Cyber-Physical Systems - Challenges and Lessons to Learn

Karl-Erik Årzén
Lund University, Sweden

LUND INSTITUTE OF TECHNOLOGY Lund University

artist

ELLIIT

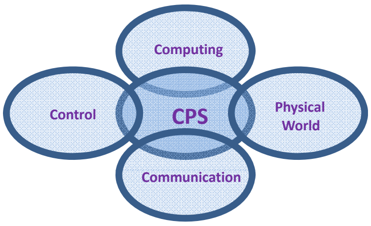
ACTORS

CPSWEEK Workshop on Architectures for Cyber-Physical Systems, Chicago, April 11, 2011

Outline

- **The Essence of CPS**
- Challenges
- Lessons to Learn

What is your perspective on CPS?



1. Resource constraints
2. Pervasive / Ubiquitous Computing
   – "Internet of Things" / Machine-to-Machine Comm.
3. Mobility and Wireless Connectivity
   – Distributed Systems

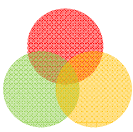
Paradigm Shift

- When designing complex artefacts **separation of concerns** is a good design principle.
  – Buildings
  – Vehicles
  – Distributed Systems

- Obtained through engineering principles, good architectures, design rules etc
- Often the main objective for these design principles is to save **human resources** (engineering time)
- However, things are changing …

Paradigm Shift

- In many areas the main objective today is to save **natural resources** rather than human
  – Often energy / emissions related



- Crosscutting concerns that often cause the traditional separation-based approaches to break down
  – Interaction and interference
- Requires integration of multiple sub-systems both during design and operation
- Several names:
  – Integration-Based Design
  – Codesign
  – Cross-layer design

Some Examples

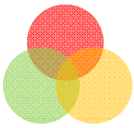
- Smart/green/low-energy buildings
  – Requires interaction been architects, mechanical engineers and control engineers
  – Requires interaction between a number of sub-systems (HVAC, lightning, security, …)

- Green cars
- Smart grid
- Server farms
  – Interaction between load balancing and energy consumption
- Battery-driven computing and communication devices
  – Smart phones, Laptops, Sensor networks, ….

- Cross-layer design and optimization in networks
- Embedded Systems
  • Resource-aware design nothing new!

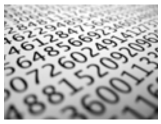
Paradigm Shift

- Eventually we will find new ways of organizing our work
- Revolutionary paradigm shifts have occ



Pervasive Computing

- Computing power embedded in our daily life
  - Sensors
  - Actuators
  - Devices
- Connected via public IP addresses
  - IPv6 - 3.4×10^{38} unique addresses
    - Number of cells in the human body (> 2×10^{14})
    - Total number of printed material in the world (1.6×10^{18} bits)
  - Every light-switch in every home in the world can easily have its own IP address
- Machine-to-machine communication

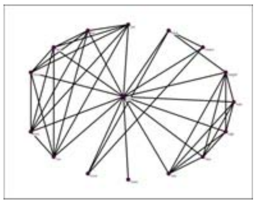


Mobility and Wireless Connectivity

- Mobile CPS systems
  - Vehicular systems
  - UAVs
  - Humans
    - Smart phones
      - As gateways to local sensors, body-area-networks, …
      - As gateways to compute servers or cloud infrastructures
- Wireless connectivity
  - Cellular radio technology
  - Wireless networks (WLAN, “Zigbee”, Bluetooth, Peanut, UWB, ….)
  - Combinations between cellular radio technology and wireless networks

Distribution

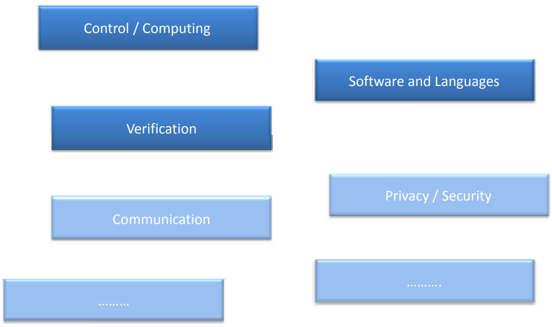
- CPS often networks of interacting elements
- Distributed and decentralized approaches rather than centralized
  - For implementation
  - But also for analysis
- Graph theory



