Taming the *flexibility versus safety* challenge in Distributed Embedded Systems

Luis Almeida

IEETA – Electronics and Telematics Eng. Inst., Aveiro, Portugal
DEEC – University of Porto, Porto, Portugal

Embedded Communication Workshop
TUBS – CITY Symposium
3rd of July, 2009
Braunschweig, Germany

Definitions

- **Safety**
  - Property of a computer system for which reliance can justifiably be placed in the **absence of catastrophic accidents** (mishaps).
    (Laprie, 1994)
  - Well defined, e.g., IEC 61508 and its 4 SIL levels
- **Flexibility ??**
  - "a **ready capability** to **adapt** to new, different, or changing requirements"
    (Merriam-Webster on-line)
  - Within embedded systems, ***flexibility*** can apply to both **design-time** and **run-time** features

Motivation

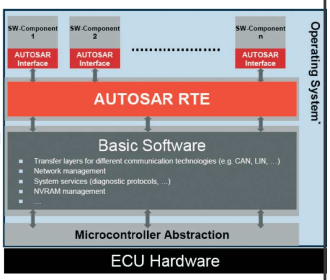
- **Typical design of safety-critical systems**
  - Adoption of **static (design-time) techniques**
    - Maximize *a priori* knowledge
    - Single well-defined operational mode
    - Simplify error detection and fault tolerance
    - Facilitate replica determinism (for modular redundancy)
  - Downside:
    - **Modular redundancy** and hooks for **future changes** imply **over-provisioning** of resources (**exclusive use**)
      - Low efficiency in resource usage → High costs
    - On-line resource reallocation typically impossible
  - **Could we apply *resource sharing* to increase flexibility and efficiency thus decreasing costs?**

Motivation

- **What can we do with *resource sharing*?**
  - **Share ECUs / networks across subsystems**
    - Reduce active components
    - Requires SW components that are unaware of HW
    - Increase design flexibility (more options for OEMs ...)
    - Purpose of **Autosar, IMA, IEC61499**, ...
  - **Multiple operational modes**
    - Modes change on-line using overlapping resources
  - **Promote functional integration**
    - Create new functionality by integrating subsystems
    - Centralized control

Design-time

Run-time



Motivation

- **And also**
  - **Post-deployment reconfiguration**
    - Adding/changing subsystems at anytime
      - e.g., adding SW components using resources that are free
  - **Graceful degradation**
    - Replicas with different QoS using remaining resources
      - Unintentional redundancy
      - Low-cost fault tolerance
  - **Dynamic QoS management**
    - Allocating resources bandwidth dynamically as needed
    - Applications must be able to execute with different QoS levels provided by the infrastructure
      - Different **modes per subsystem** (including *standby*)

All run-time

Problem

- **Can we reconcile flexibility and safety?**
  - **Unforeseen and unsafe changes can occur because of**
    - Insufficient resources
    - Coupling between functionality and timing
    - Mutual interference across subsystems
  - Hypothesis: **Yes, if we...**

Rubber band model

Resource bandwidth

dedicated

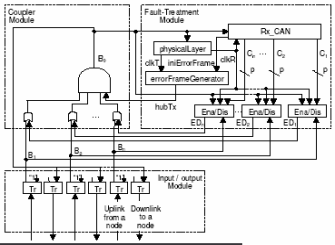
- **Bound flexibility**, i.e., keeping the system within a well defined operational region → strict policing
- Guarantee the **minimum resources needed** for critical tasks
  - **And manage** (share) **the remaining resources**
- Use **components (networks...)** providing needed properties
  - Partitioning (virtualization), policing, QoS management, error confinement, atomic broadcast,...

Our contributions in this topic

- **Boosting error confinement in CAN**
  - **CANcentrate** and **ReCANcentrate**
    - The first CAN hubs designed to improve dependability
- **Towards flexible partitions**
  - **FTT-CAN**, **FTT-SE** (software solutions)
    - Fully configurable ET/TT partition
    - Support for further partitioning: **Server-CAN, Server-SE**
      - Confine errors in the temporal domain
    - Support for flexible modes
      - Modes defined at subsystem level, independently
  - **FTT-switches** (hardware solutions)
    - HaRTES (Hard Real-Time Ethernet Switch)
    - Partitions enforced by the switch (a boost on robustness)

CANcentrate

- **First CAN-hub designed for error-confinement**
  - **Wired-AND of CAN bus replaced by logical AND**
  - **Uplinks separated from downlinks**
  - **Allows fast detection of several types of errors**
    - ***Link isolation when error threshold crossed***
      (latency to isolate stuck-at or bit-flipping faults: 73µs, 150...600µs)
    - ***Automatic reintegration after error-free period***
      (latency to reintegrate isolated links: 5.2ms)
- **Works with COTS CAN controllers and any existing application**
  - It is just a wiring replacement

