Distribution Middleware Technologies for Cyber Physical Systems

Encuentro UPV/EHU – CIC Energigune

12 May 2014 – Vitoria-Gasteiz

Isidro Calvo

Ismael Etxeberria

Adrián Noguero

Isidro.calvo@ehu.es

Dept. de Ingeniería de Sistemas y Automática

E.U. de Ingeniería de Vitoria-Gasteiz

UPV/EHU

Escuela Universitaria de Ingeniería Vitoria-Gasteiz
Ingeniaritzako Unibertsitate Eskola Vitoria-Gasteiz

eman ta zabal zazu
Universidad del País Vasco
Euskal Herriko Unibertsitatea

Index

Distribution Middleware Technologies for Cyber Physical Systems

1. Introduction

2. Requirements of CPS
3. Distribution middleware for CPS
4. Middleware architectures for CPS
5. Infrastructure modelling of CPS
6. Conclusions

Cyber Physical Systems (CPS)

CPS integrate computation and physical processes

The term CPS was coined around 2006 by researchers of different disciplines:

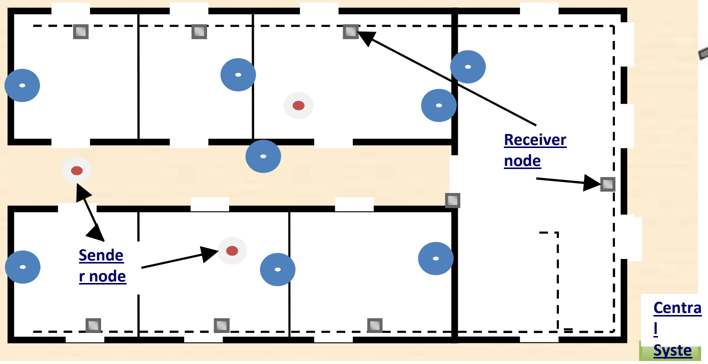
- Real-time systems
- Hybrid systems
- Control systems

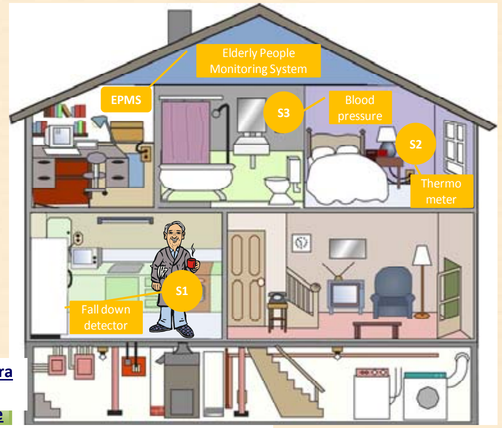
Internet of things vs. CPS => Two different communities

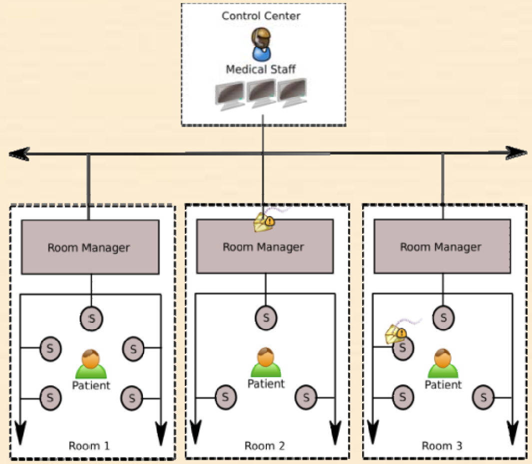
CPS are increasingly used in several domains:

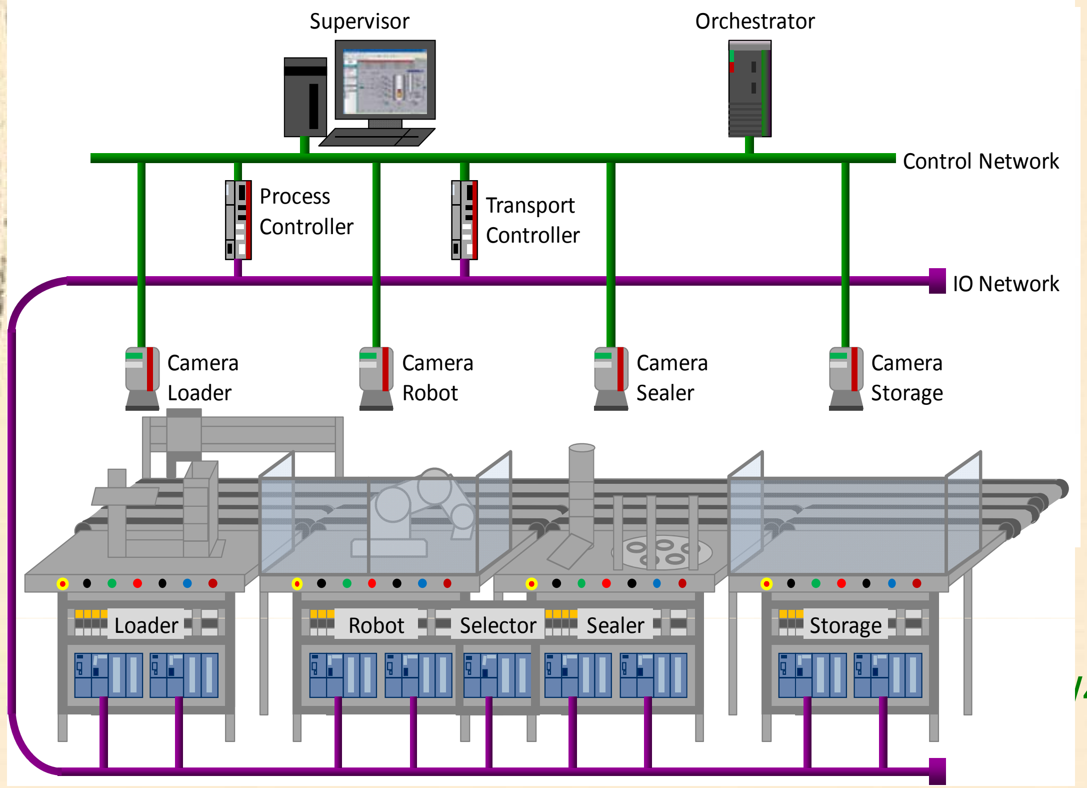
- Healthcare
- Transportation
- Process control
- Manufacturing
- Electric power grids