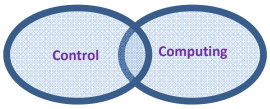
Outline

- The Essence of CPS
- **Challenges**
- Lessons to Learn

CPS Challenges



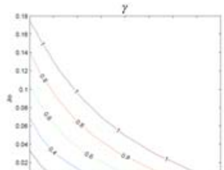
Control/Computing System Challenges



- Temporally robust and fault-tolerant control
  - Scarce resources in embedded and networked control → non-periodic sampling & control
- Feedback-based resource management in computer systems
  - Reduce power consumption

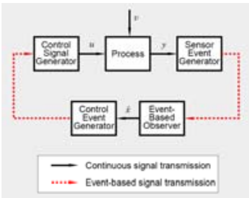
Temporally Robust Control

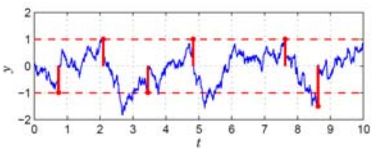
- New jitter stability margin result by Anton Cervin
- Tradeoff between jitter in sampling and jitter in input-output latency
- Sufficient condition based on Structured Singular Value Analysis
- Can be combined with real-time scheduling theory in a nice way



Event-Based Control

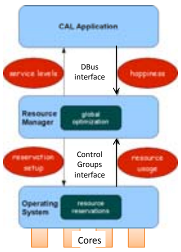
- When control, sensing, actuation, or communication is “expensive”
- Only act when needed, e.g., error threshold crossed
- Theoretically quite involved
- However, some results can be obtained
- Work by Cervin & Henningsson

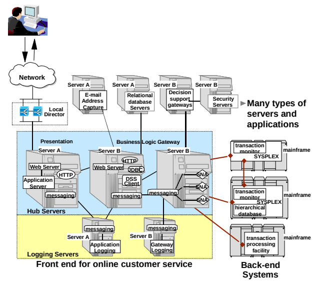




Adaptive Resource Management

- Data servers and centers
- Embedded systems





CPS Challenges

Control / Computing

Software and Languages

Verification

Privacy / Security

Communication

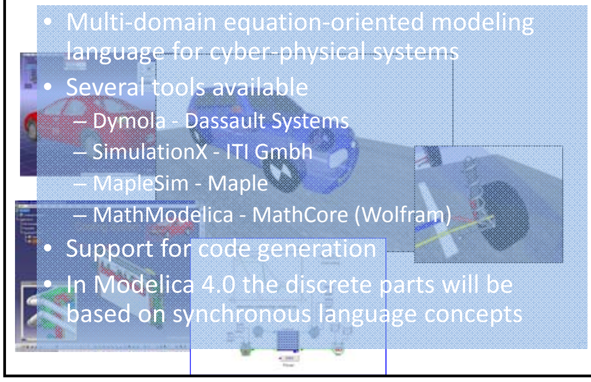
………

………

Software and Language Challenges

- Support for Heterogeneity
  - CPS are heterogeneous
  - Mix multiple models of computations in a semantically well-defined way
  - E.g. Ptolemy II
- Languages that cover both the physical domain and the computing domain
  - For modeling, analysis, verification, simulation, optimization, code generation
  - E.g. Modelica

Modelica



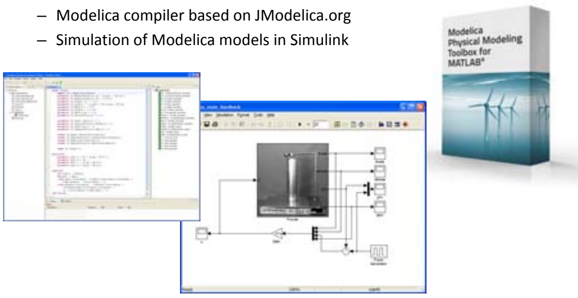
- Multi-domain equation-oriented modeling language for cyber-physical systems
- Several tools available
  - Dymola - Dassault Systems
  - SimulationX - ITI Gmbh
  - MapleSim - Maple
  - MathModelica - MathCore (Wolfram)
- Support for code generation
- In Modelica 4.0 the discrete parts will be based on synchronous language concepts

JModelica.org

- Open source project (www.jmodelica.org) based on thesis by Johan Åkesson
- Beyond simulation
  - Optimica language extension
  - Large-scale dynamic optimization
    - Grade change optimization
    - Polyethylene plant

