Distributed Embedded Systems: An Introduction

Julián Proenza. University of the Balearic Islands. SPAIN

Luís Almeida. University of Aveiro. PORTUGAL

UIB Universitat de les Illes Balears

universidade de aveiro

The aim of this presentation

- Introduce the concept of Distributed Embedded System
- Discuss the role that the communication subsystem plays in this kind of systems
- Identify the main problems to address when designing a Distributed Embedded System
- Present the communication features specifically adapted to control applications

NOTE: The presentation is built on the assumption that the audience is familiar with the basic concepts of data communication networks

Presentation Outline

1. Distributed Embedded Systems
    1. From centralization to distribution
    2. Definition and advantages
2. The Communication Subsystem's role
    1. Two sources of problems: delays and unreliability
    2. Two disciplines: Real-Time and Dependability
3. Networks adapted to control applications
    1. The communication subsystem
    2. Requirements related to traffic
    3. Control networks and the OSI stack: structure and features

Presentation Outline

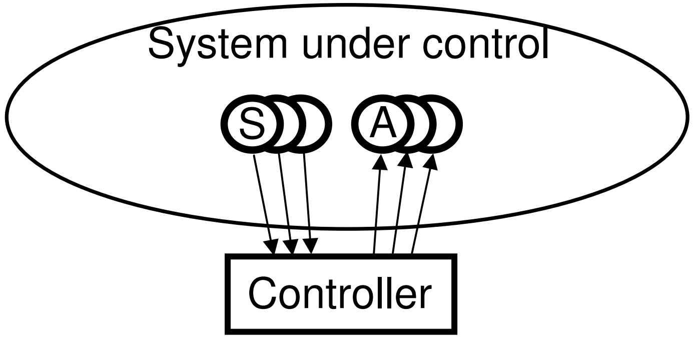
1. Distributed Embedded Systems
    1. From centralization to distribution
    2. Definition and advantages

2. The Communication Subsystem’s role
    1. Two sources of problems: delays and unreliability
    2. Two disciplines: Real-Time and Dependability

3. Networks adapted to control applications
    1. The communication subsystem
    2. Requirements related to traffic
    3. Control networks and the OSI stack: structure and features

Centralized Embedded Systems

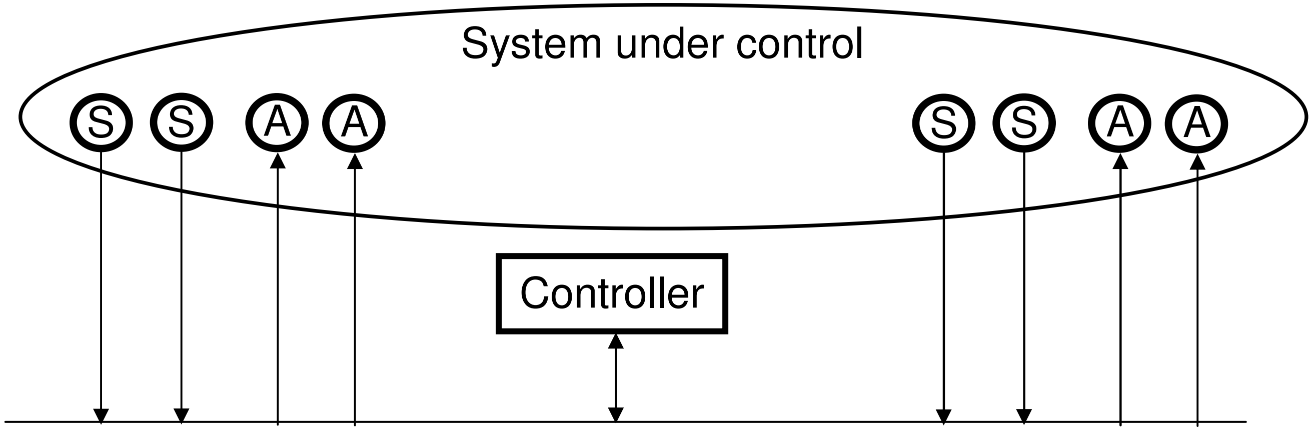
- Control application
- The controller (computer or PLC) reads some sensors and sends outputs to some actuators



- The system under control (plant) requires a specific time response: the controller has to be real-time
- The architecture is centralized: all sensors and actuators are connected through dedicated links.

Decentralized Embedded Systems

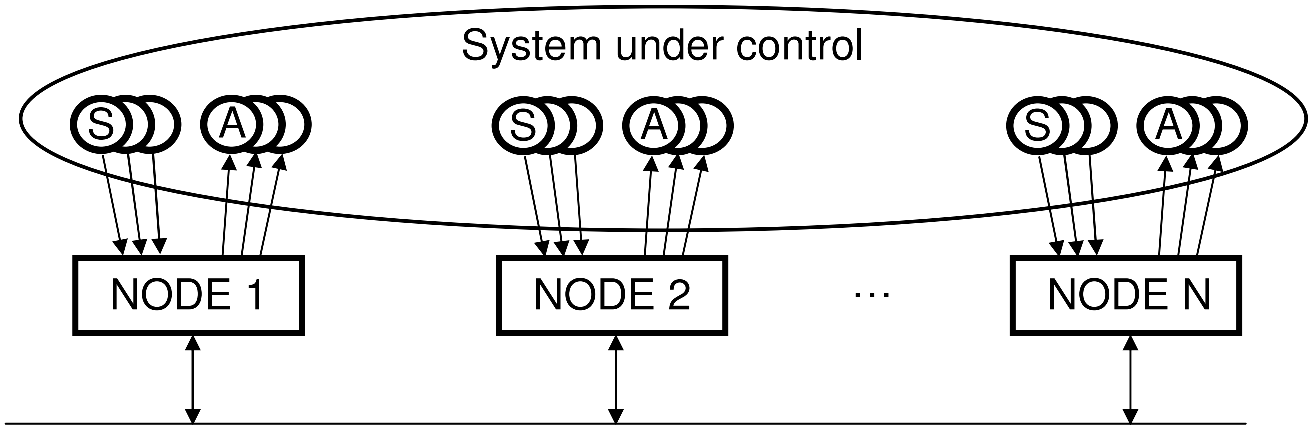
- In some applications, cabling started to be a problem: too costly (buildings) and too heavy (cars)
- Decentralized solutions were proposed



- In this kinds of systems, cabling is cheaper and simpler. Moreover addition of new s/a is easier
- In fact, for sharing the comm medium some comm controller had to be added to each s/a…

Distributed Embedded Systems

- … so it was natural to add some extra hardware and provide several controllers each one performing part of the control tasks



- The result is a **distributed control system**
- They are widespread in numberless applications: factories, buildings, vehicles, etc.

Presentation Outline

1. Distributed Embedded Systems
   1. From centralization to distribution
   2. Definition and advantages
2. The Communication Subsystem's role
   1. Two sources of problems: delays and unreliability
   2. Two disciplines: Real-Time and Dependability
3. Networks adapted to control applications
   1. The communication subsystem
   2. Requirements related to traffic
   3. Control networks and the OSI stack: structure and features

But what is a Distributed System?

- An **accurate definition**:
  - Multiple computers interconnected by a network that share some common state and cooperate to achieve some common goal. Essentially what was expressed by Schroeder in [Mul93]
- Some **remarkable definitions** by Leslie Lamport:
  - "A system is distributed if the message transmission delay is not negligible compared to the time between events in a single process." [Lam78b].
  - "A distributed system is the one that prevents you from working because of the failure of a machine that you had never heard of." Leslie Lamport
- Despite the differences between computing and control systems, it is clear we will have to solve some **problems**.

