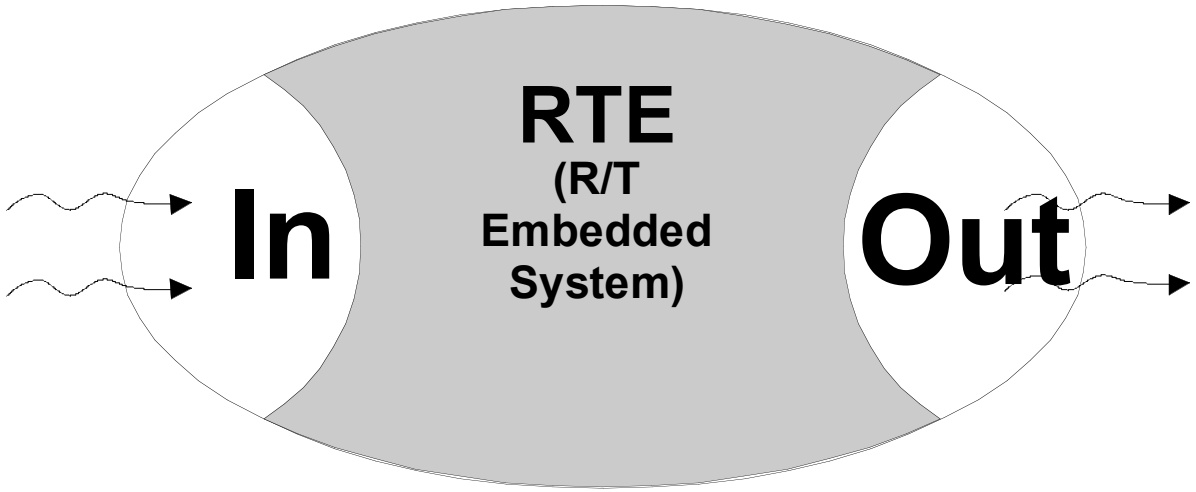
Architectures and Frameworks for Systems of Embedded Systems

António Casimiro
University of Lisbon Faculty of Sciences
LASIGE – Navigators group

casim@di.fc.ul.pt
http://www.navigators.di.fc.ul.pt

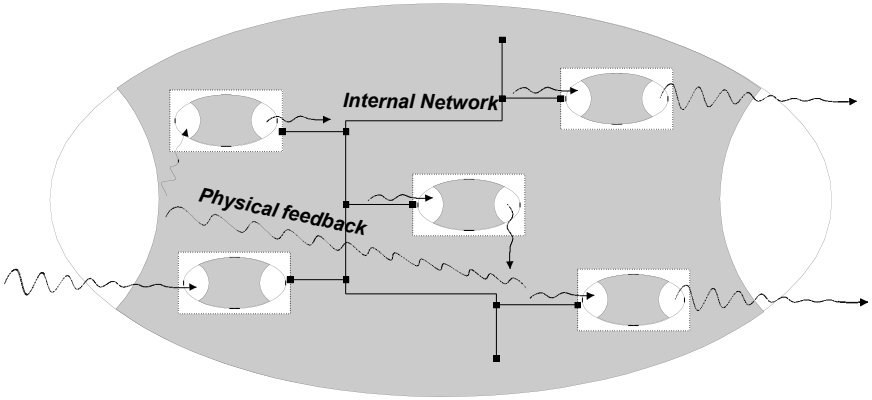
53rd IFIP WG 10.4 Meeting, February 23, 2008