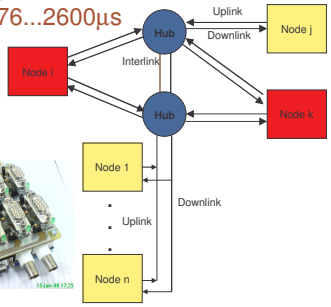


Manuel Barranco, Julian Proenza, Luis Almeida. Boosting robustness in CAN systems with new star topologies: CANcentrate and ReCANcentrate, IEEE Computer, 42(5): 66-73, May 2009.

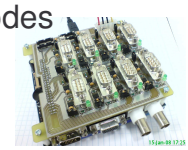
ReCANcentrate

- First replicated CAN-hub architecture
  - Targets very demanding safety requirements
  - Replicated hubs are synchronized bit-by-bit
  - Made by two interconnected CANcentrate hubs
    - ***Hubs can isolate / reintegrate one another***

Latency to isolate stuck-at errors: 216µs
Latency to isolate bit-flipping errors: 476...2600µs
Reintegration latency: 4.2ms

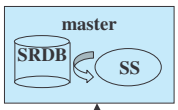


- Supports **mixed architectures** with critical / non-critical nodes as well as bus segments



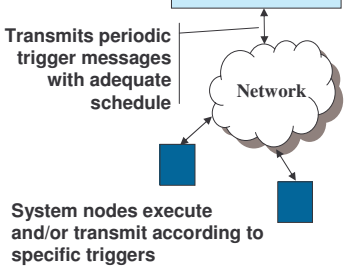
FTT-xx (software)

The **FTT** protocols provide a **synchronous framework** for distributed applications that, nevertheless, is amenable to **operational flexibility** as required for on-line adaptation, on-line reconfiguration,... but **under adequate control**



Examples:

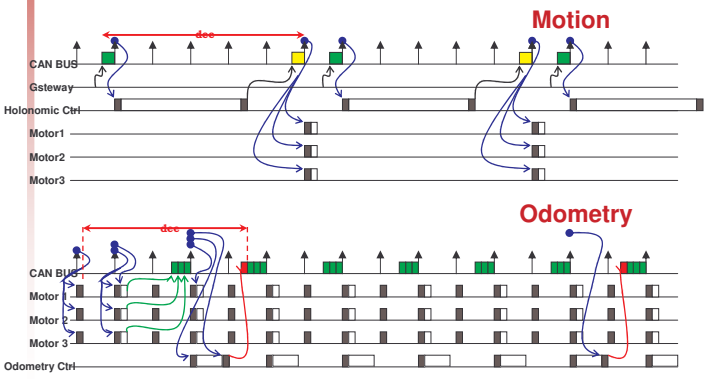
- ✓ Supporting **adaptation** in **control applications** (reacting to overloads)
- ✓ Creating **virtual channels** with guaranteed QoS, even with varying load (partitioning)

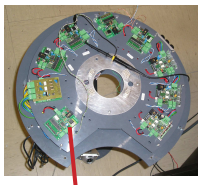


F-T-T-
Flexible Time-Triggered architecture

FTT-CAN – timing control

- **Flexibility to**
  - Separate **functionality from timing**
  - **Update timing easily** (phases)

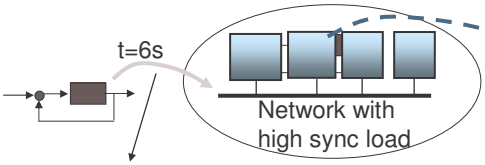




V. Silva, R. Marau, L. Almeida, J. Ferreira, M. Calha, P. Pedreiras, J. Fonseca. Implementing a distributed sensing and actuation system: The CAMBADA robots case study. ETFA 2005, Catania Italy, September 2005.

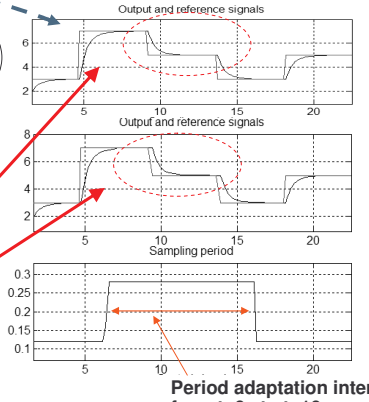
FTT-CAN – rate adaptation

- **Flexibility to**
  - **Adapt rates** of control loops dynamically (e.g., for overloads)