Advantages of distribution

- Processing closer to data source / sink
  - Intelligent sensors and actuators
- Dependability
  - Independence of failures among nodes
  - 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

Presentation Outline

1. Distributed Embedded Systems
    1. From centralization to distribution
    2. Definition and advantages

2. The Communication Subsystem's role
    1. Two sources of problems: delays and unreliability
    2. Two disciplines: Real-Time and Dependability

3. Networks adapted to control applications
    1. The communication subsystem
    2. Requirements related to traffic
    3. Control networks and the OSI stack: structure and features

The communication subsystem's role

- The communication subsystem is the **backbone**!
- Lamport's definitions pointed at **two key problems**:
  - Communication introduces **delay**
    - More delays to consider in order for the control to be timely
    - Inexact global state (In a system in which the events occur faster than the communications among computers, it is impossible to obtain an exact global state of the system from the information available at any computer in any circumstance).
  - Communication is **unreliable** (compared with the intra comms in a computer) and is a means for error propagation
    - A network is more prone to errors
    - If the network fails, the whole system fails
    - Errors in one computer may propagate to the other computers
- This subsystem has to be **adapted to control**

Overcoming the two key problems

- In order to solve the two key problems of extra delays and unreliability it is necessary to use concepts and techniques related to **two disciplines**:
  - Real-Time Systems and
  - Dependable Systems

- It is possible to provide a non-exhaustive list of the specific issues that each discipline has to deal with

- Aspects to take into account:
  - The set of specific issues to solve depends on the application
  - The properties achieved can/should be extended to the system as a whole

Issues solved by each discipline

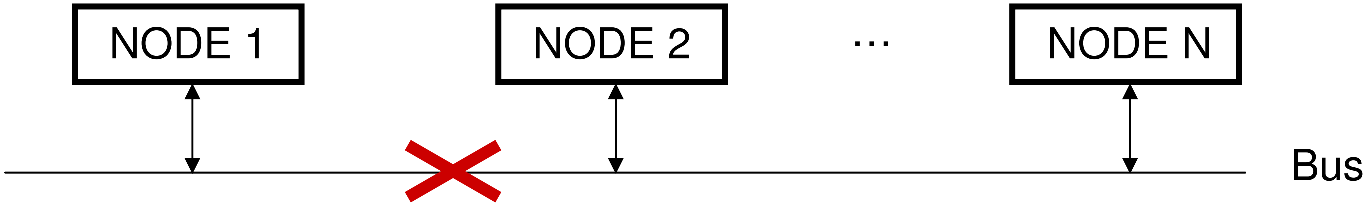
- Issues solved with **Real-Time** techniques
  - **Jeopardized system timeliness**: Provide guarantees that the network is going to exchange the messages on time

Issues solved by each discipline

- Issues solved with **Real-Time** techniques
  - **Jeopardized system timeliness**: Provide guarantees that the network is going to exchange the messages on time
- Issues solved with **Dependability** techniques
  - **Hostile environments** (EMI, vibrations, dust…): Robustness

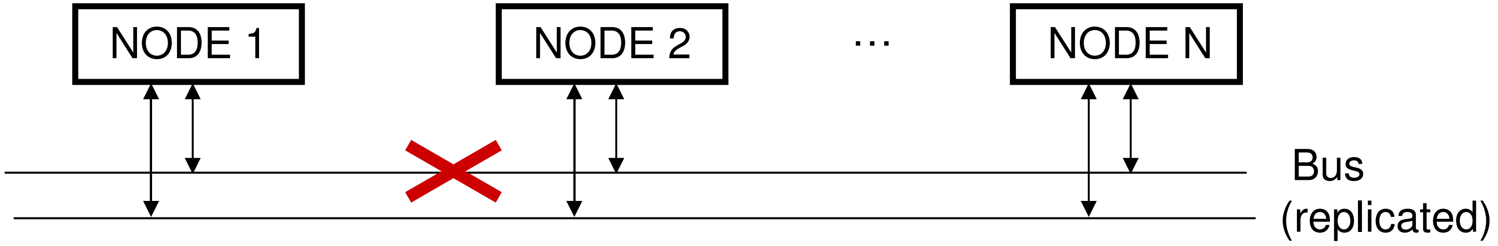
Issues solved by each discipline

- Issues solved with **Real-Time** techniques
  - **Jeopardized system timeliness**: Provide guarantees that the network is going to exchange the messages on time
- Issues solved with **Dependability** techniques
  - **Hostile environments** (EMI, vibrations, dust…): Robustness
  - **Single point of failure**:



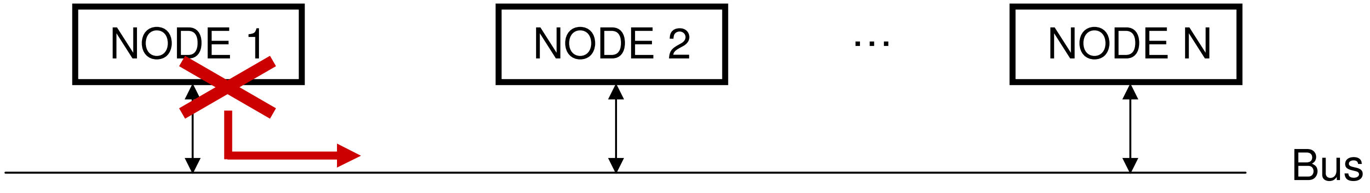
Issues solved by each discipline

- Issues solved with **Real-Time** techniques
  - **Jeopardized system timeliness**: Provide guarantees that the network is going to exchange the messages on time
- Issues solved with **Dependability** techniques
  - **Hostile environments** (EMI, vibrations, dust…): Robustness
  - **Single point of failure**: Add redundant channels or ensure high reliability



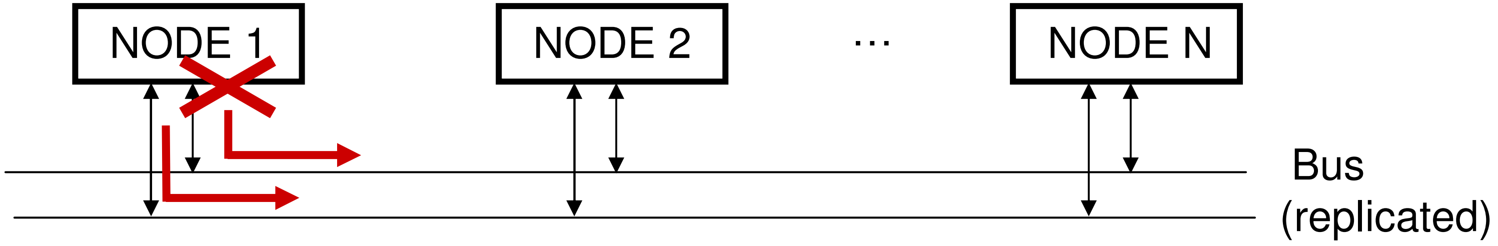
Issues solved by each discipline

- Issues solved with **Real-Time** techniques
  - **Jeopardized system timeliness**: Provide guarantees that the network is going to exchange the messages on time
- Issues solved with **Dependability** techniques
  - **Hostile environments** (EMI, vibrations, dust…): Robustness
  - **Single point of failure**: Add redundant channels or ensure high reliability
  - **Error propagation**



Issues solved by each discipline

- Issues solved with **Real-Time** techniques
  - **Jeopardized system timeliness**: Provide guarantees that the network is going to exchange the messages on time
- Issues solved with **Dependability** techniques
  - **Hostile environments** (EMI, vibrations, dust…): Robustness
  - **Single point of failure**: Add redundant channels or ensure high reliability
  - **Error propagation** (even if redundancy is present): Ensure error containment



Issues solved by each discipline

- Issues solved with **Real-Time** techniques
  - **Jeopardized system timeliness**: Provide guarantees that the network is going to exchange the messages on time
- Issues solved with **Dependability** techniques
  - **Hostile environments** (EMI, vibrations, dust…): Robustness
  - **Single point of failure**: Add redundant channels or ensure high reliability
  - **Error propagation** (even if redundancy is present): Ensure error containment
  - **Inconsistent global state**: Enforce consistency
    - **Consistent comms** are a means for, among others, **replica determinism**; and replica determinism is a particular case of **consistent view of system state**.
    - Redundant channels may also cause inconsistencies: Need for **consistent management of redundancy**

Presentation Outline

1. Distributed Embedded Systems
   1. From centralization to distribution
   2. Definition and advantages

2. The Communication Subsystem's role
   1. Two sources of problems: delays and unreliability
   2. Two disciplines: Real-Time and Dependability

3. Networks adapted to control applications
   1. The communication subsystem
   2. Requirements related to traffic
   3. Control networks and the OSI stack: structure and features

