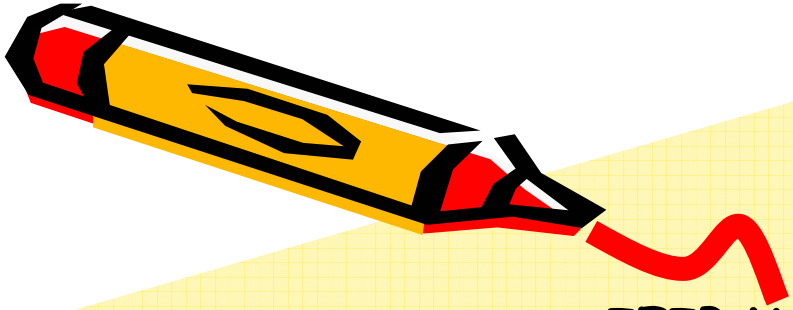


IFIP WG10.4 Workshop
June 2008 – Alaska- USA

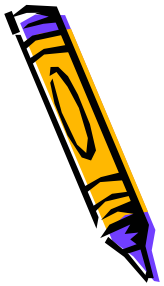
Dependable Ambient Computing

An example with Car2Car Coordination

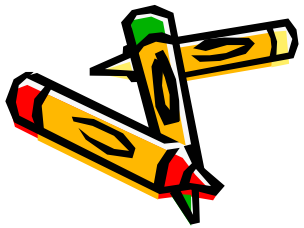
Paulo Esteves Veríssimo

Navigators Research Group,
LaSIGe, Laboratory for Large-Scale Informatic Systems
Univ. Lisboa
pjv@di.fc.ul.pt
http://www.di.fc.ul.pt/~pjv

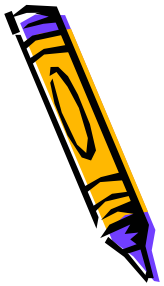
Status-quo w.r.t. 'Embedded Systems'



- TODAY: Several families of "unrelated" systems:
  - Embedded systems (e.g. cars)
  - Ubiquitous computing systems (e.g. mobile phones)
  - Peer-to-peer and ad-hoc networked systems
  - Pervasive systems and gadgets (home, office, active environment)
- TOMORROW: Bound to become akin, under an Ambient Computing vision



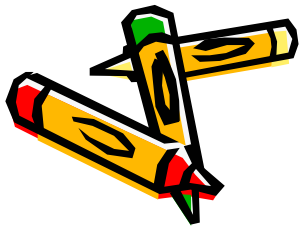
Mirror on the wall: the Challenge



- What is the next grand challenge in the road ahead, for Embedded Systems research?
- To master complexity, modularity, autonomy, dynamics of configurations, heterogeneity of compositions, pervasiveness of devices, ubiquity of computations, uncertainty of timeliness, predictability of behaviour (sec&dep, QoS)
- In other words, think about:

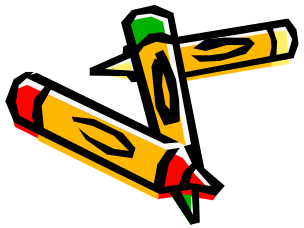
  Complex R/T systems of embedded components

Dependable



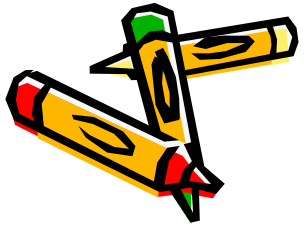
Meeting the Challenge

- The right programming model
- The right architecture
- The right algorithms



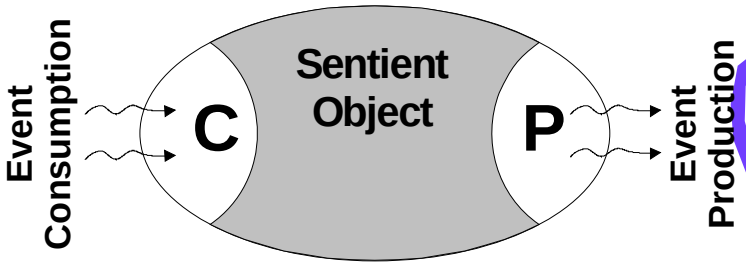
A few examples

- Uncertainty versus predictability
- Dependable adaptation
- Sentient objects models
- Anomalous behaviour
- Generic events architectures
- Fault+intrusion tolerance
- Trusting trustworthy components