Modelica Physical Modeling Toolbox for Matlab

- Modelica for Matlab from Modelon
  - Physical modeling with Modelica in the Matlab/Simulink environment
  - Ideal for plant modeling in control systems development
  - Modelica compiler based on JModelica.org
  - Simulation of Modelica models in Simulink



Software and Language Challenges

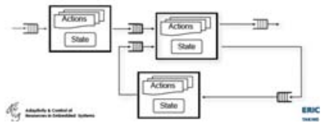
- Support for time
  - Time is central in CPS
  - But the temporal support in most programming languages is small

Software and Language Challenges

- Support for "space"
  - CPS systems are inherently distributed and parallel in nature
  - Strong focus towards parallelism in computing hardware (multicore, manycore, FPGA)
  - Suitable abstractions are needed in also programming languages

Software and Language Challenges

- Languages that better match a parallel execution platform
  - E.g. dataflow / actor languages



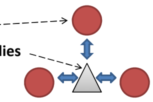
  - Parallelism explicit
  - "Small" languages → Code transformations to fit the execution platform are possible
  - Signal processing and media processing applications

Software and Language Challenges

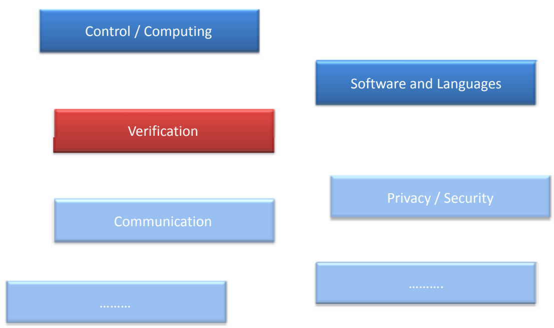
- Languages with built-in hardware abstractions
  - E.g.
    - Sensors and actuators
    - Processors
    - Memory hierarchies
      - Shared, caches, scratchpads
    - Buses and networks

Software and Language Challenges

- Better support for composability
  - Remove built-in, hidden dependencies
  - Make assumptions explicit
  - Separate:
    - Computation, Configuration, Coordination
  - E.g.:
    - PalCom - Platform, middleware, for ad-hoc combination of distributed devices, services
      - EC project : http://www.ist-palcom.org/
      - Computation within **services**
      - Coordination and coordination within **assemblies**
      - Mobile service robotics project
    - Service-oriented architectures (SOA)
  - Property-preserving composability



CPS Challenges



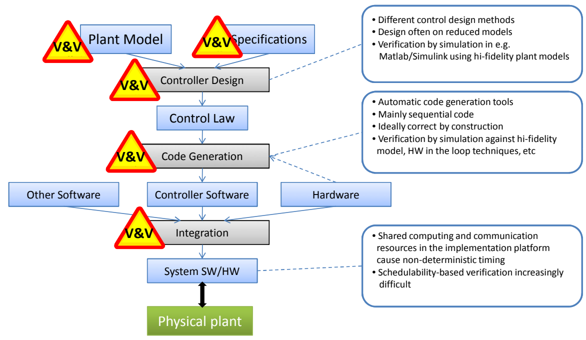
Verification Challenges

- Verification of the entire system including the physical, not only the SW/HW part
  - Hybrid verification
  - Often our main interest is specifications related to the physical system
- Formal verification originates mostly from CS
  - Verification of hardware and/or software systems
  - Models used in analysis very close to reality
  - Worst-case deterministic analysis

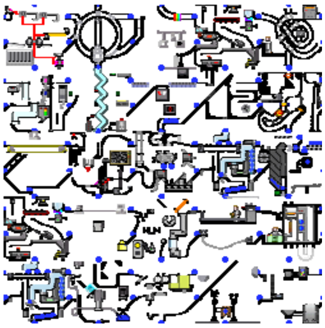
Verification Challenges

- Verification of physical systems are quite different in nature
- Models not as close to reality
  - Un-modeled dynamics
  - Un-modeled nonlinear phenomena (friction, backlash, slip, ...)
  - External disturbances
- Verification through extensive simulation plays an important role
  - High-Fidelity models
  - Hardware in the loop / Software in the loop
- Statistical verification
- It is likely that this will be the case also for CPS
- We do not really know how to best combine the two verification approaches