- **Rejected** with **fixed h**
- **Accepted** with **variable h**

h	ISE	Degrad.	BW
0.12s	48.2	0	1
0.12s / 0.28s	50,3	4,4%	0,66



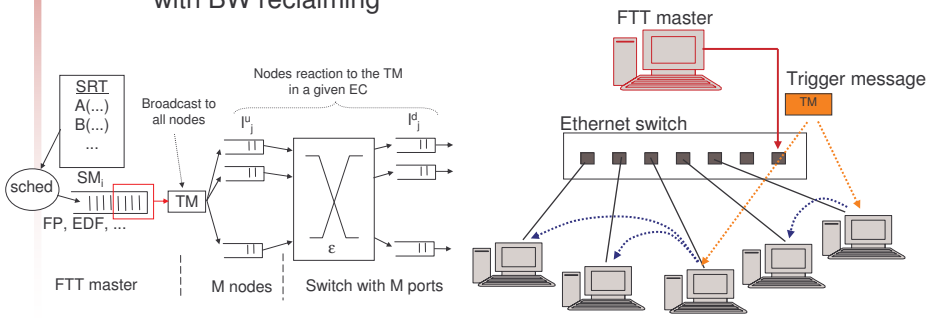
Period adaptation interval from t=6s to t=16s

A. Antunes, P. Pedreiras, A. Mota, L. Almeida, ETFA 2005, ANIPLA 2006, INDIN 2007

FTT-SE – load control

- ✓ **Flexibility to**
  - ✓ Support **any scheduling** policy
  - ✓ Support **dynamic QoS** management
  - ✓ Create **partitions** (virtual channels)
  - ✓ **Seamlessly integrate** sync/async/NRT traffic with BW reclaiming

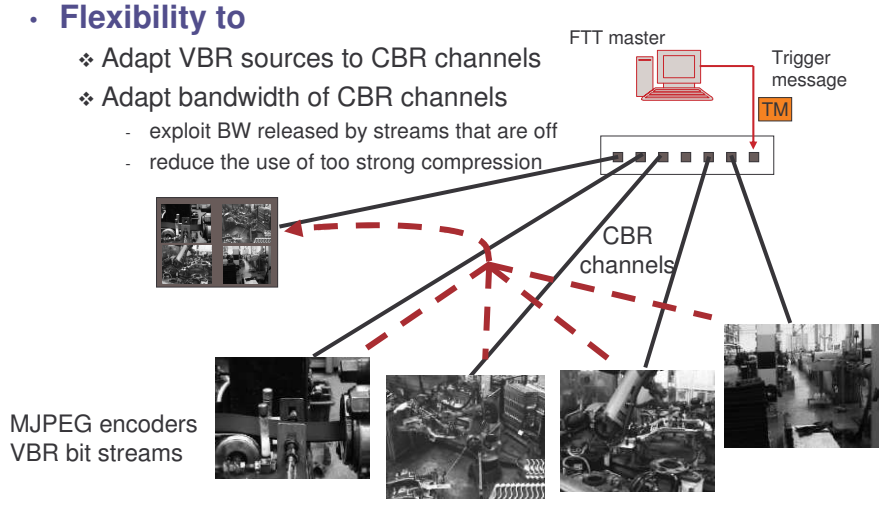
Traffic is **scheduled per cycles** considering links capacity. **Memory overloads are eliminated**



R. Marau, L. Almeida, P. Pedreiras. Enhancing Real-Time Communication over COTS Ethernet switches. WFCS 2006, IEEE 6th Workshop on Factory Communication Systems, Turin, Italy. June 2006.

FTT-SE – QoS adaptation

- **Flexibility to**
  - ❖ Adapt VBR sources to CBR channels
  - ❖ Adapt bandwidth of CBR channels
    - exploit BW released by streams that are off
    - reduce the use of too strong compression



J. Silvestre, L. Almeida, R. Marau, P. Pedreiras. Dynamic QoS Management for Multimedia Real-Time Transmission in Industrial Environments. ETFA 2007, Patras, Greece, September 2007.