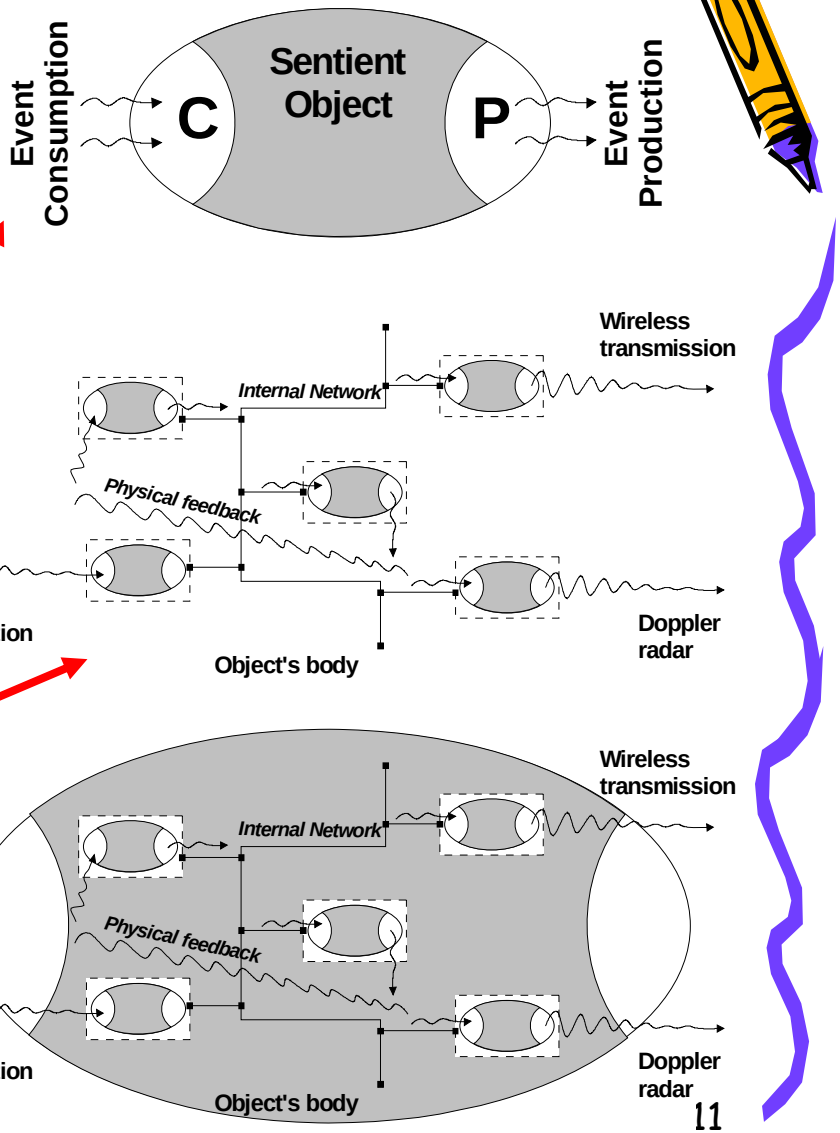
Component-based Sentient Object Model

- Recall:
  - R/T objects [Kim&Kopetz 94]
  - Sentient computing [Hopper 99]
- Sentient objects [Cortex 02]:
  - objects that accept input events from a variety of different sources, process them, produce output events
  - actuate on the environment and/or interact with other objects
- Component-based object model:
  - maybe simply software-based components,
  - but can also comprise mechanical and/or hardware parts
  - e.g. mechanics, sensorial apparatus that substantiates ``sentience''