Embedded Systems today: simple, integrated

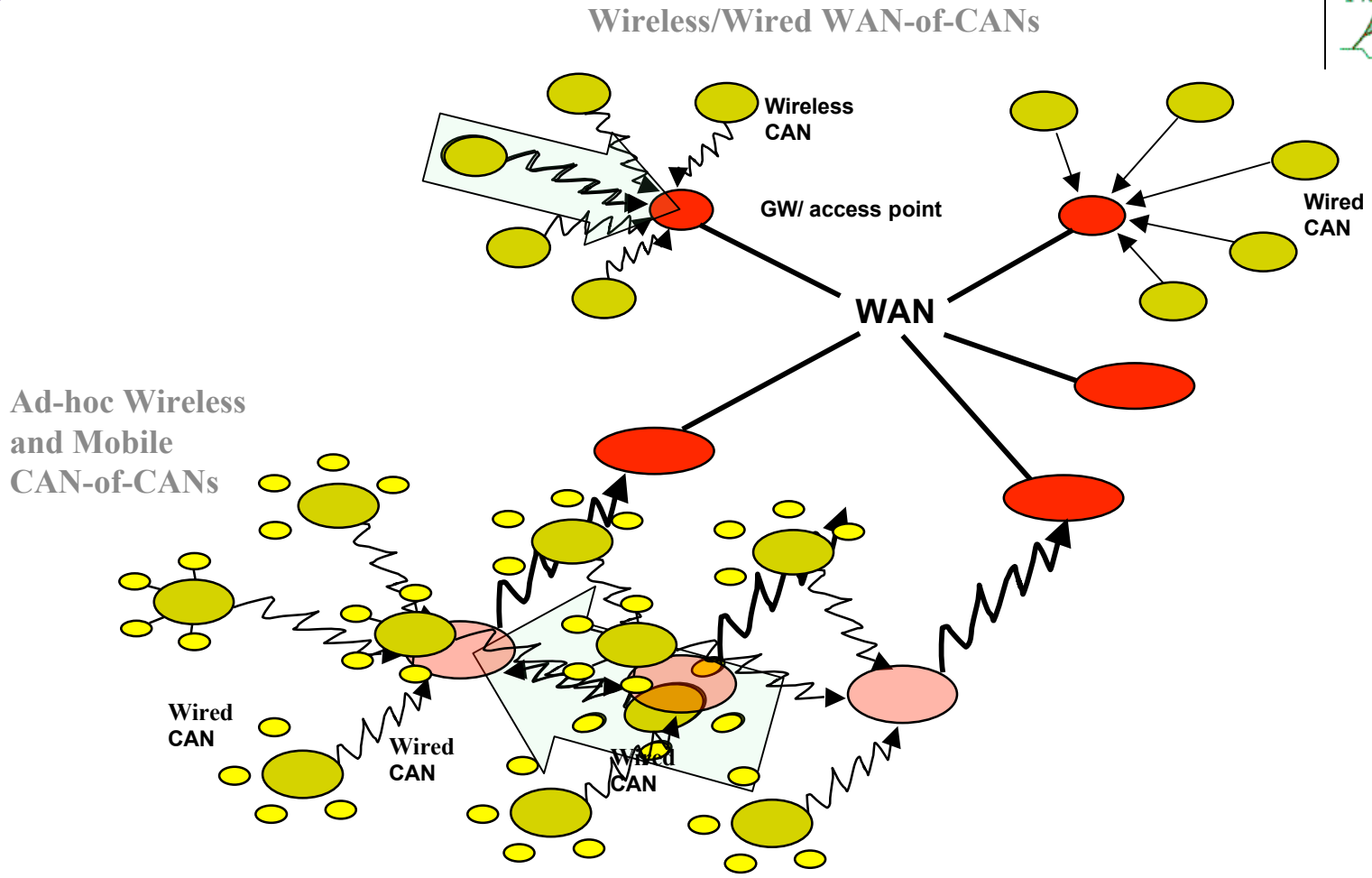


Networked embedded systems: short-range, static

- Take a car or robot
- Local control network



Systems of embedded systems



Challenge

- The design of correct, trustworthy real-time systems of embedded systems is a grand challenge

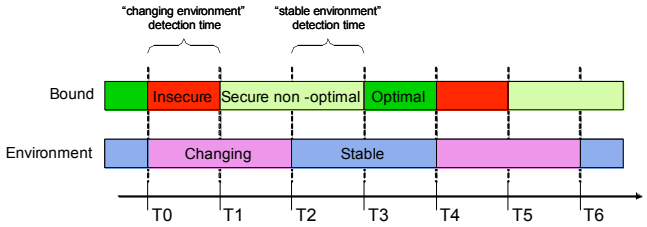
- Need to deal with:
  - complexity, modularity, autonomy, dynamics of configurations, heterogeneity of compositions
- But also
  - pervasiveness of devices, ubiquity of computations, lack of perceived global state, unreliability of communication, uncertainty of timeliness (delays), insecurity

Systems research topics under a distributed systems context:

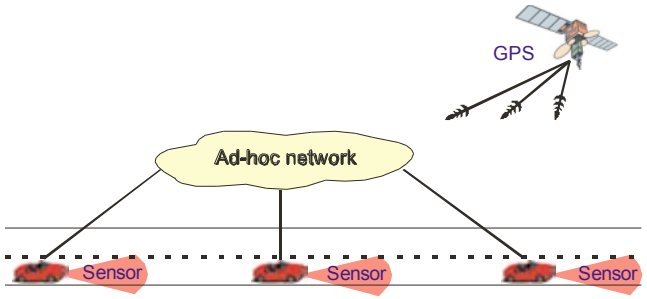
- Reference architectures for pervasive/ubiquitous/embedded systems
- Event-based middleware
- Innovative paradigms featuring adaptation, reconciling uncertainty with predictability
- Adequate programming models to exploit the existing architectural support
- System evaluation

Summary

- Adapting to environment changes



- Improved control with hybrid system architecture



Adapting to environment changes

Problem Motivation

- Design and deployment of distributed applications is faced with the confluence of antagonistic aims: *uncertainty vs. predictability*
- Current and future large, massive-scale pervasive and/or ubiquitous computing systems will amplify this
- Key lies with a changing notion of service guarantees, not with their absence

Dealing with uncertainty

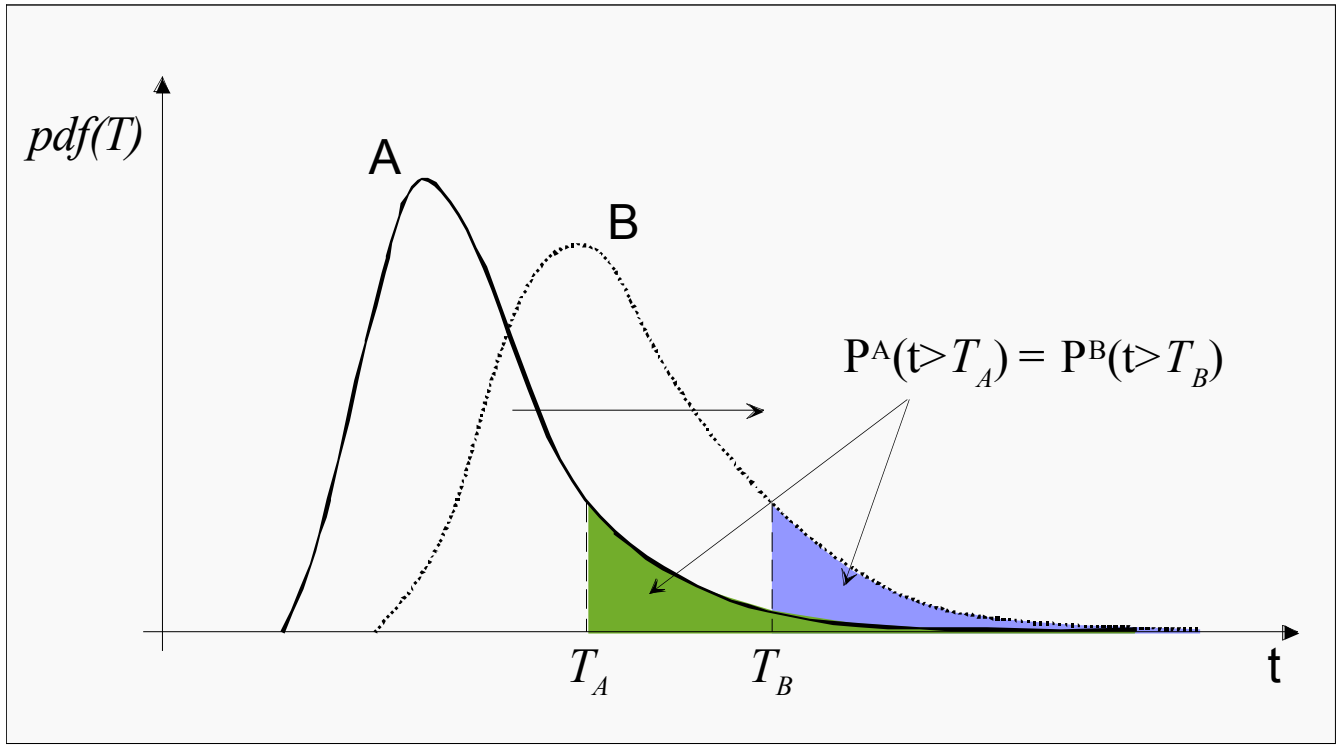
- We defined a generic approach to reconcile uncertainty with the need for predictability:

Dependable adaptation

- Make the application behave [safely, timely, securely, etc] in the measure of what can be expected from the environment
- The objective is to ensure coverage stability

Dependable adaptation

- The chosen bound should be secured with a stable probability



Implementation

- Pessimistic approach (weak model of the environment):
  - The environment is probabilistic, but we cannot know or identify a specific probabilistic distribution
  - Hence, based on estimated mean and variance, we use a pessimistic formulae to calculate an upper bound for the probability of a bound, t, being violated

$$P(D > t) \leq \frac{V(D)}{V(D) + (t - E(D))^2}, \text{ for all } t > E(D)$$

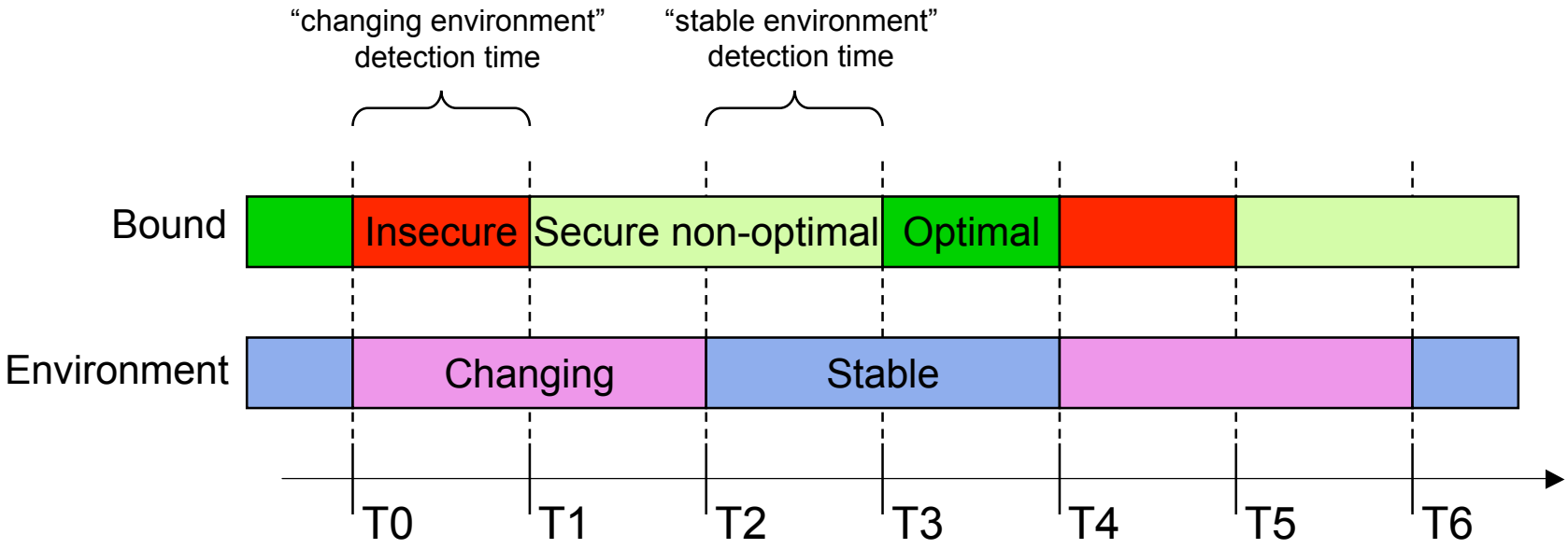
Assumptions

- Characterizing delays
  - Possible distributions (e.g., network behavior) include: Weibull, shifted gamma, exponential, truncated normal
  - Distributions may change over time
  - Assume **interleaved probabilistic behavior**
- Determining actual distributions and changes
  - Several approaches, including: time-exponentially weighted moving histograms, Kolmogorov-Smirnov test or Mann-Kendall test
  - Possibly use several simultaneously
  - Assume **recognition abilities**

Assumptions

- Applying recognition methods in practice
  - Is the recognition process fast enough for the dynamics of the environment?
  - Assume **sufficient stability**
  - Is there enough and statistically related data to allow the recognition of a probabilistic state?
  - Assume **sufficient activity**

How does the environment behaves?



How does it works?