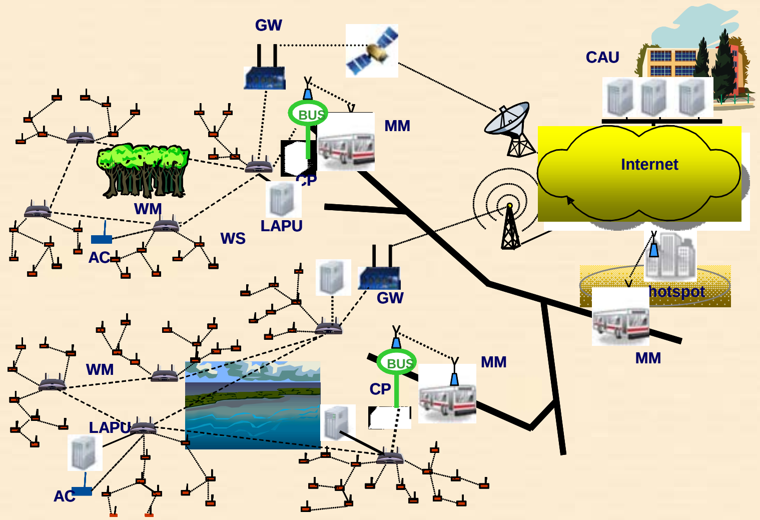
Application domains





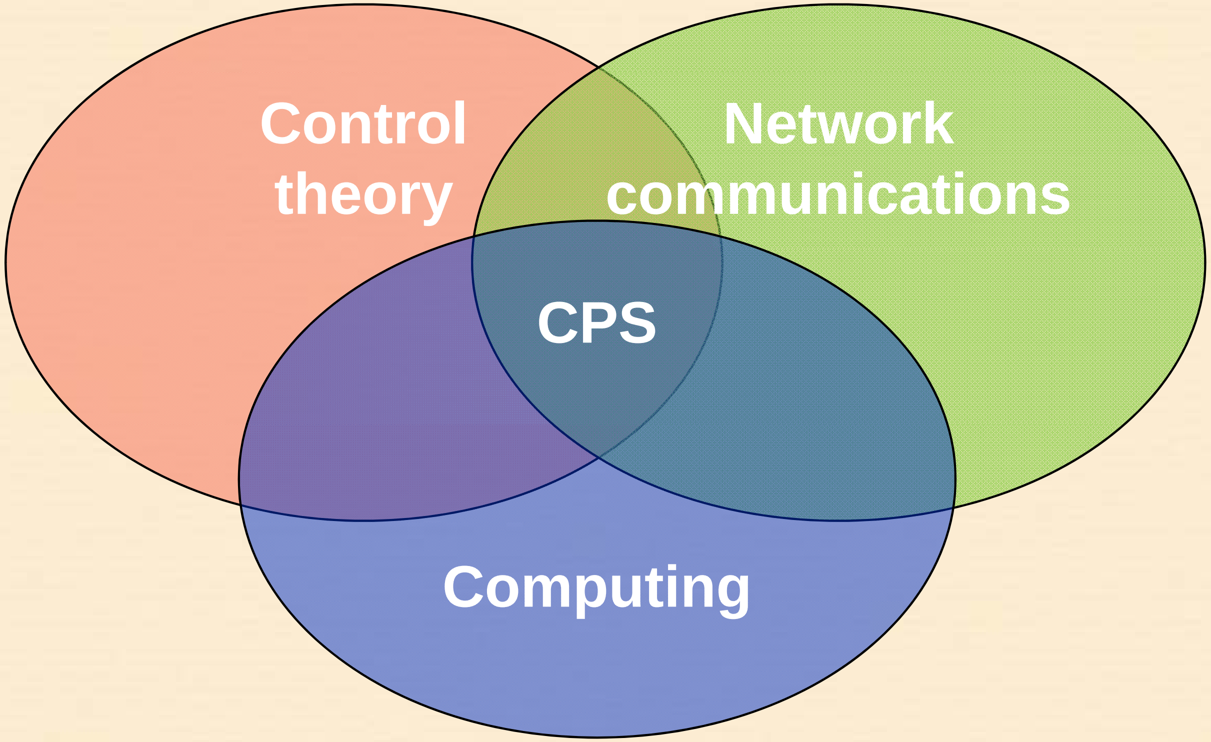






eman ta zabal zazu

Cyber Physical Systems (CPS)



Technology implications

CPS benefit from **advances in several research domains**:

- Networked control systems
- Hybrid systems
- Real-time computing and networking
- Wireless technologies
- Smart sensors and actuators
- Security
- Model driven development
- Evolution in computing platforms:
  Low cost / Small size / High performance
- High bandwidth networks
- Improvements in energy capacity and management

Difficulties of building CPS

Building CPS is not a trivial task:

- **Integration** of different technologies
- **Different points of view** must be considered:
  Computing (concurrency and real-time issues), Control theory, Network communications
- **Heterogeneous** embedded hardware and software platforms
- **Fault-tolerance**: Dependable and safe systems
- Flexibility and adaptability to changes
- Scalability to build large systems
- Maintenance: CPS are created to work for long periods of time (e.g. several years)

- **Of course: Building new systems on schedule and keeping low costs!**

Communication issues

- Radical approaches:

  Design of new specific technologies and protocols from scratch for CPS

- **Pragmatic approach:**

  Use in short to medium term of worldwide accepted standards even though they provide lower performance:

  - Internet Protocols (IP)
  - IEEE802.11 (Wifi)

  Use of patches to improve their performance

Use of middleware technologies

- Ease the construction of new applications
- Hide low level implementation details
- Generic middleware technologies (J2EE, .NET, CORBA, DDS, or Web Services) drawbacks:
  - Tend to be excessive and introduce performance overhead
  - Do not match some specific requirements of CPS

- **Need of specific middleware architectures for CPS which introduce specific services and abstractions**
  - **Avoiding reinventing the wheel**
  - **Keep an adequate performance**