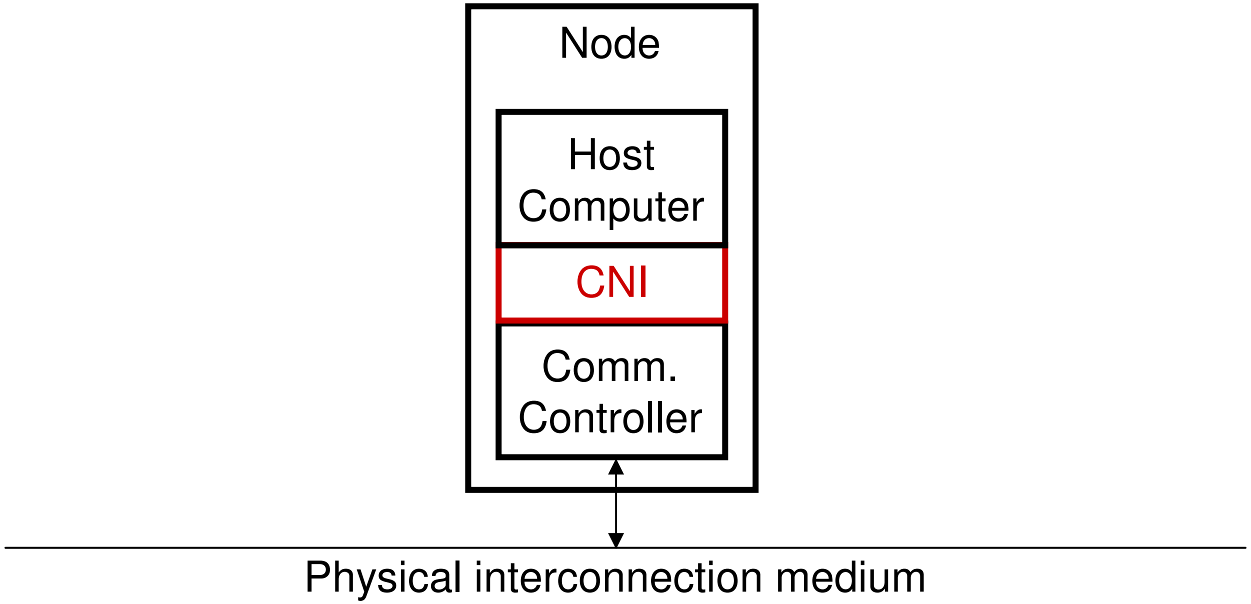
Adapting the network for control

- As indicated before, besides solving the problems we have just discussed, the communication subsystem has to be adapted to the specific characteristics of the control applications.

- Items we are going to cover:
  - Start clarifying **what the communication subsystem is and what its basic function is**
  - Continue with some **requirements** related to the traffic
  - Discuss the usual **adaptation of the OSI model**
  - **For each layer of the OSI model** describe some **typical features** of this kind of networks

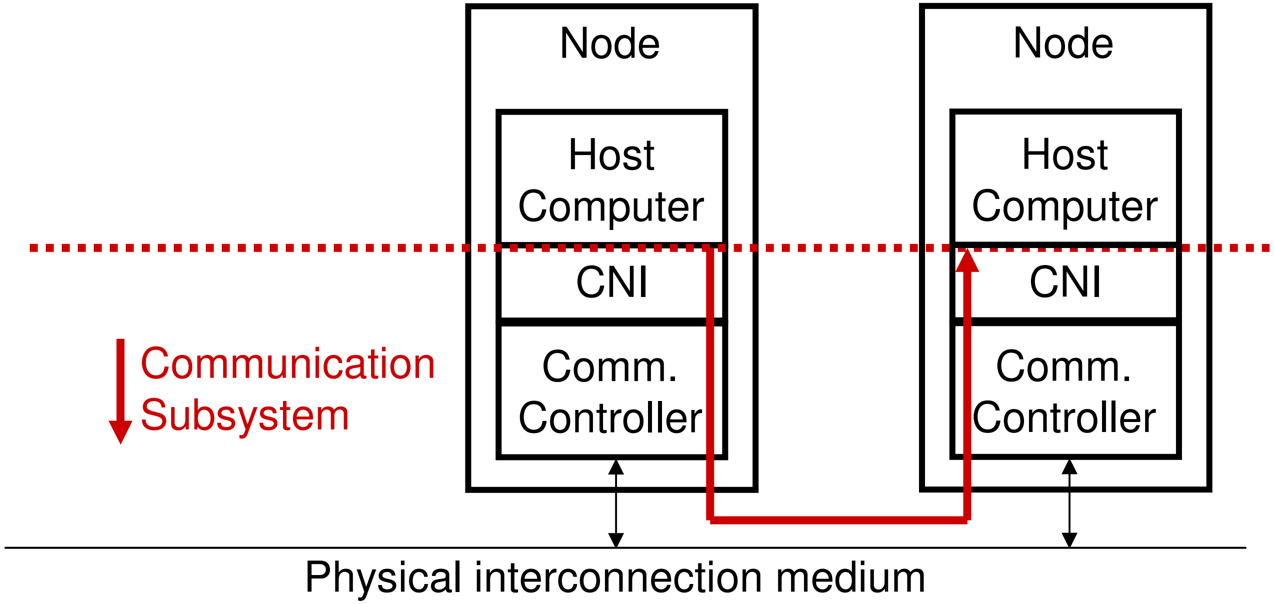
Adapting for control
The communication subsystem (1)

- A node usually includes at least two components: the local communication controller and the host computer
  - The **Communication-Network Interface** (CNI) between host and comm. controller is located at the **transport level** of the OSI stack



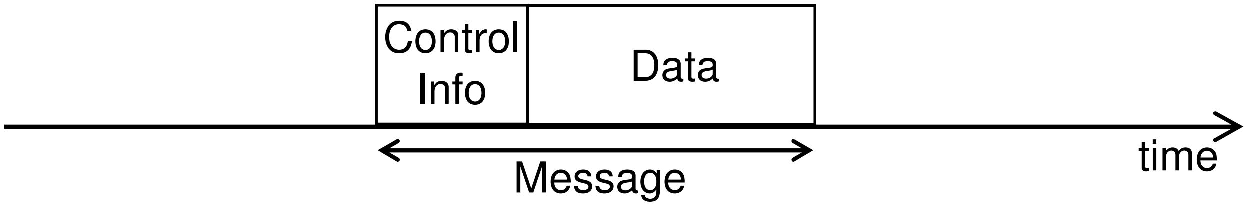
Adapting for control
The communication subsystem (2)

- The communication subsystem includes from the interconnection medium up to the CNIs of each node
  - The purpose of this system is to transport messages from the CNI of the sender to the CNI of the receiver in a **dependable** and **timely** manner



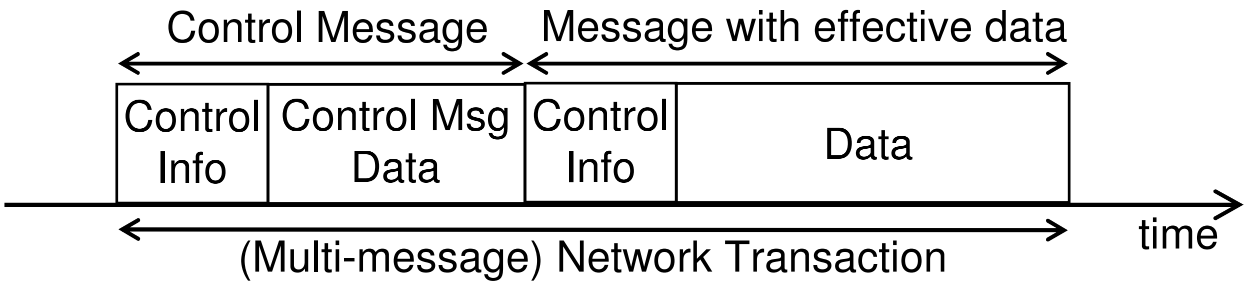
Adapting for control
The communication subsystem (3)

- **Exchange of messages** is the mechanism used by nodes to interact among them
- 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 transmission of the data (e.g. sender, destination, contents).



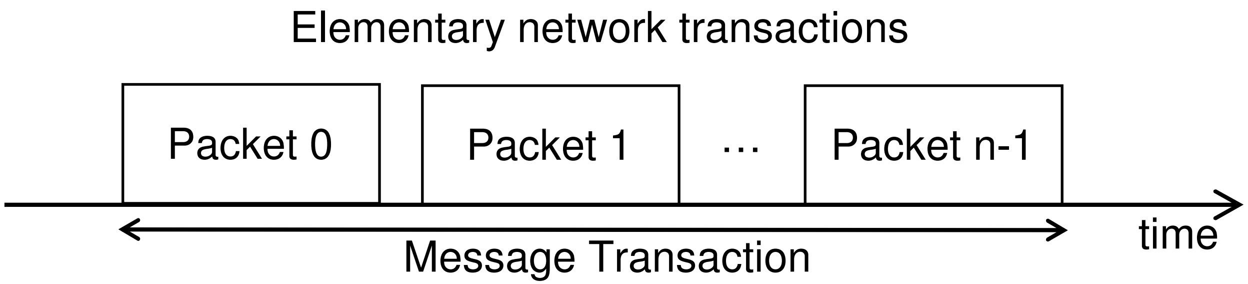
Adapting for control
The communication subsystem (4)

- 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.



