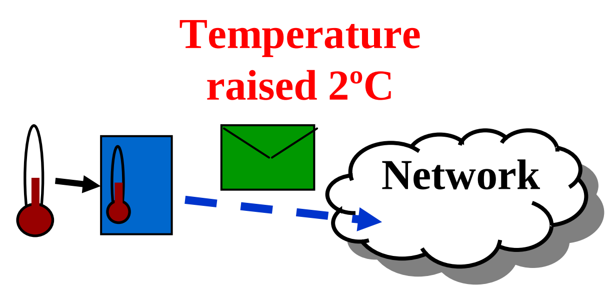
- **Events** are perceived changes in the system state.
  All are significant for **state consistency** across sender and receiver
- **External events** refer to the environment outside the computing system (normally asynchronous)
- **Event messages** must be **queued** at the receiver and **removed** upon reading. **Correct order** in delivery must be enforced
- **State messages** (containing state data) can be read many times and overwrite the values of previous messages concerning the same real-time entity.

Event or Time triggering

- According to the type of messages (event or state) conveyed by the network, it can be
  - **event-triggered** (event messages)

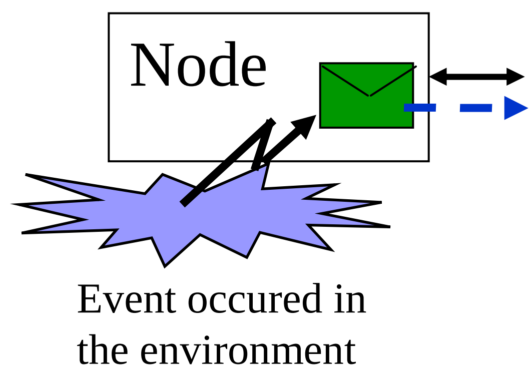
    or
  - **time-triggered** (state messages)



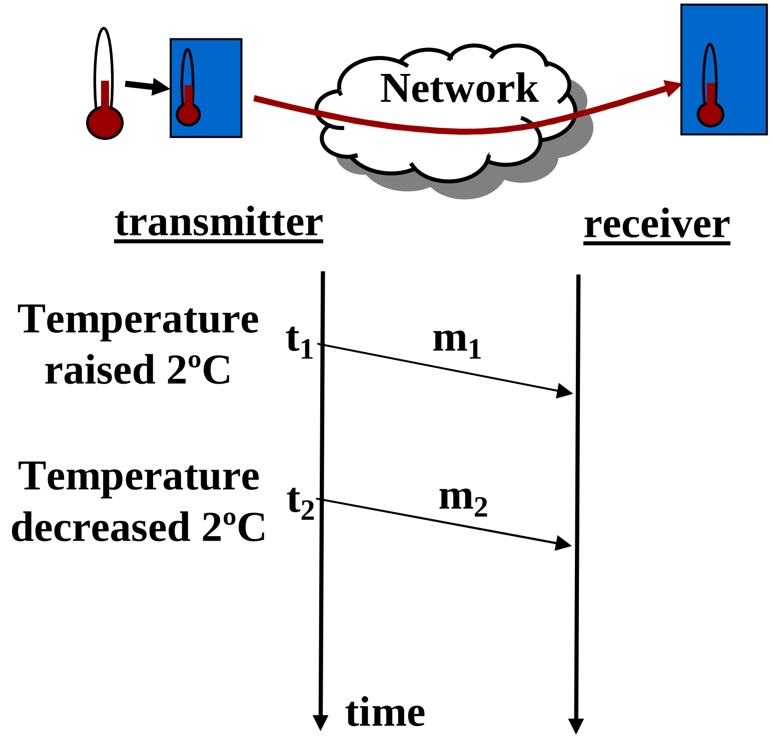


Event triggered network

- Transactions, carrying event data, are **triggered** upon **event occurrence**.
- Consequently, transmission instants are generally asynchronous wrt the nodes or network.
- Receivers know about **system state** by means of the **received events**.
- The submitted communication load (number of *simultaneous* message transmission requests) may be very high (**event showers**).



Event triggered network



What if:

m_2 is lost?