Main requirements of CPS

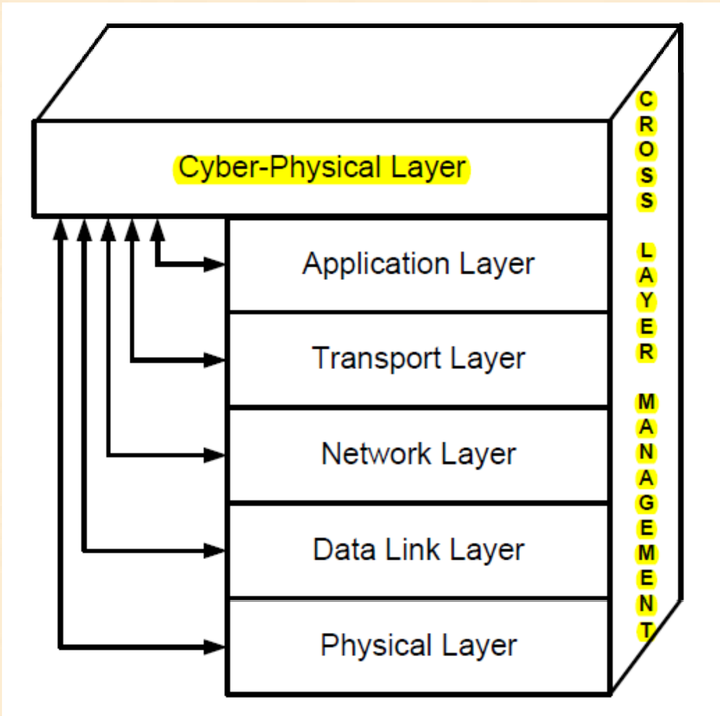
The nature of CPS imposes specific requirements different from general purpose computing:

- **Dealing with time:** Physical world is concurrent and reacting in time is critical
- **Close integration:** Highly coupled systems
- **Solving heterogeneity:** Mixture of platforms and technologies (computing platforms, OS, programming languages, network technologies)
- **Low resource devices:** CPU, memory, network bandwidth and energy consumption
- **Dynamic reconfiguration and reorganization:** Capable of adapting to changes in the physical world or changing requirements
- **Dependability and robustness:** Safety must be ensured even in adverse situations. Sometimes, CPS require certification.

New theories and tools are required

Lack of theory and **tools that help designers to build CPS** in an efficient way

- Extending **abstractions** that integrate in the existing nextwork infrastructures and reference models
- Koubaa, A and Anderson, B. (2009) proposed a **protocol stack** for CPS on top of the TCP/IP stack



Escuela Universitaria de Ingeniería Vitoria-Gasteiz | Ingeniaritzako Unibertsitate Eskola Vitoria-Gasteiz

eman ta zabal zazu

Need of distribution middleware in CPS

- Preeminence of TCP/IP stack
- Best effort techniques => Difficult to achieve timing predictability
- CPS require combining different types of traffic with different Quality of Service (QoS) requirements
- Convergence of Internet technologies with embedded systems
- Programming directly over TCP/IP sockets is complex, especially as the number of devices increases
- According to T. Pearson (2005), **the use of middleware produce up to 50% decrease in software development and costs**
- Middleware technologies:

  Provide an additional layer on top of the TCP transport layer

  Follow modular approaches

  Solve scalability and heterogeneity needs

General purpose middleware specs

- Success of some middleware specifications

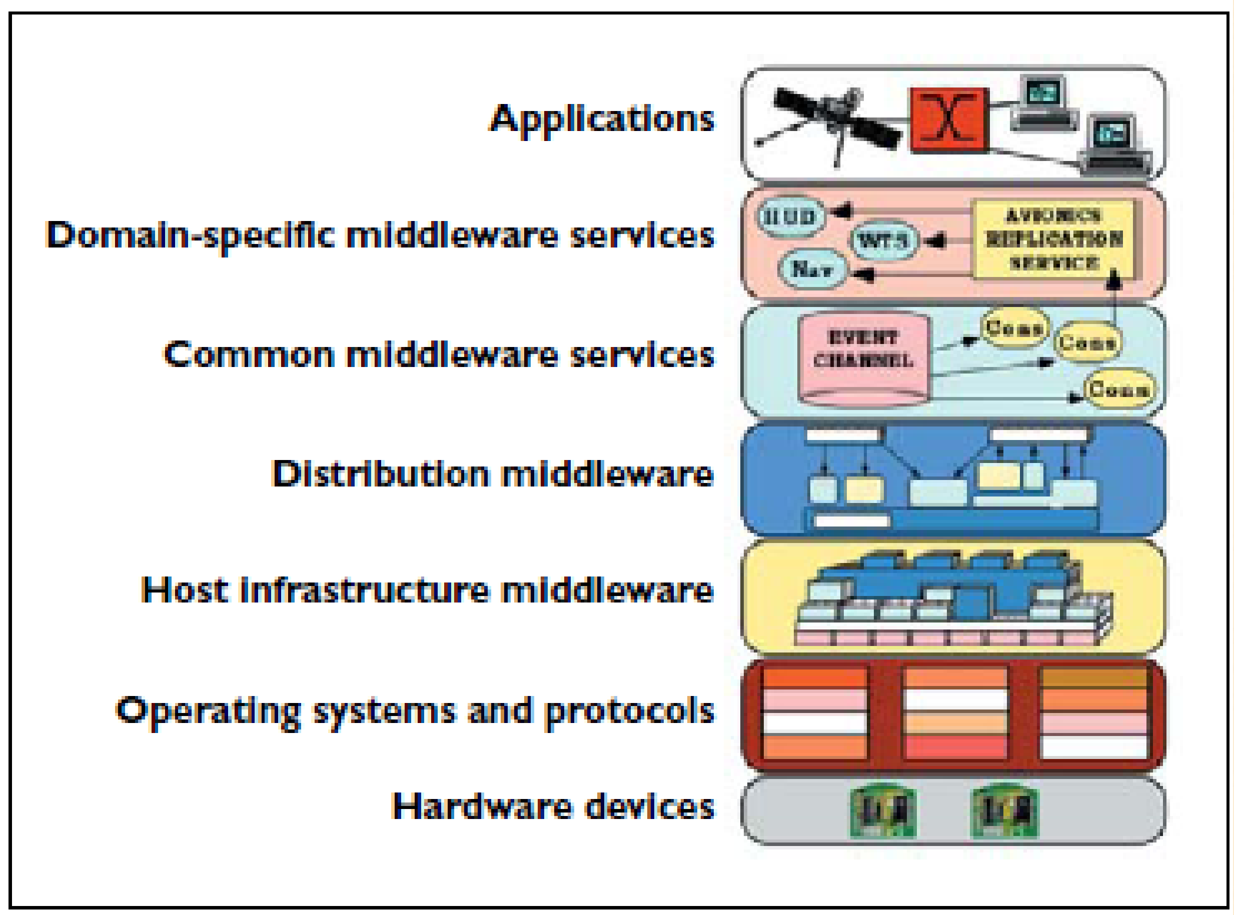
  CORBA, ICE, DDS, Web Services, OPC (in industrial environments)
- Used for distributing information (Distribution middleware specifications)
- Do not solve specific challenges involved at the construction of CPS:

  Real-time, management of the resources, management and schedulability of distributed systems, etc.
- Lack of specific features for CPS domain applications

1. Introduction
2. Requirements of CPS
3. Distribution middleware for CPS
4. Middleware architectures for CPS
5. Infrastructure modelling of CPS
6. Conclusions