Adapting for control
The communication subsystem (5)

- 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).



Adapting for control
The communication subsystem (6)

- 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.

- Data_eff = data_tx_time / transaction_duration

- In general: The **shorter** the data per transaction, the **lower** the efficiency is

Adapting for control
Some requirements related to traffic

- Efficient transmission of **short data** (few bytes)
- **Periodic** transmission (*monitoring, feedback control*) with **short periods** (ms) and **low jitter** (time variation)
- **Fast** transmission (ms) of **aperiodic** requests (*alarms*)
- Transmission of **non-real-time data** (*configuration, logs*)
- **Multicasting**

Presentation Outline

1. Distributed Embedded Systems
   1. From centralization to distribution
   2. Definition and advantages

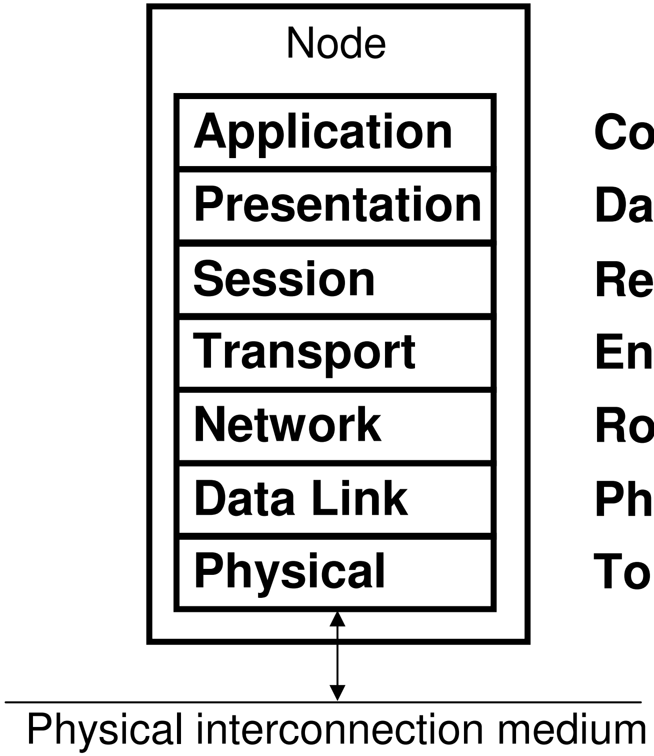
2. The Communication Subsystem’s role
   1. Two sources of problems: delays and unreliability
   2. Two disciplines: Real-Time and Dependability

3. Networks adapted to control applications
   1. The communication subsystem
   2. Requirements related to traffic
   3. Control networks and the OSI stack: structure and features

Adapting for control

Control networks and the OSI stack (1)

- The OSI stack (reference model for open systems)
  - Each layer provides specific services to the one above
  - Each layer adds some control information to the message



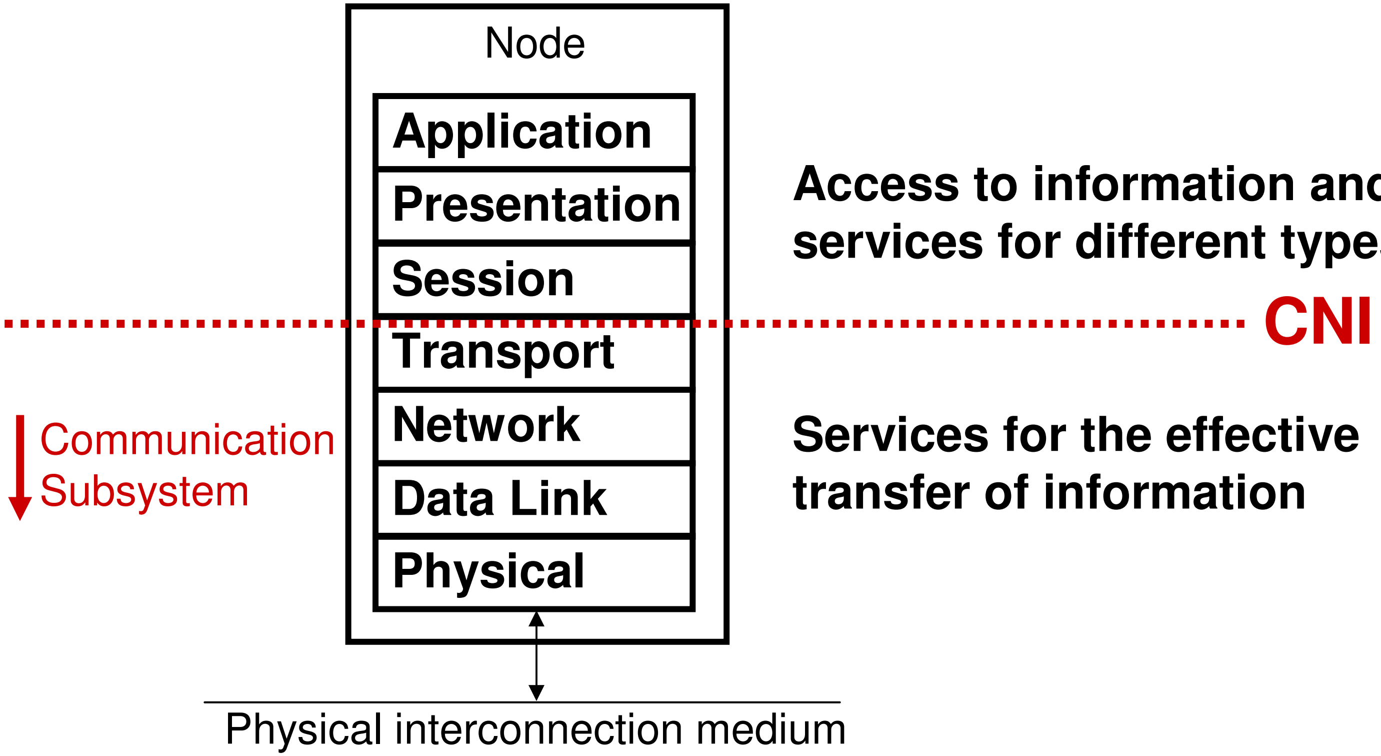
Node

Application	**Common services for classes of applications**
Presentation	**Data semantics**
Session	**Remote actions**
Transport	**End-to-end communication control**
Network	**Routing, logical addressing**
Data Link	**Physical addressing, MAC, LLC**
Physical	**Topology, medium, bit-encoding**

Physical interconnection medium

Adapting for control
Control networks and the OSI stack (2)

- The OSI stack (reference model for open systems)
  - Each layer provides specific services to the one above
  - Each layer adds some control information to the message



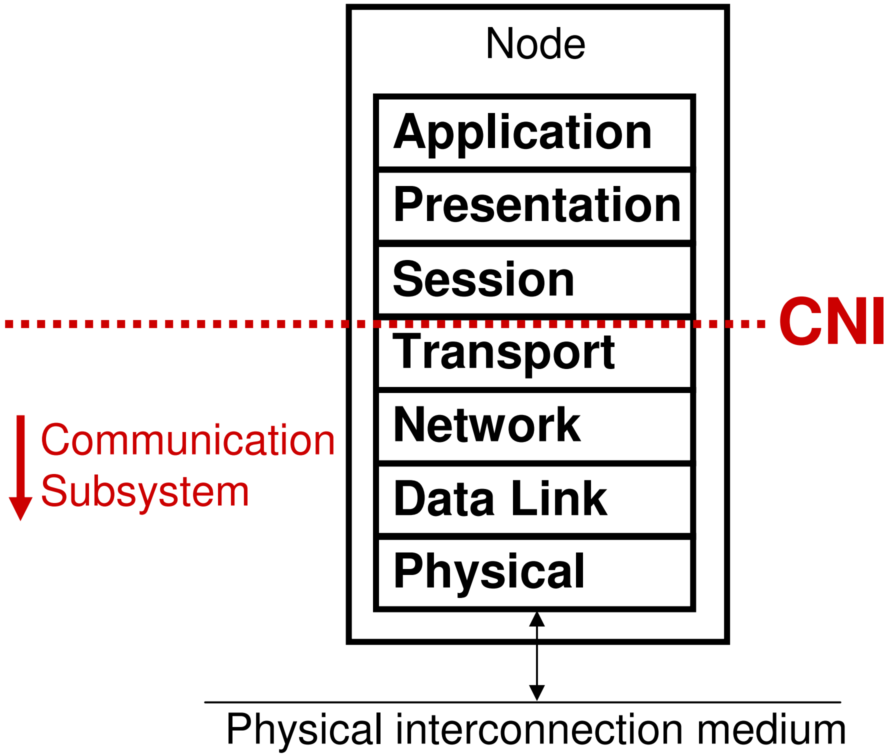
Access to information and communication services for different types of applications