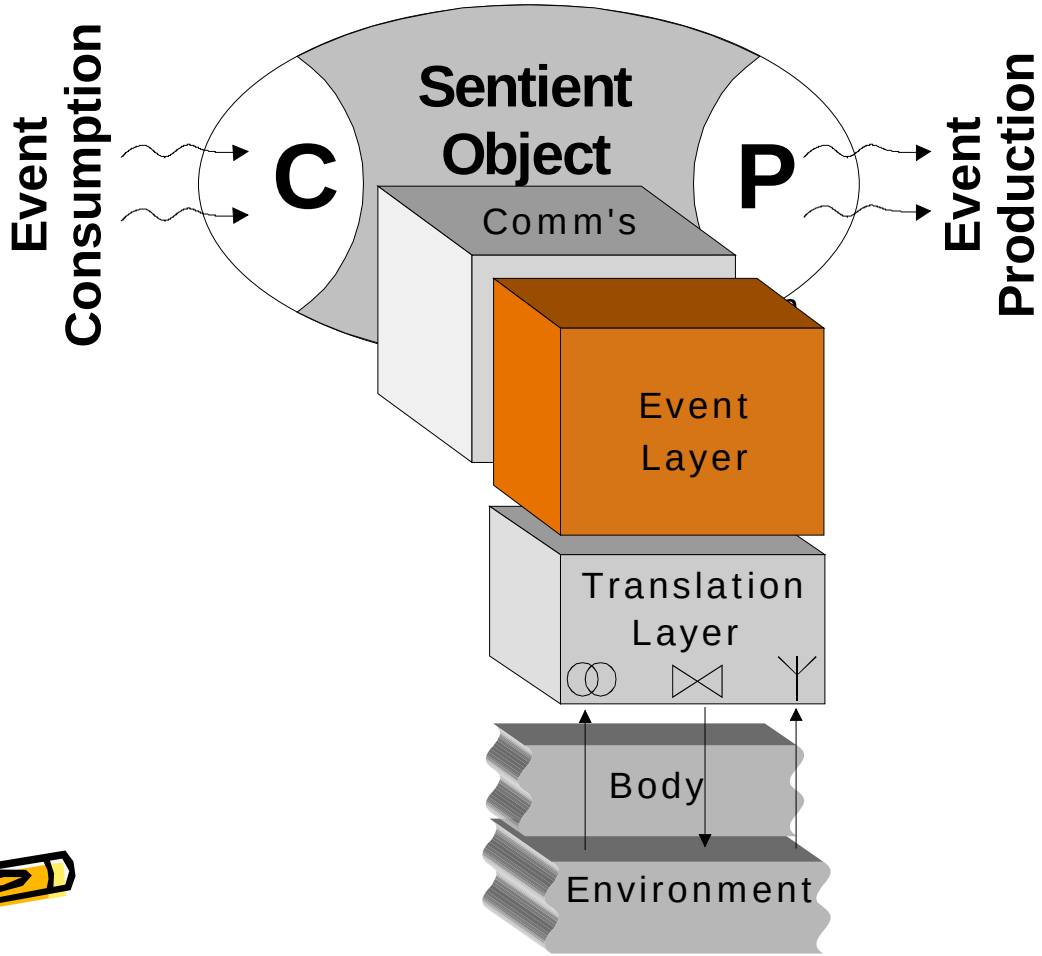


Composability of sentient objects

- Component-aware sentient object composition
  - clustering
  - hierarchical composition
  - normally constrained by the actual hardware component's structure
- To provide an example, imagine a robot and manipulator controllers:
  - see each controller + control software as a sentient object
  - imagine structure and interconnnections
  - see robot itself as a (composite) sentient object, of the controller
  - objects plus all the robot hardware (body)