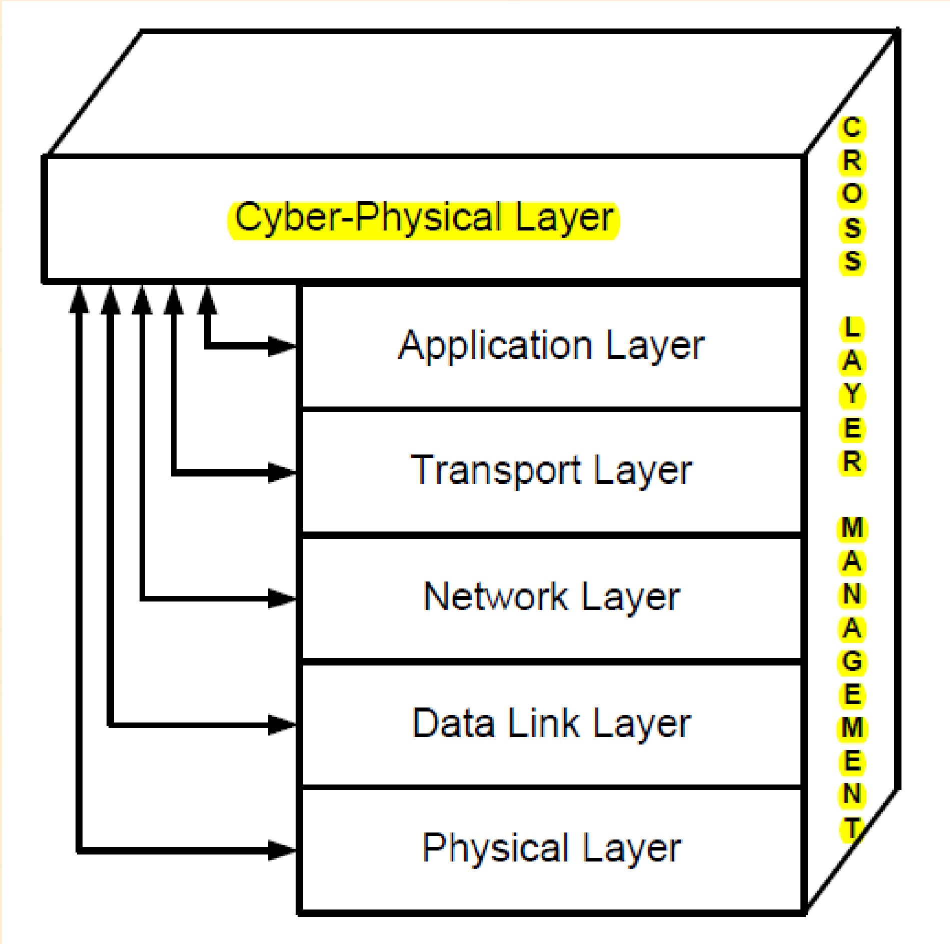
Layered middleware architectures

- Middleware is organized in a hierarchy of several layers (D.C Schmidt, 2002)