Services for the effective transfer of information

Adapting for control

Control networks and the OSI stack (3)

- The OSI stack (reference model for open systems)
  - Each layer provides specific services to the one above
  - Each layer adds some control information to the message



Node

Application

Presentation

Session

CNI

Transport

Network

Data Link

Physical

Communication Subsystem

Physical interconnection medium

- This approach causes important computation and communication overheads

- The end-to-end communication delay must be bounded

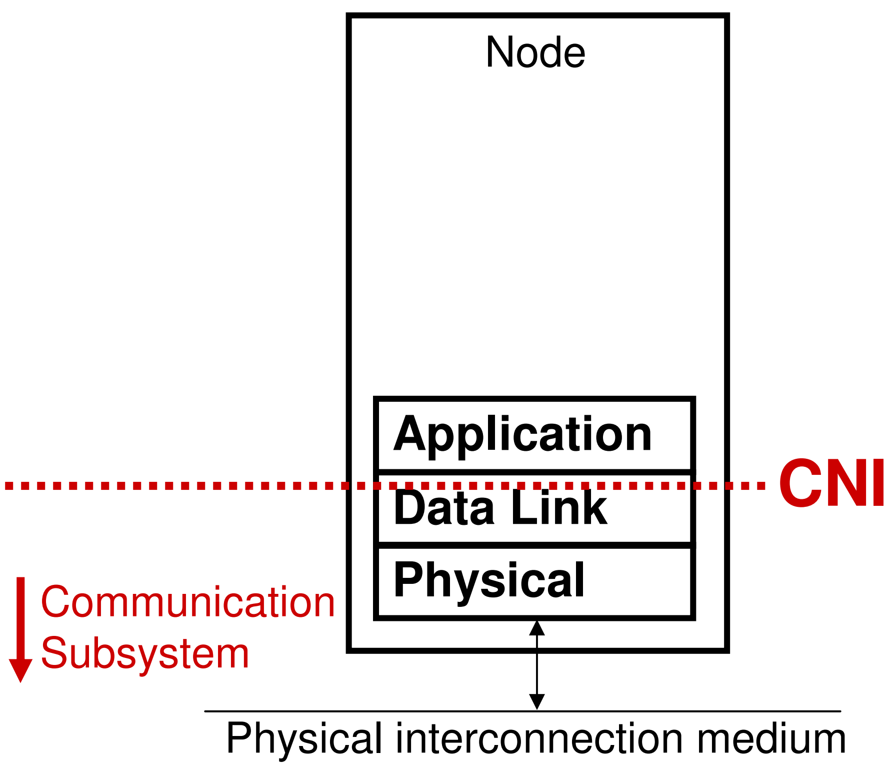
- In most real-time networks, most layers are unnecessary

No data conversion, no routing, no data fragmentation…

- Therefore…

Adapting for control
Control networks and the OSI stack (4)

- The **OSI collapsed model** is used instead
  - The seven layers are collapsed into three: Physical, Data Link and Application.



- Some functions of the transport layer (error management) are transferred to the Data Link layer.

- Any more sophisticated function is implemented at the application layer.

- Therefore the CNI is moved to the interface between Data Link (usually implemented at the controller) and Application layer

- This is the typical approach in the so-called fieldbuses (developed for Factory Automation)

Adapting for control
Physical Layer features

- In many cases low cost of the cabling scheme is fundamental!

- Therefore the bus topology has been the favorite.
  - Is simplifies cabling
  - Makes unnecessary for the protocol to perform routing
  - It facilitates a consistent reception of the same information in broadcast mode,
  - It facilitates the achievement of a consistent order in the reception of the messages.

- Some other topologies are suitable for specific goals

Adapting for control
Data Link Layer features (1)

- Among the functions of the Data Link Layer we are going to briefly discuss some aspects of:
  - **Addressing**
  - **Logical Link Control (LLC)**
  - **Medium Access Control (MAC)**
    - **Which is fundamental for RT response**

Adapting for control
Data Link Layer features (2)

- **Addressing:** identification of the parts involved in a network transaction
  - **Direct addressing:** The sender and receiver(s) are explicitly identified in every transaction, using **physical** addresses (as in Ethernet)
  - **Indirect (source) addressing:** The message **contents** are explicitly identified (e.g. temperature of sensor X). Receivers that need the message, retrieve it from the network (as in CAN and WorldFIP)
  - **Indirect (time-based) addressing:** The message is identified by the **time instant** at which it is transmitted (as in TTP)

Adapting for control
Data Link Layer features (3)

- **Logical Link Control (LLC).** More specifically **transmission error control**

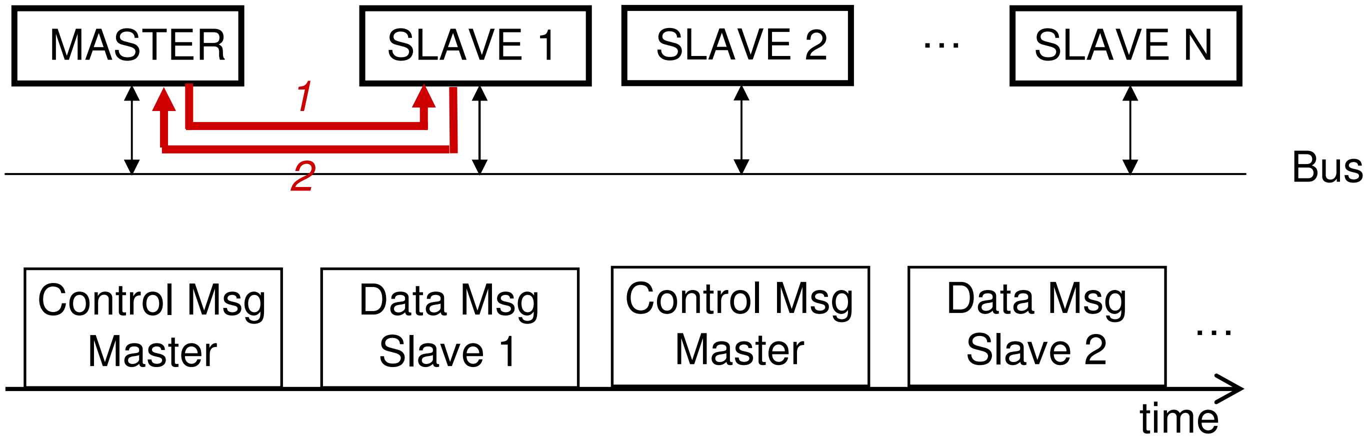
- Errors are supposed to be **detected** and then some **action is taken to solve the problem**:
  - **Forward error correction (FEC)**
    - Error correcting codes (more related to the physical layer)
    - Or the receiver waits for the next periodic transmission
  - **Automatic Repeat reQuest (ARQ)** The receiver triggers a repeat request upon error
  - **Positive Acknowledge and Retry (PAR)** The sender resends if ACK is not received
  - From a **real-time** perspective, **ARQ** and **PAR** may induce **longer delivery delays** as well as extra **communication load**

Adapting for control
Data Link Layer features (4)

- **Medium Access Control (MAC)**
  - Decides **who is to get access to the shared medium** (if there is any).
  - Therefore determines the **network access delay**
  - It is fundamental for the real-time behaviour of a network
  - Among the multiple possibilities we are going to describe:
    - Master-slave
    - TDMA
    - CSMA/CD
    - CSMA/BA
    - CSMA/CA

Adapting for control
Data Link Layer features (5)

- **Medium Access Control (MAC): Master-Slave**
  - Access granted by the Master
  - Slave msgs can be addressed to any other node
  - "Natural" in many automation applications. The central controller becomes the master of the communications
  - Nodes synchronized with the master
  - Requires one control message per data message