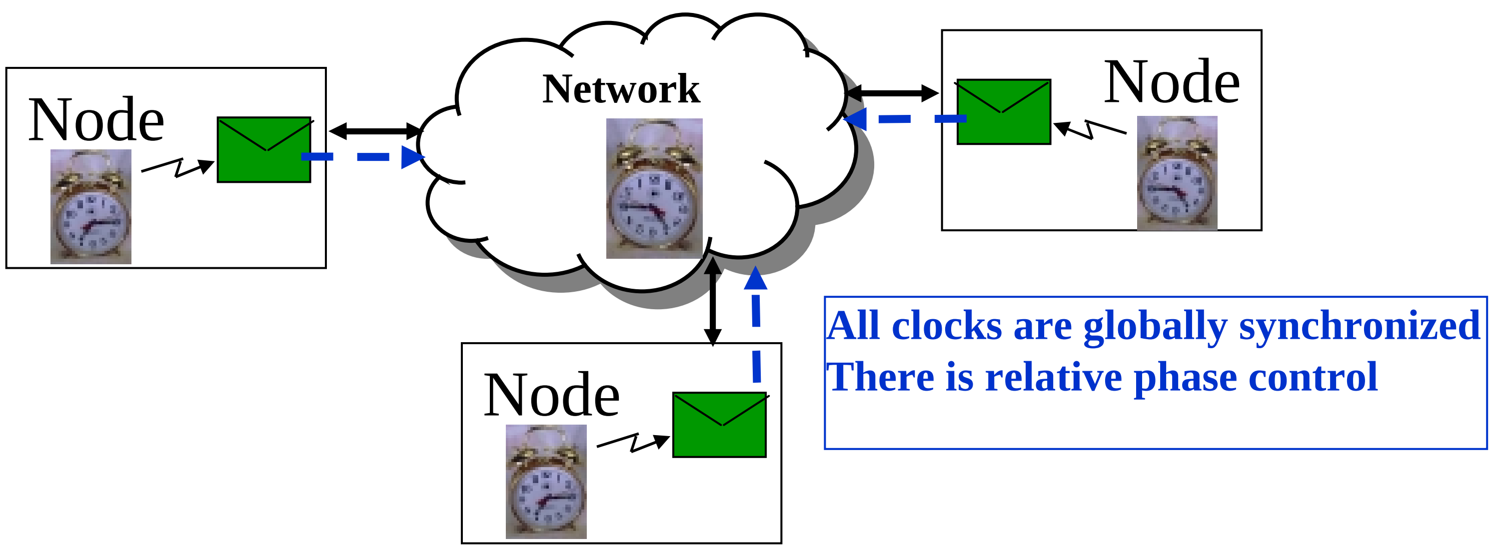
Notice that:

$\Delta t = t_2 - t_1$ is unbounded

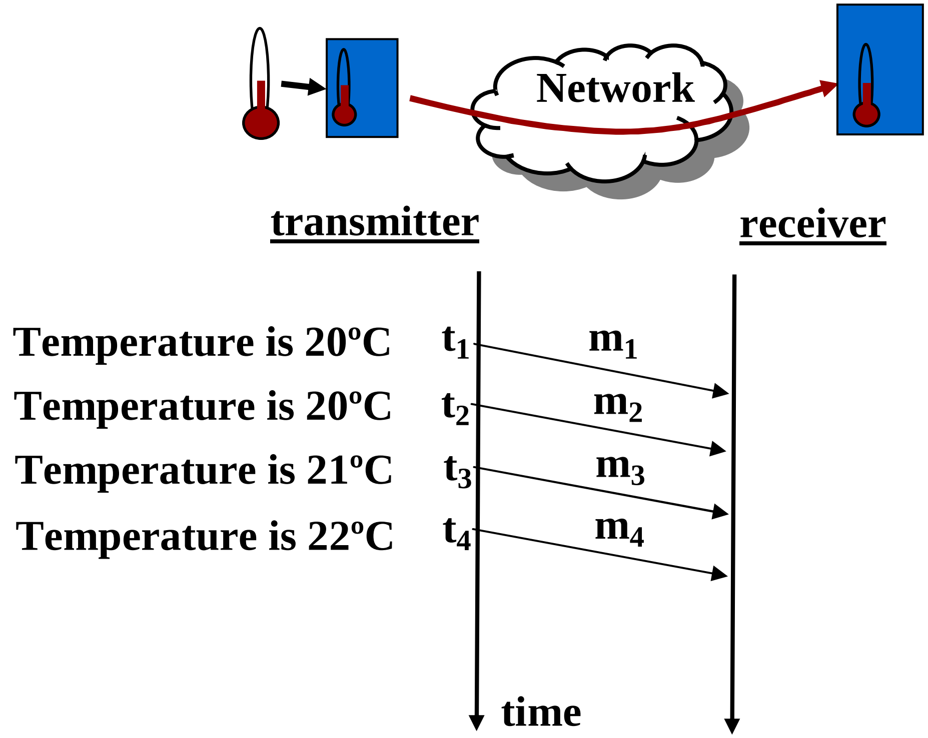
Typical solutions involve limiting Δt (e.g. with heartbeat)