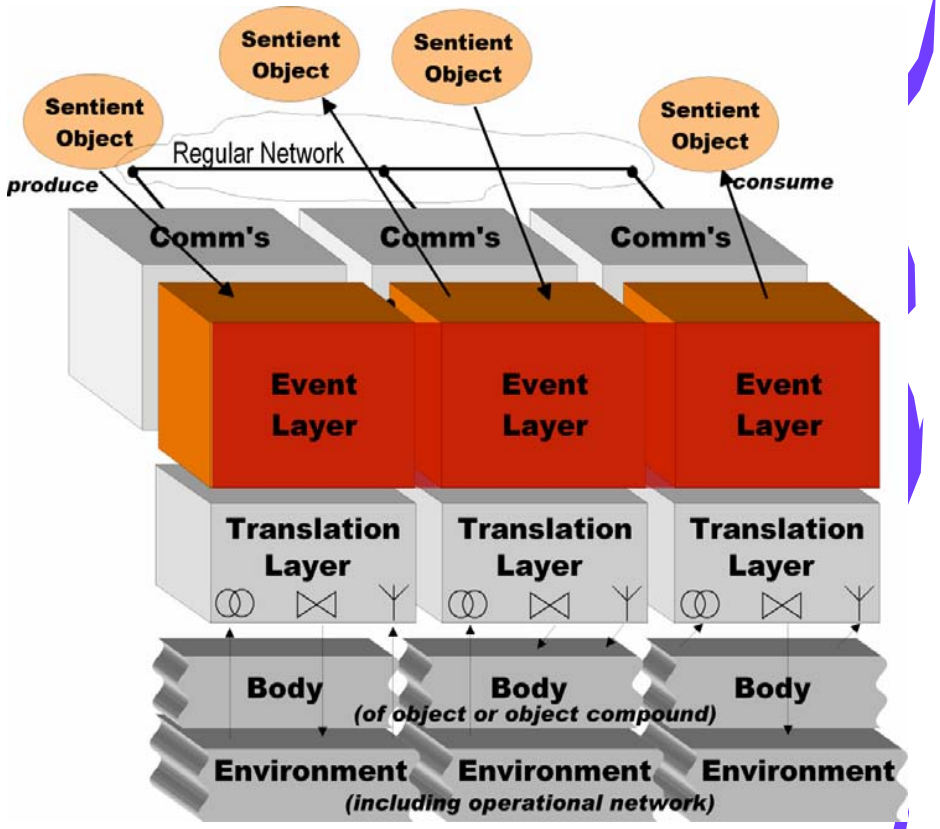
Generic architecture and middleware for component-based embedded systems



GEAR/COSMIC architecture

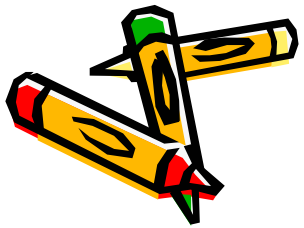
L-shaped structure

- [Environment]
  - physical surroundings of sentient objects
- [Body]
  - physical embodiment of a sentient object
- [Translation Layer]
  - layer responsible for physical event transformation from/to their native form to Event Channel (EC) dialect
- [Event Layer]
  - layer responsible for event propagation in whole system
- [Communication Layer]
  - layer responsible for 'wrapping' events (in EC dialect) into 'carrier' event-messages
- [Regular Network]
  - support for the Communication Layer