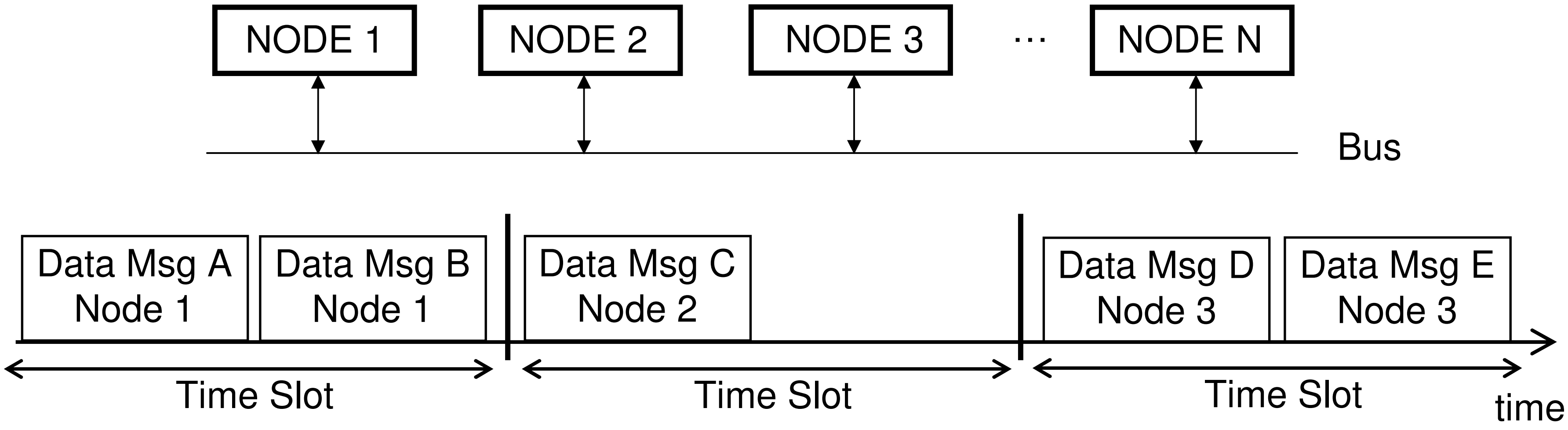
Adapting for control
Data Link Layer features (6)

- **Medium Access Control (MAC): Master-Slave**
  - The **traffic scheduling** problem becomes **local to the master.** This implies a good flexibility wrt scheduling algorithms (on-line or off-line, any type of processor scheduling)
  - The master represents a **single point of failure.** For high reliability the master must be **replicated**
    - Note that in hierarchical control systems the master is likely to be the “main” controller. Therefore it is already a single point of failure wrt control.
  - Master messages are natural synchronization points supports **precise tx triggering**
    - Note that reception instants may vary from node to node, depending on the distance to the master
  - Examples: WorldFIP or Ethernet Powerlink

Julián Proenza. UIB. Nov 2008
Luís Almeida. UA.

Adapting for control
Data Link Layer features (7)

- **Medium Access Control (MAC): TDMA (Time-Division Multiple Access)**
  - Access granted in dedicated time-slot
  - Time-slots are pre-defined in a cyclic framework
  - Tight synchronization with bus time (bus access instants are predetermined)
  - Requires global (clock) synchronization



Adapting for control

Data Link Layer features (8)

- **Medium Access Control (MAC): TDMA (Time-Division Multiple Access)**
  - Addressing can be based on time.
    - This implies **High data efficiency**
  - Quality of synchronization bounds efficiency (length of **guarding windows** between slots)
  - Typically uses **static table-based scheduling**
  - Examples: TTP/C, FlexRay-sync, TT-CAN, PROFINET

Adapting for control
Data Link Layer features (9)

- **Medium Access Control (MAC): CSMA (Carrier-Sense Multiple Access)**
  - Set of protocols based on sensing bus inactivity before transmitting (asynchronous bus access)
  - There may be collisions
  - Upon collision, nodes **back off and retry** later, **according to some specific rule** (this rule determines, to a large extent, the real-time features of the protocol)

Adapting for control
Data Link Layer features (10)

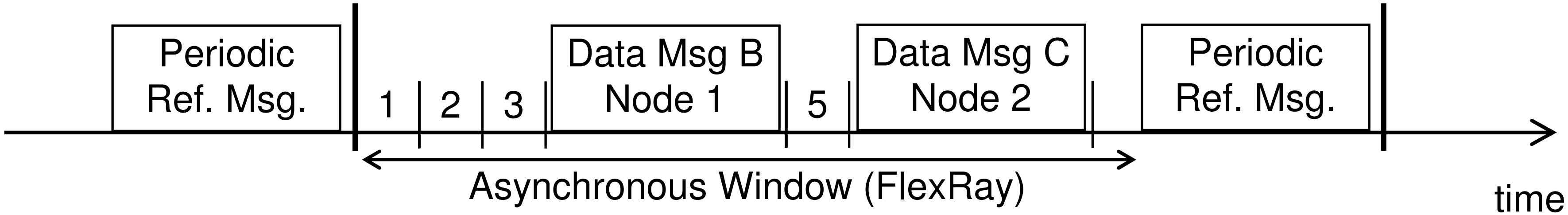
- **Medium Access Control (MAC): CSMA/CD (Carrier-Sense Multiple Access with Collision Detection)**
  - Used in shared Ethernet (hub-based)
  - Collisions are **destructive** and are detected within **collision windows** (*slots*)
  - Upon collision, the retry interval is **random** and the randomization window is doubled for each retry until 1024 slots (BEB - Binary Exponential Back-off)
  - Non-deterministic (particularly with chained collisions)

Adapting for control
Data Link Layer features (11)

- **Medium Access Control (MAC): CSMA/BA (OR CA) (Carrier-Sense Multiple Access with Bit-wise Arbitration)**
  - **Bit-wise** arbitration with **non-destructive** collisions.
  - Upon collision, highest priority node is unaffected. Nodes with lower priorities retry right after.
  - Deterministic
  - Example: CAN

Adapting for control
Data Link Layer features (12)

- **Medium Access Control (MAC): CSMA/CA (Carrier-Sense Multiple Access with Collision Avoidance)**
  - Access based on sensing bus inactivity during a number of predefined time-slots (mini-slots) after reception of a synchronous reference message
  - Corresponding mini-slot determines priority
  - Collision-free and deterministic
  - Examples: FlexRay-async (Byteflight), ARINC629-async



Adapting for control
Application Layer features (1)

- Among the functions related to the Application Layer we are going to review some aspects of:
  - **Cooperation Models**
  - **MMS**
  - **Clock Synchronization**

Adapting for control
Application Layer features (2)

- **Cooperation model: Client-Server**
  - Transactions are **triggered** by the **receiver** of the requested information (**client**).
  - Nodes that **generate** information are **servers** and only react to client requests.
  - The model is based on **unicast** transmission (one sender and one receiver)

Adapting for control
Application Layer features (3)

- **Cooperation model: Producer-Consumer**
  - Transactions are **triggered** by the nodes that **generate** information (**producers**).
  - The nodes that **need** the information, identify it when transmitted and retrieve it from the network (**consumers**)
  - The model is based on **broadcast** transmission (each message is received by all nodes)

Adapting for control
Application Layer features (4)

- **Cooperation model: Producer-Distributor-Consumer**
  - Basically similar to Producer-Consumer
  - Transactions are **triggered** by a particular node, the **distributor**, upon request from the producers or according to a pre-established schedule.
  - It is an implementation of Producer-Consumer (PC) over master-slave

Adapting for control
Application Layer features (5)

- **Cooperation model: Publisher-Subscriber**
  - Elaborate version of Producer-Consumer using the concept of **group communication**
  - Nodes must adhere to groups either as publisher (produces information) or as subscriber (consumes information)
  - Transactions are **triggered** by the **publisher** of a group and **disseminated** among the respective **subscribers**, only (**multicast**).

Adapting for control
Application Layer features (6)

- **Manufacturing Message Specification (MMS)**
  - OSI application layer messaging protocol
  - Exchange of real-time data and supervisory control information between networked devices and applications
  - Defines common functions for distributed automation systems
  - Originally published in 1990 by ISO TC 184, (application layer of GM MAP)
  - Standardized as ISO 9506-1/2
  - Nowadays implemented in a wide range of networks (Ethernet, fieldbuses, RS485, ...)