Time triggered network

- ✓ There is a notion of **network time** (explicit or implicit)
- ✓ Transactions, carrying **state data**, are **triggered** at **predefined time instants**.
- ✓ Receivers have a **periodic refresh** of the system state
- ✓ The submitted communication load is well determined.



Time triggered network



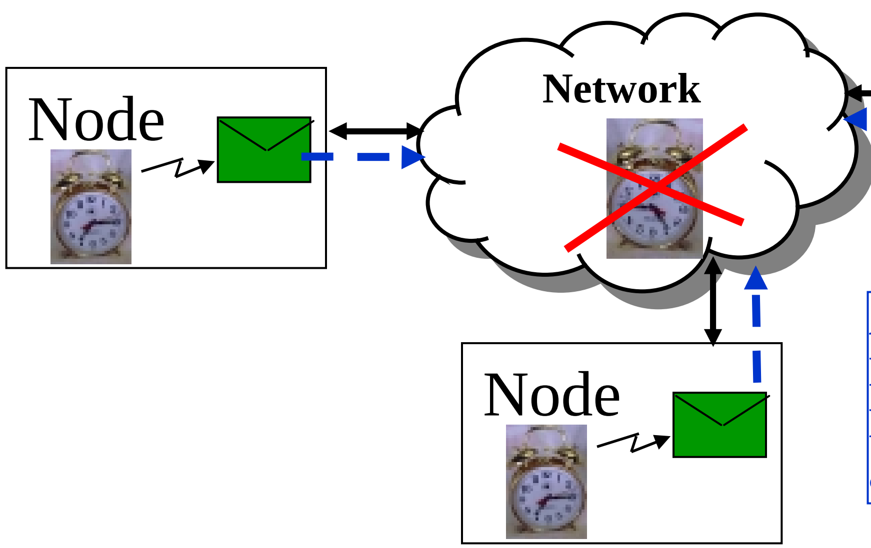
Notice:

$\Delta t = t_i - t_{i-1}$ set according to system dynamics

Losing one message causes inconsistency during Δt

Timer-driven triggering

- ✓ Network transactions can be triggered by a **local timer** without a global notion of time (thus asynchronous wrt the network access). This is known as the **timer-driven model** exhibiting properties between ET and TT (it is basically an ET model with burst control)   (Verissimo, 1996)



All clocks are local
No relative phase control
But communication bursts are bounded