A few examples from research

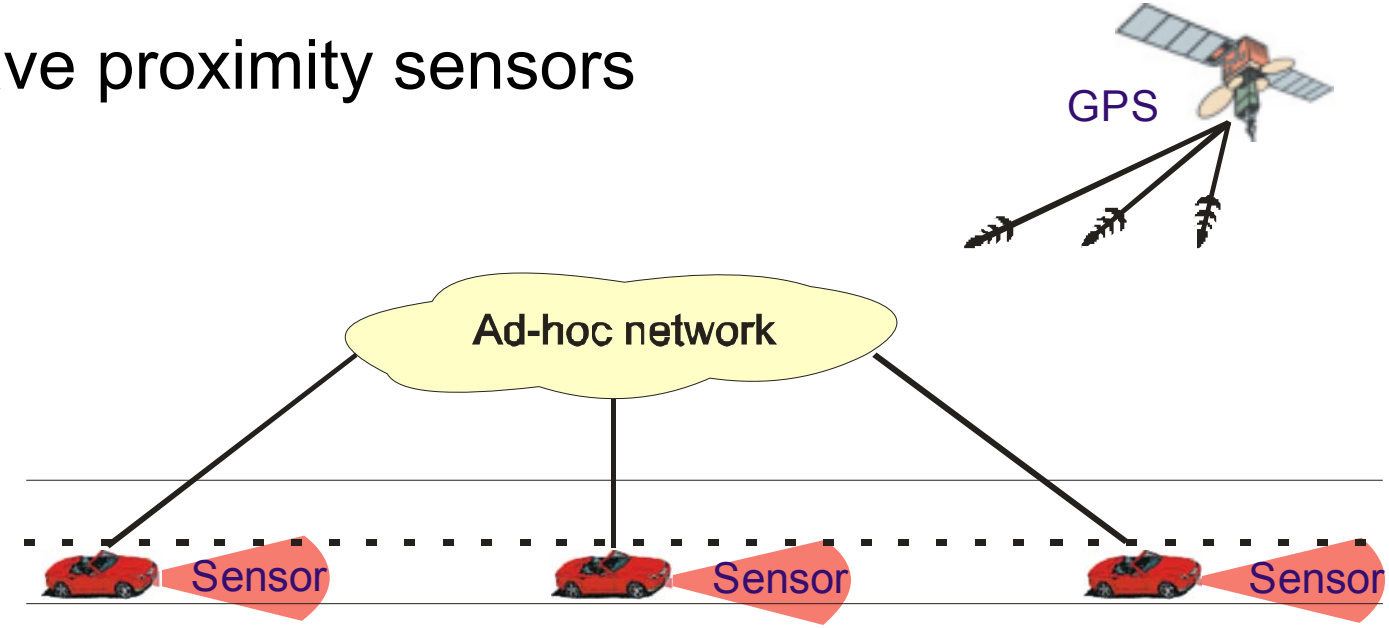
- Uncertainty versus predictability
- Dependable adaptation
- Sentient objects models
- Anomalous behaviour
- Generic events architectures
- Fault+intrusion tolerance
- Trusting trustworthy components



Adaptive and secure real-time control using a hybrid component-based system architecture

Platooning scenario [Hidenets]

- A platoon is composed of cars that:
  - Communicate with each other (ad-hoc network)
  - Receive GPS coordinates from satellites
  - Have proximity sensors



16

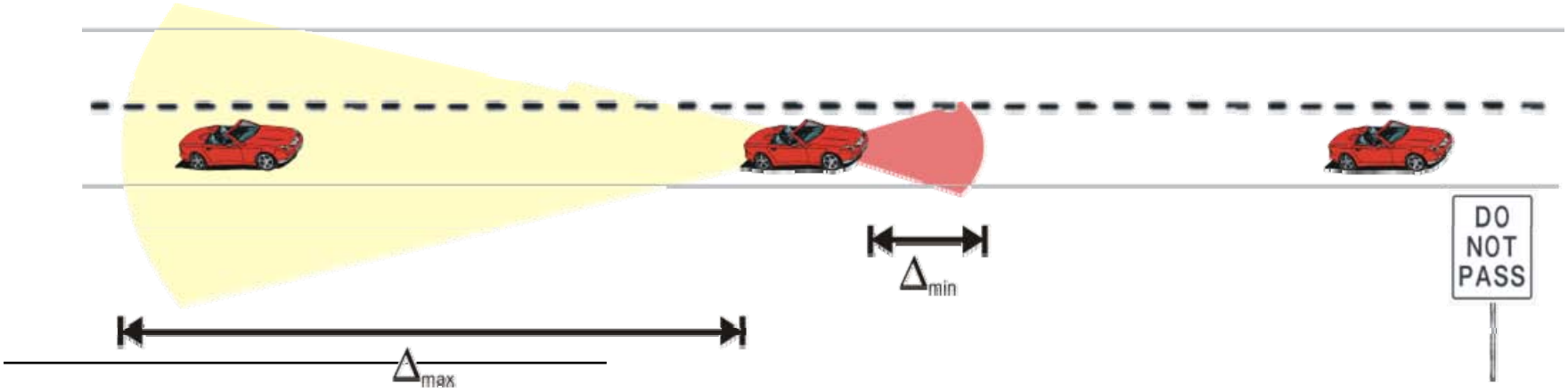
Challenging scenarios

- The motion of a platoon can be affected by:
  - Disturbances in the ad-hoc environment
  - Disturbances in the speed of a vehicle
  - Intentional attacks to the communication