Adapting for control

Application Layer features (7)

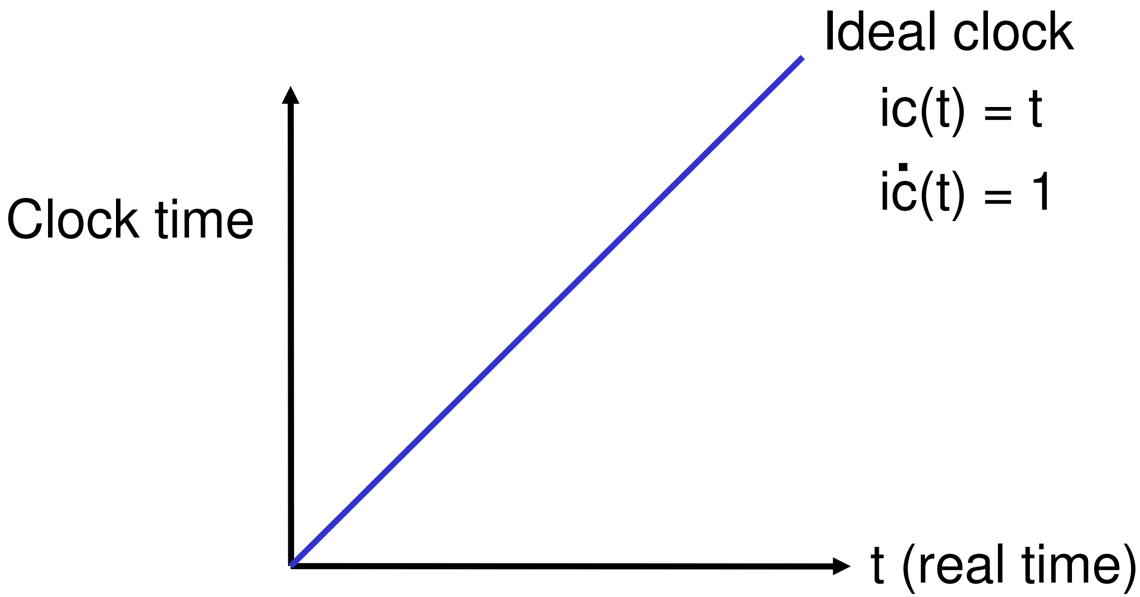
- **Clock Synchronization – Needed** in Distributed Systems **to have a common notion** of time in order to:
  - Carry out actions at desired time instants
    - e.g. synchronous data acquisition, synchronous actuation
  - Time-stamp data and events
    - e.g. establish causal relationships that led to a system failure
  - Compute the age of data
  - Synchronization of communications
    - e.g. TDMA communication schemes
  - **In fact,** several mechanisms **to improve the dependability** rely on **the presence of** such a service. E.g.:
    - Error containment in the time domain
    - Channel and node replication schemes
- **But Clock Synch is not mandatory (to be discussed)**

Adapting for control
Application Layer features (8)

- **Clock Synch – Some concepts related to time**:
  - **Newtonian Time** (the model used in computer systems): time is considered an external and continuous dimension, which is perceived equally everywhere.
    - This notion of time, which is often referred to as ***real time***, can be represented by the set of positive real numbers.
    - Any time instant belonging to this external dimension is usually denoted with the letter ***t***.

Adapting for control
Application Layer features (9)

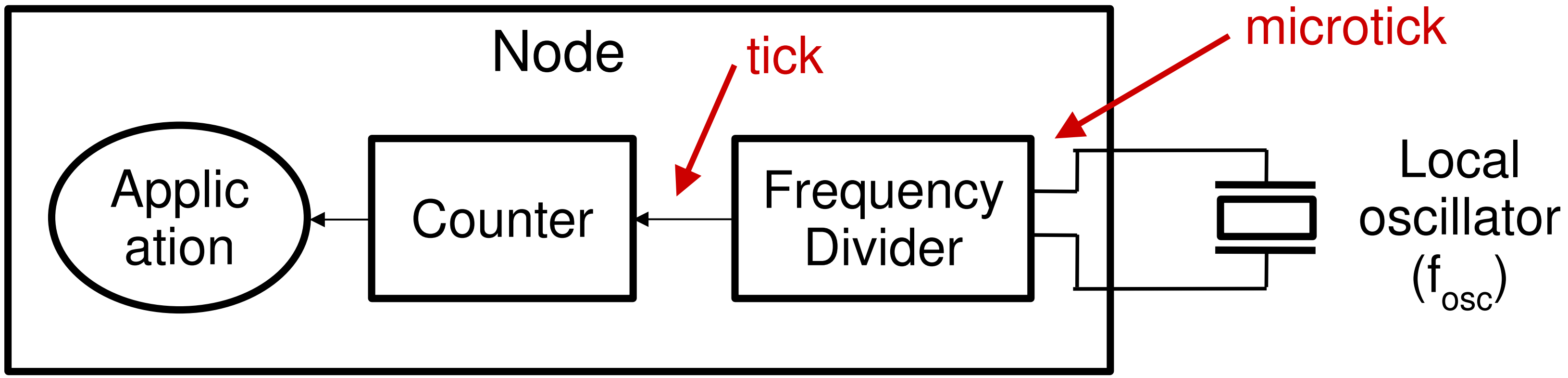
- **Clock Synch – Some concepts related to time**:
  - **Ideal Clock**: the clock that always reflects the value of real time. This clock does not have physical existence, and can be considered only theoretically. **AKA**: Newtonian clock, Perfect clock



Adapting for control

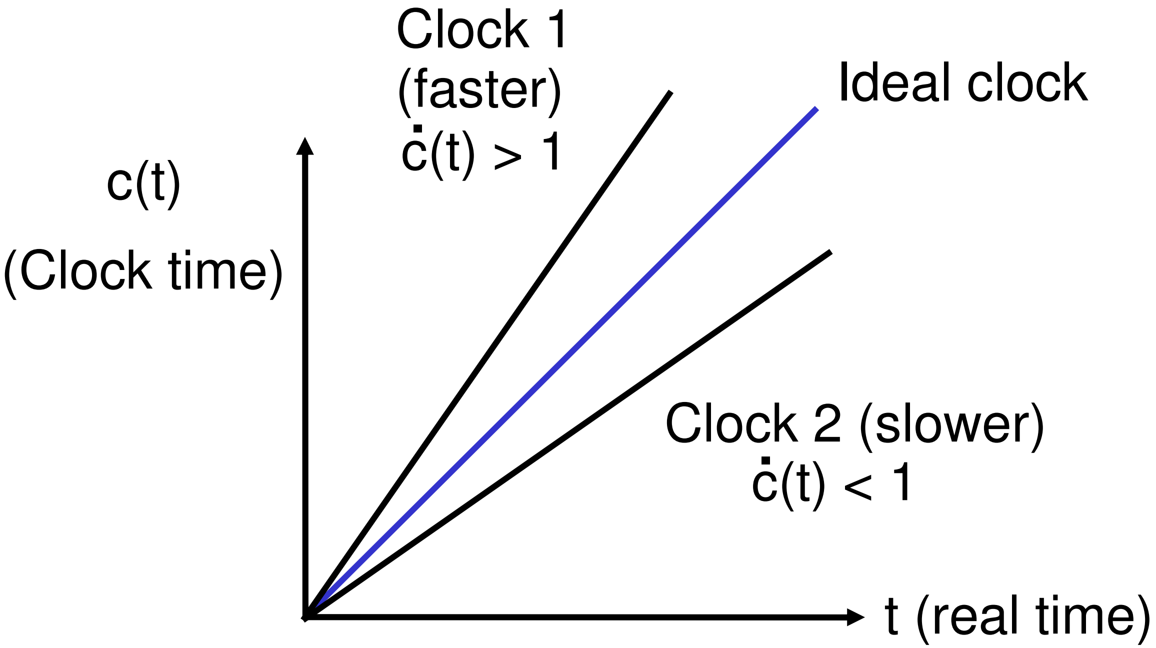
Application Layer features (10)

- **Clock Synch – Some concepts related to time**:
  - **Physical Clock**: the value of a counter that is incremented according to a local oscillator in each node. This clock has physical existence. **AKA**: Hardware clock



Adapting for control
Application Layer features (11)

- **Clock Synch – Some concepts related to time**:
  - **Physical Clock**: local oscillators inevitably deviate from their nominal frequency: c'(t) = 1+ r (t), where r(t) is called the ***drift*** of the phys. clock. There is always a Max(|r(t)|), called ***max. drift***.