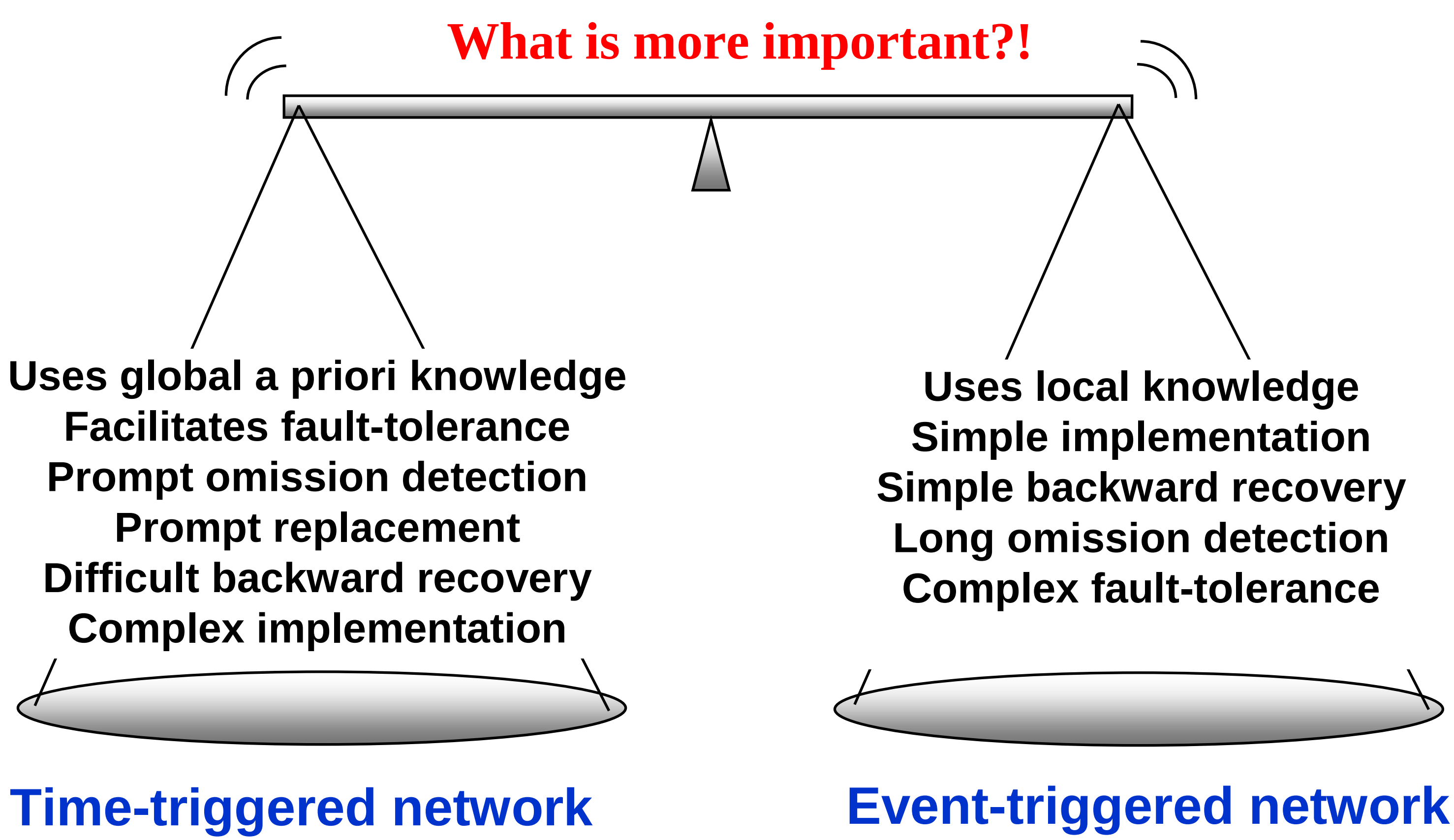
Event vs time triggering

- **Time-triggered network**
  - There is **more knowledge about the future**
    - Transmission instants are predefined
  - It is **easier to design fault-tolerance mechanisms**
    - Low omission detection latency
    - Simple management of spatial replication
  - However, there is **less flexibility to react** to errors
    - Retransmissions are often not possible because the traffic schedule is fixed
    - Without spatial replication, **latency to recover** from an omission is **normally long** (about one period of the message stream)
  - The communication **protocol is substantially complex** because of the amount of global a priori knowledge

Event vs time triggering

- **Event-triggered network**
  - There is **few knowledge about the future**
    - Events can occur at any time
  - More complex **fault-tolerance mechanisms**
    - Use of failure detectors, e.g. using **heartbeats** (maximum inter-transmission or silence time)
    - Omission detection takes about one heartbeat period
    - Replication is harder to manage (replicas may arrive at different instants in each channel)
  - However, there is **high flexibility to react** to errors
    - Retransmissions can usually be carried out promptly (or at least tried…)
    - Without spatial replication, **latency to recover** from an omission can be **very short** (time for one retransmission)
  - The communication **protocol is normally simple** to deploy, using local information, only, with few or no a priori knowledge

Event vs time triggering



What about both?

Summary:

- ✓ The **network** is the **kernel** of a **distributed system**
- ✓ Transmits **messages** by means of network **transactions**
- ✓ **Real-time** messages can be **event-** or **state-messages**
- ✓ Networks can be **event-triggered** or **time-triggered** leading to opposed characteristics wrt simplicity of the protocol implementation or the simplicity of enforcing fault-tolerance