- This can lead to:
  - Platoon with a slower motion
  - Platoon breaking (cars unable to communicate with neighbors)
  - Collisions

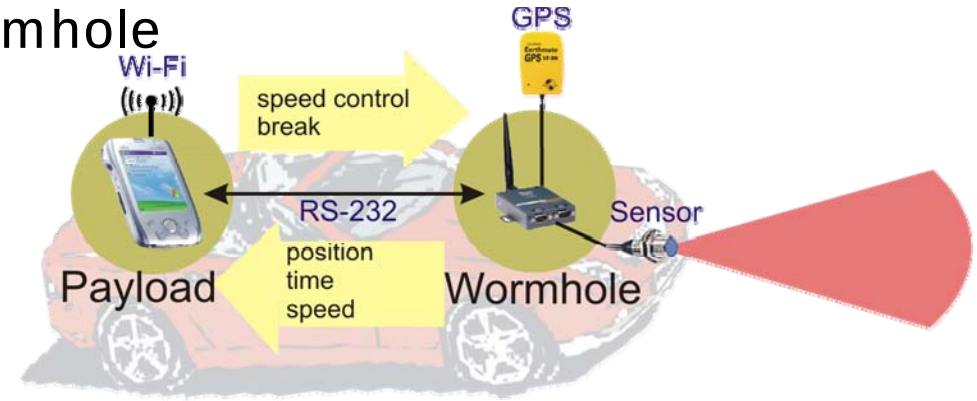
Requirements

- The fundamental requirements are:
  - Safety-critical: Δ_{min} to the front car
  - Non safety-critical: Δ_{max} to the car behind
  - Security-critical: integrity of data (value **AND** time)
  - Fail-safe state is: stop the car



Hybrid Architecture
(coping with different Dependability requirements)

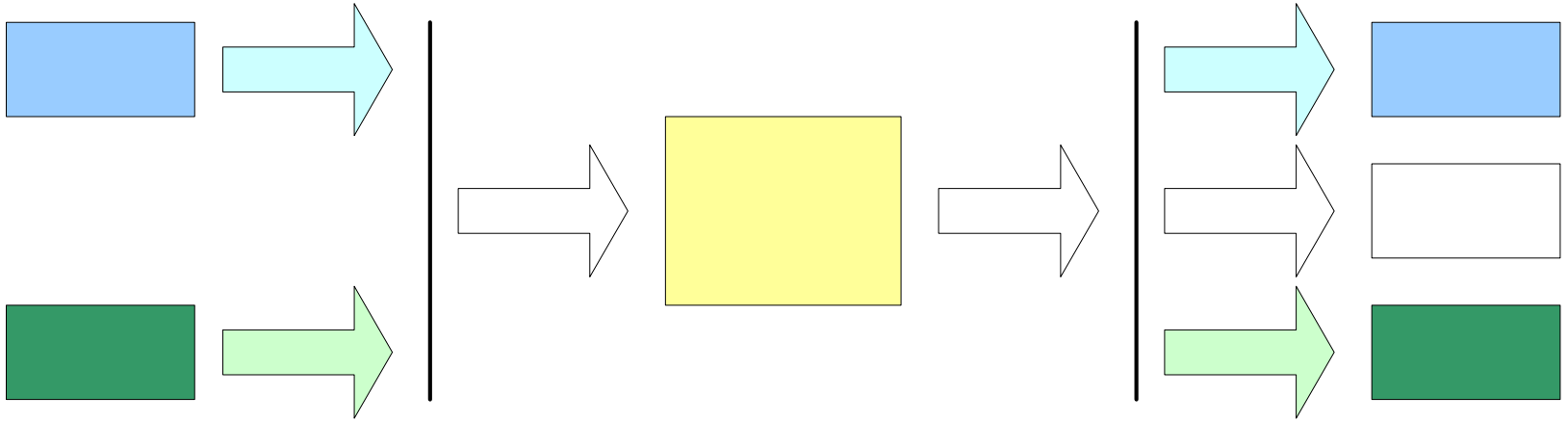
- A car has to have two main components:
  - Control Wormhole
  - Payload