$| c_1(t) – c_2(t) |$
is **unbounded**

Adapting for control
Application Layer features (12)

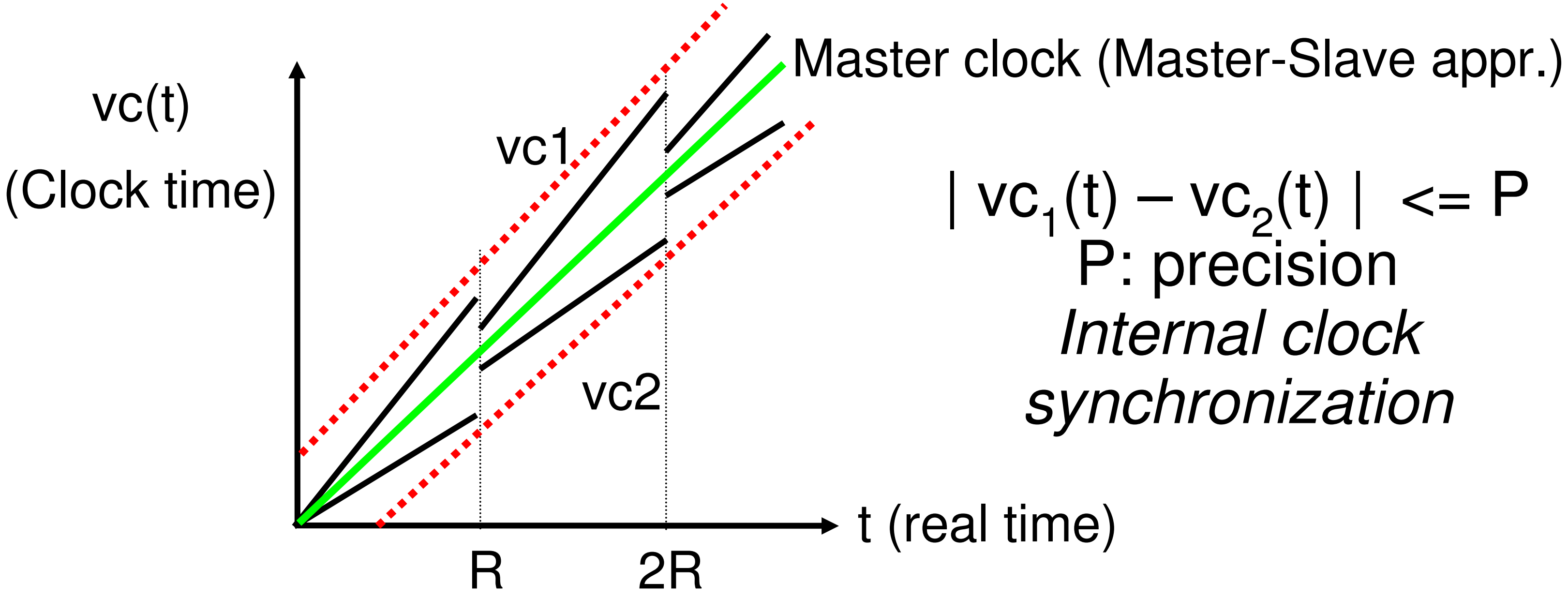
- **Clock Synch – Some concepts related to time**:
  - **Global Clock**: Some distributed embedded systems are built upon the assumption that all the nodes have access to a time reference which is perceived equally everywhere within the system. The clock that provides the value of this **absolute time reference** is called the ***global clock***.
    - Does not ensure that all nodes actually perceive it equally!
    - It does not map real time perfectly, but may deviate from real time.
    - **AKA**: Absolute clock, Reference clock, System-wide clock
  - The global clock **may have physical existence or not**
    - Physical existence, for instance, in a master-slave clock synch (master's physical clock is the global clock)
    - No physical existence when the global clock is defined as a function (*convergence function*) of the values of a number of physical clocks (e.g., average of the value of several physical clocks)

Adapting for control
Application Layer features (13)

- **Clock Synch – Some concepts related to time**:
  - **Virtual Clock**: Is the result of **synchronizing** a physical clock with the global clock.
    - The physical clocks by themselves cannot be used to approximate the value of the global clock, because their different drifts make them diverge as time passes by. To solve this problem, every node must periodically execute a ***clock synchronization algorithm*** to estimate its deviation with respect to the global clock and correct it*.*
    - The clock obtained after this correction is applied on the physical clock is called the ***virtual clock***, and is denoted as vc(t).
    - This clock has physical existence: it is the physical clock when it is periodically synchronized.
    - **AKA**: Logical clock, Synchronized clock

Adapting for control
Application Layer features (14)

- **Clock Synch – Some concepts related to time**:
  - **Virtual Clock.** The three steps of a clock synch algorithm.
    - Detection of the periodical synchronization instant.
    - Estimation of the value of the global clock at that time instant, and therefore of the error between the virtual clock and the global clock.
    - Correction of this error.



$$| vc_1(t) - vc_2(t) | <= P$$

P: precision

Internal clock synchronization

Adapting for control
Application Layer features (15)

- **Clock Synch – But this service is not always needed. Two kinds of systems (paradigms)**:
  - **Timer-driven system**: assumes that the clock of every node is independent of the clocks of the rest of the nodes (only physical clocks are used). Each node can only measure durations that are relative to its own physical clock
    - timestamps do not give any information about the order of events if such events have happened in different nodes (taken with different physical clocks).
    - Moreover it is not possible to measure the duration of actions that start at one node and finish at another one, i.e., actions that are distributed.
    - To achieve some coordination among nodes, local timers are used that are set by the nodes upon the occurrence of a given event (e.g. the transmission or reception of certain messages)
  - **Clock-driven system**: uses virtual clocks (synchronized).

Adapting for control
Application Layer features (16)

- **Clock Synch – Synchronization Requirements**:
  - Generic data networks:
    - **Applications**: distributed file systems, financial transactions, office applications
    - **Precision**: from milliseconds to seconds
    - **Protocols**: Network Time Protocol (NTP) (covers the LAN and WAN area)
  - Distributed real-time systems:
    - **Applications**: supervision, measurement and control systems
    - **Precision**: from sub-microseconds to milliseconds
    - **Low cost! (low cost devices, little network resources)**
    - **Protocols**: IEEE1588, SynUTC, **protocol-tailored solutions**

Adapting for control
Application Layer features (17)

- **Clock Synch – IEEE1588 overview**
  - Hierarchic, master/slave
    - *grandmaster clock:* best clock in the system
    - *subnet master:* best clock in a subnet

    (single subnet: grandmaster and master are the same)
  - In each subnet nodes synchronize with the subnet master
  - Subnet master synchronize with the grandmaster
  - Master election is automatic (Best Master Clock algorithm). Mechanisms for indication a preferred set of masters
  - External synchronization possible (e.g. GPS)

Presentation Outline

1. Distributed Embedded Systems
   1. From centralization to distribution
   2. Definition and advantages

2. The Communication Subsystem’s role
   1. Two sources of problems: delays and unreliability
   2. Two disciplines: Real-Time and Dependability

3. Networks adapted to control applications
   1. The communication subsystem
   2. Requirements related to traffic
   3. Control networks and the OSI stack: structure and features

In the rest of the course…

- An introduction to Dependable systems and some ideas on how to solve the corresponding issues in distributed systems.

- Intensification on methods for validating the design of systems (both wrt R-T and Dependability related features).

- A description of the application of R-T techniques to the scheduling and schedulability analysis of R-T networks

References

Sources for these slides and useful for further reading

- L. Almeida et al. *Slides of the course "Redes de Comunicação em Ambientes Industriais".* Universidade de Aveiro. 2006.
- H. Kopetz. *Real-Time Systems: Design Principles for Distributed Embedded Applications*. Kluwer Academic Publishers. 1997.
- P. Veríssimo and L. Rodríguez. *Distributed Systems for System Architects*. Kluwer Academic Publishers. 2001.
- J. Pimentel et al. *Dependable Automotive CAN Networks*. Chapter in the Automotive Embedded Systems Handbook. CRC Press. To appear in 2008
- G.Rodríguez-Navas et al. *Using Timed Automata for Modeling the Clocks of a Distributed Embedded System*. Chapter in Behavioral Modeling for Embedded Systems and Technologies: Applications for Design and Implementation. IGI Global. To appear in 2009