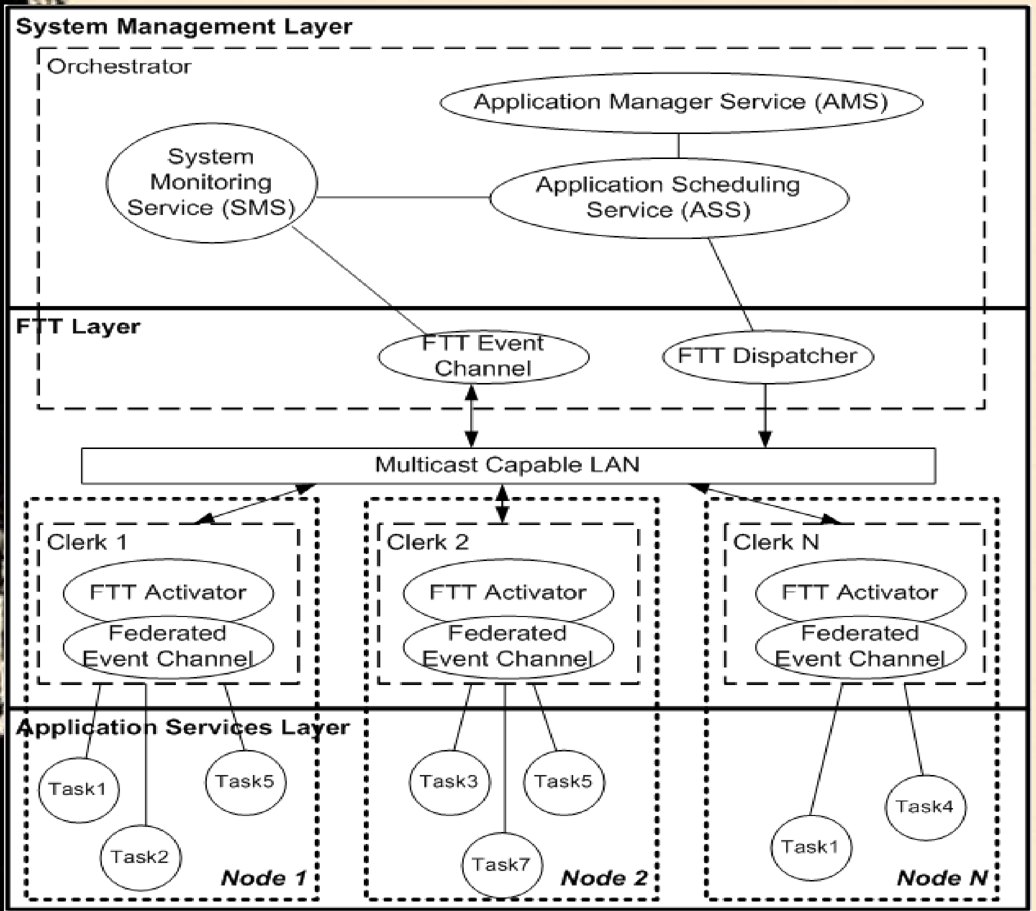
Middleware architectures for CPS

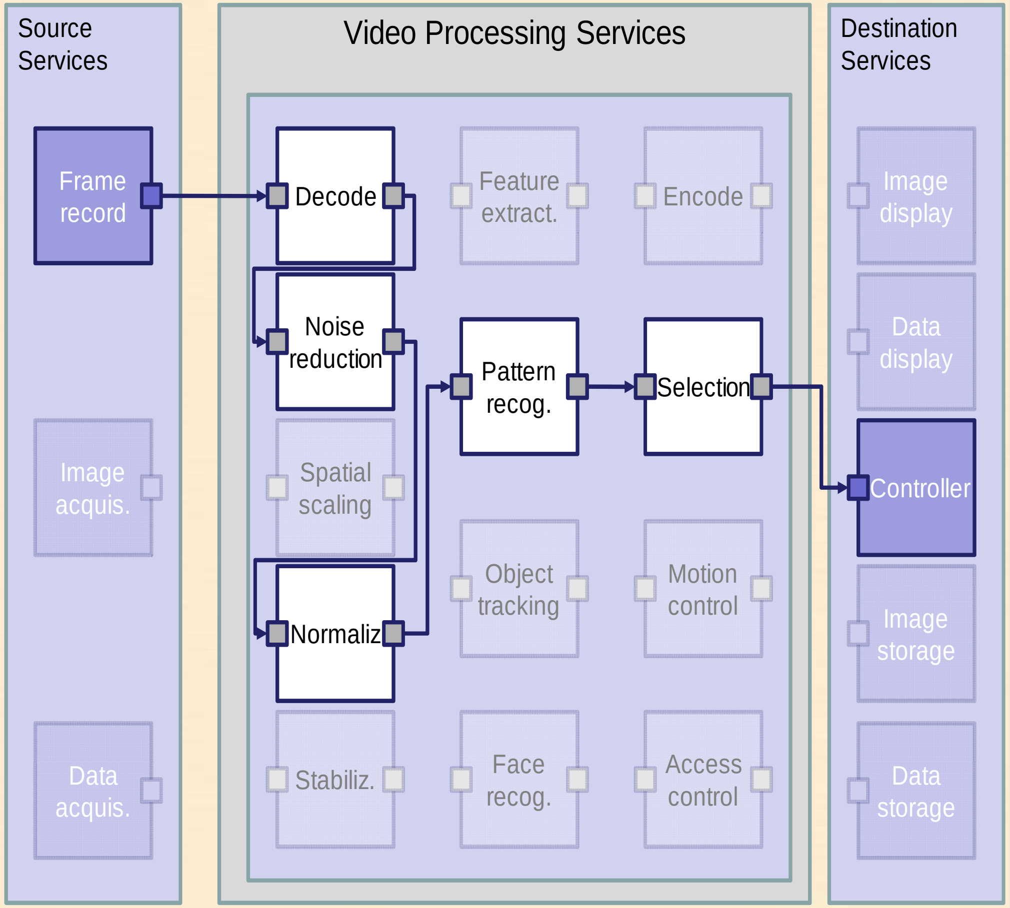
- Provide specific services for CPS and software abstractions
- The Cyber-Physical Layer