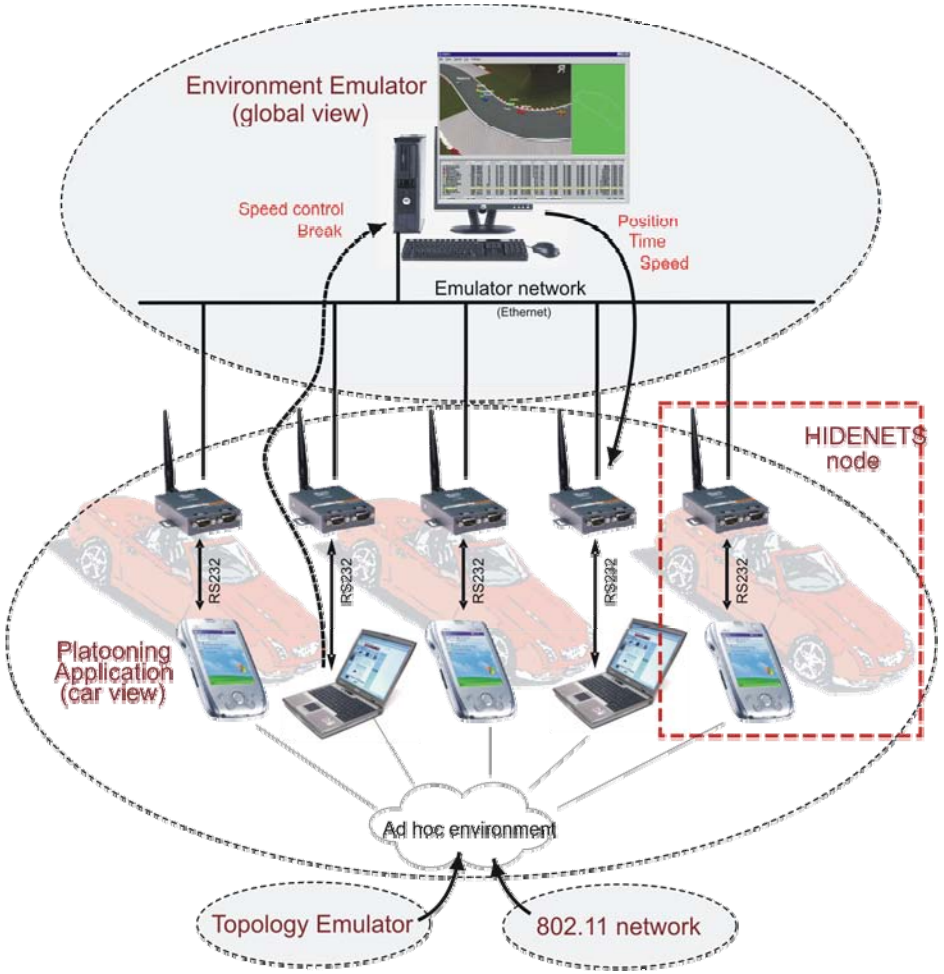
- The **wormhole** provides information to the payload about the position, time and speed. It receives data from:
  - GPS
  - Front sensor
- The **payload**:
  - Communicates with the neighbor cars and
  - Sets the speed.