Embedded Control System Design



Complexity



31

Outline

- The Essence of CPS
- Challenges
- **Lessons to Learn**

Industrial Process Control

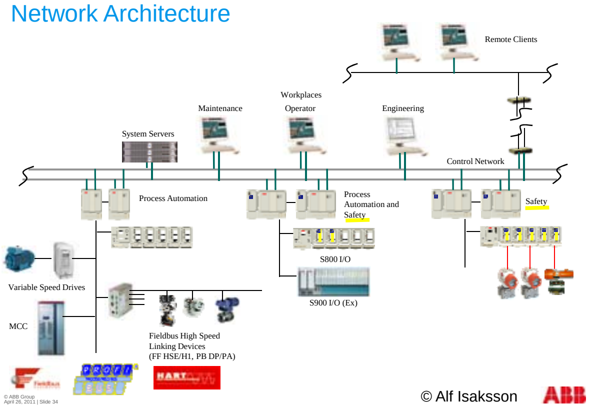
"A cyber-physical system (CPS) is a system featuring a tight combination of, and coordination between, the system's computational and physical elements." (Wikipedia)

"Cyber-Physical Systems are integrations of computation, networking, and physical processes. Embedded computers and networks monitor and control the physical processes, usually with feedback loops where physical processes affect computations and vice versa." (CHESS, Berkeley)

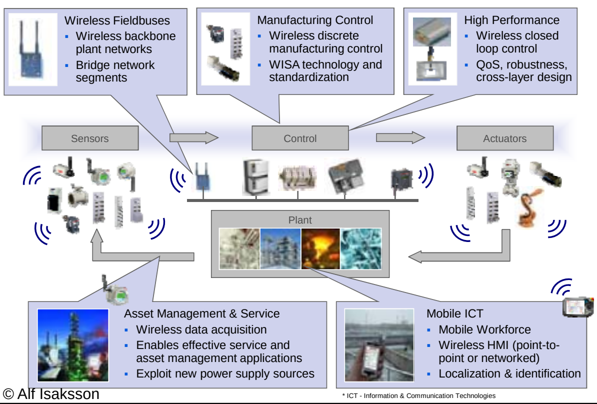
CPS properties:
• Typical CPS applications are large-scale and complex
• Controlled sub-systems with local interactions affecting global performance
• Distributed sensing, actuation, and control

- All this also applies to process control!
  - CPS != Process Control
- An area with a long history
  - Much to learn from

Distributed control system today
Network Architecture



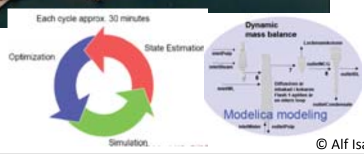
Industrial Wireless applications



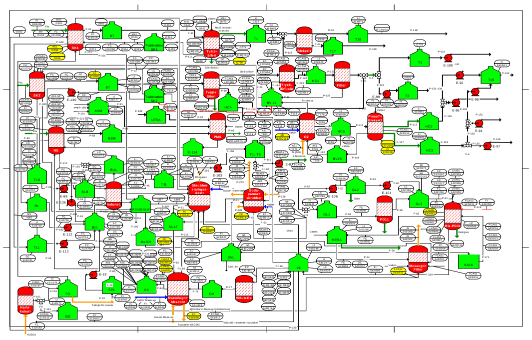
* ICT - Information & Communication Technologies

On-Line Plant-Wide Optimization

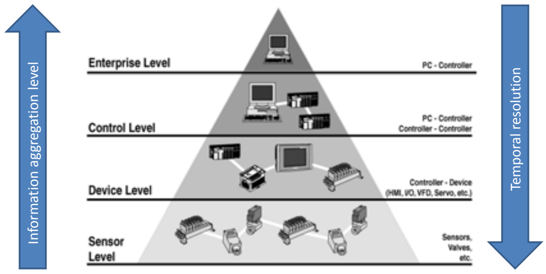
25 Production units
38 Buffer tanks
250 Streams
250 Measurements
2500 Variables



On-Line Plant-Wide Optimization



Automation Pyramid



Hierarchical approach

Conclusions

- CPS > the combination of control, computing and communication
  - Resource constraints
  - Pervasiveness
  - Mobility and wireless
- CPS != Process Automation
  - But, they are quite close
  - And CPS can learn