FTT-MA

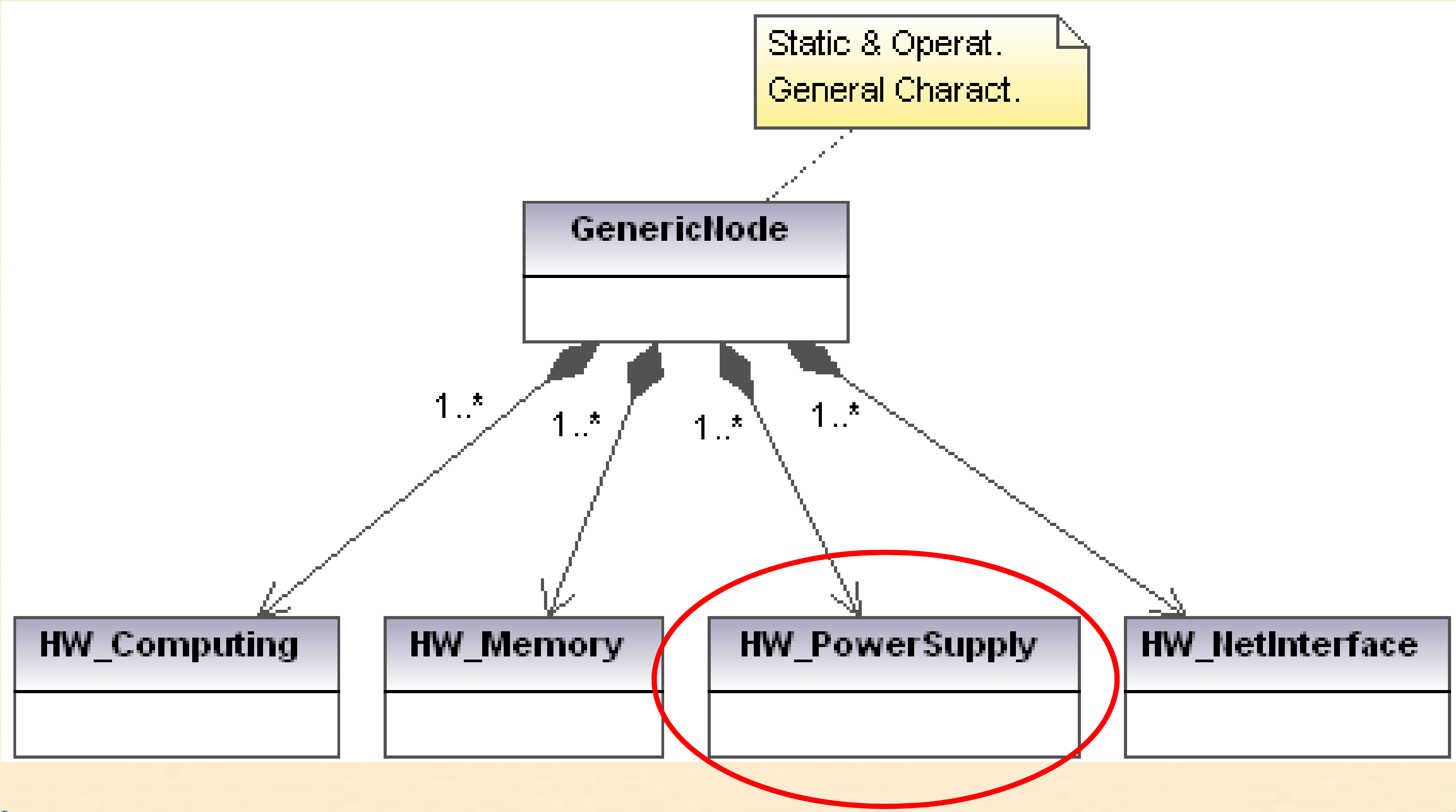
- Synchronizing the tasks activations of a distributed system according to a plan that may be changed at run-time (*PhD Adrian Noguero*)





Infrastructure modelling

- CPS require a close interaction with the underlying infrastructure.
- It is necessary to model both static and dynamic attributes



Conclusions

The design of CPS:

- ❑ Involve the use of different computing, communication and control technologies
- ❑ Require satisfying simultaneously several restrictive constrains
- ❑ Frequently, IP technologies are accepted
- ❑ New abstractions that represent CPS entities are needed
- ❑ Specific middleware architectures may provide services and abstractions for CPS
- ❑ Infrastructure modelling must be considered

Algunos proyectos recientes:

- Proyectos Europeos:
  - iLAND, m**I**dd**L**ew**A**re for deterministic dynamically reconfigurable **N**etworke**D** embedded systems (2009-2012 ARTEMIS)
- Proyectos Nacionales:
  - QoS DREAMS, QoS **D**riven **RE**configur**A**tion of Distributed Syste**MS** (2013-15 MINECO)
- Proyectos autonómicos / EHU
  - Diseño de sistemas distribuidos ciber-físicos con el estándar IEC61499 (EHU13/42, 2013-2015)
  - FACTWARE, Arquitectura Middleware para la composición y reconfiguración dinámica y determinista en aplicaciones de automatización industrial (SAI11/31, 2011-12)
  - Diseño de un entorno para la construcción de laboratorios remotos (EHU09/29, 2007-2009)

Docencia en Masters

- Máster en Ingeniería de Control, Automatización y Robótica
  - Dept. Ing. Sistemas y Automática

- Máster en Ingeniería de Control, Automatización y Robótica
  - E.U. de Ingeniería de Vitoria-Gasteiz

Thank you

eman ta zabal zazu

Universidad del País Vasco
Euskal Herriko Unibertsitatea

UPV/EHU

Escuela Universitaria de Ingeniería Vitoria-Gasteiz
Ingeniaritzako Unibertsitate Eskola Vitoria-Gasteiz