Luis Almeida U.PORTO ieeta

FTT-SE – QoS adaptation

5 Mbit/s
V
1 Mbit/s

after 20s

1 Mbit/s
V
5 Mbit/s

6Mbit/s at all times

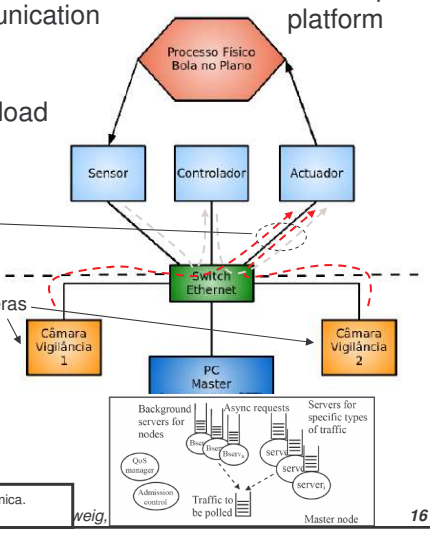
Luis Almeida U.PORTO ieeta

FTT(Server)-SE – partitions

- **Flexibility to**
  - Support aperiodic RT communication with servers (e.g., sporadic)
  - Enforce temporal partitions resilient to strong interfering load

(see videos)

Ball on plane platform



Total load varied between 18Mbit/s and 91Mbit/s

1 short control packet together with 2 long video frames

Surveillance cameras

Without isolation, too high load causes control packet losses and large jitter

Nuno Figueiredo. Controlo Distribuído de Plataformas para Experiências de Mecatrónica. Master Dissertation. University of Aveiro, July 2008. (in Portuguese)

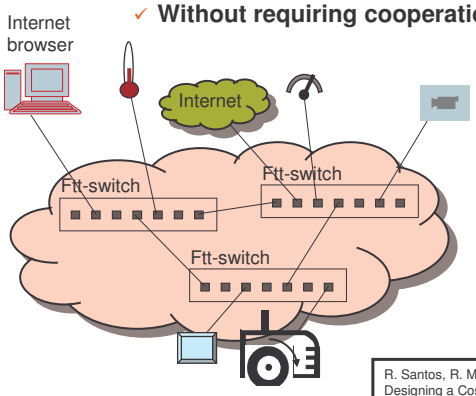
weig,

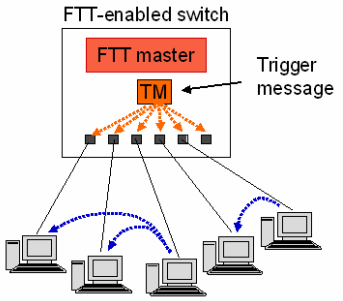
16

FTT-xx (hardware)

Integrating FTT features in the switches results in **timeliness, flexibility and high robustness**

- ✓ Enforce negotiated channel characteristics (**Policing**)
- ✓ Confine specific traffic to separate windows (**Partitioning**)
- ✓ **Without requiring cooperation of end-nodes**



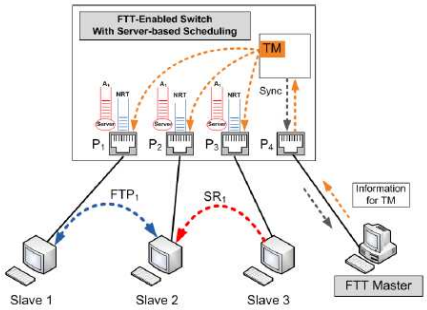


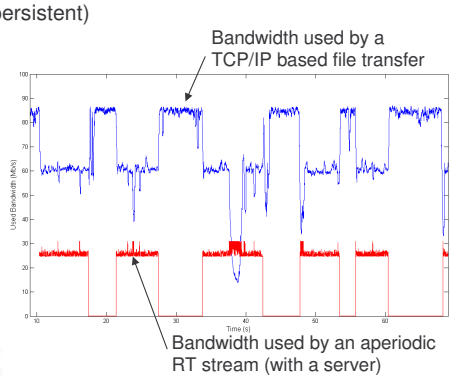
R. Santos, R. Marau, A. Oliveira, P. Pedreiras, L. Almeida. Designing a Costumized Ethernet Switch for Safe Hard Real-Time Communication. WFCS 2008. Dresden, Germany. 21-23 May 2008.

HaRTES – server-based scheduling

- **Flexibility to**
  - ❖ Enforce temporal partitions with efficient servers (e.g., sporadic)
    - That support BW reclaiming by NRT traffic (background).

- ✓ 1 **real-time aperiodic** stream (switched on and off, handled with a server)
- ✓ 1 **non-real-time stream** (bursty and persistent)
  - ✓ **Large file transfer**
- ✓ Sent to the same output link





Conclusions

- **Resource sharing** is a key to build more **resource efficient** distributed embedded systems
- Further benefits may arise from **Dynamic reconfiguration** and **QoS management** to always adjust resource usage to current needs
- Using these techniques in **safety-critical** systems requires
  - Strong protection mechanisms across resource partitions
  - Prompt temporal control for fast adaptations of resource allocations
- We have shown **several networking components** that provide such features and can potentially open the way to **flexible safety-critical** applications (thus less expensive)
  - **(Re)CANcentrate, FTT(Server)-xx**