Platooning dataflow

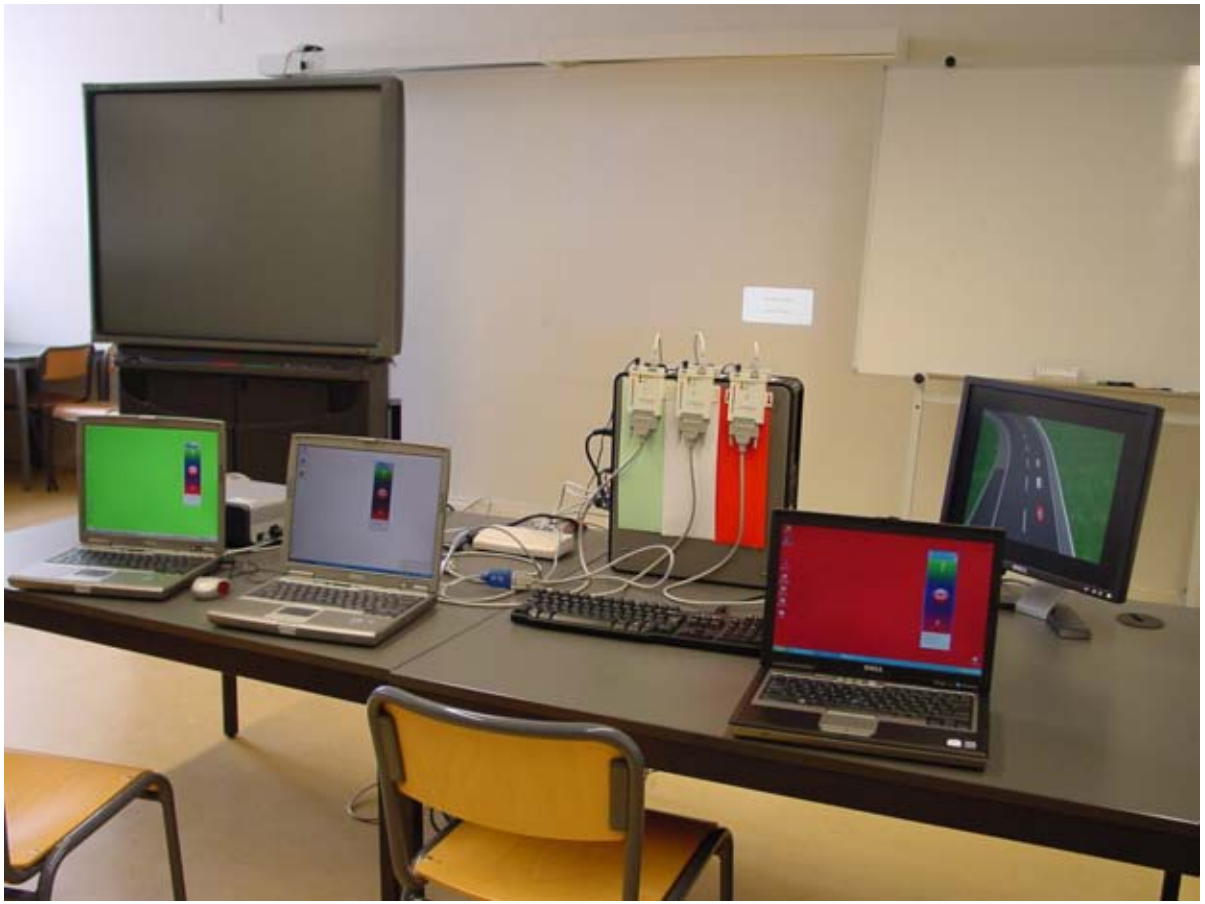
- With the local wormhole: collecting information from car sensors and GPS and also sending control information to car actuators
- With peers (other cars): exchange of context information



Proof-of-concept – Global view



Demo real set-up



Demo real set-up

Demo real set-up

Platoon

Platoon

Relevant results shown in the demo

1. Basic platooning prototype
2. Architectural hybridization using a wormhole
3. Timely detection (and reaction) upon payload timing failures
4. Dependable estimation of communication delays by using input from the QoS Coverage service
5. Secure, intrusion-tolerant speed agreement
6. Improved (more optimistic) platooning behavior, while still ensuring the main safety properties

This solution combines **efficiency** with **safety**. The efficiency comes from the applications that exchange data through the wireless network, while the safety comes from the wormhole that guarantees a low probability of collisions.

Where to find us

Navigators Group at LaSiGE
Univ. Lisboa, Portugal
http://www.navigators.di.fc.ul.pt

related research lines in the site:

Fault and Intrusion Tolerance in Open Distributed Systems

Timeliness and Adaptation in Dependable Systems

Feel free to email:
Paulo Veríssimo --- pjv@di.fc.ul.pt