- We need Phase Detection Mechanisms that:
  - Detect stable phases
  - For the detected stable probabilistic distribution estimate parameters that characterize this pdf
- Several phase detection mechanisms can be run in parallel
- Then we need Bound Estimators for each distribution to derive the bound that secures the required coverage

How to provide ‘safe’ time bounds?

- During stable periods, use optimistic estimators for determining time bounds, for example:

Exponential

$$bound = \frac{1}{\ddot{e}} \ln \frac{1}{1 - coverage}$$

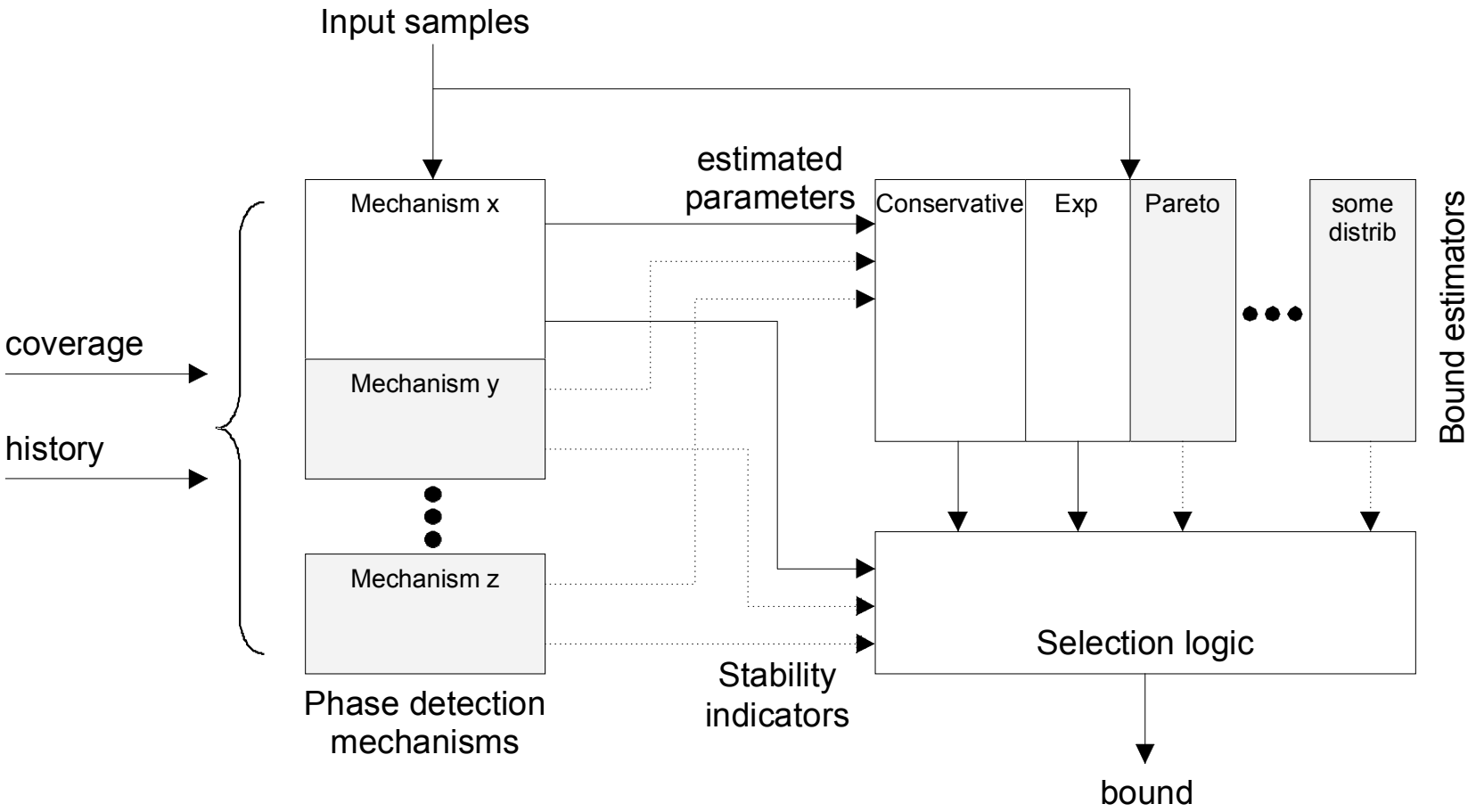
Pareto

$$bound = \frac{k}{\sqrt[\alpha]{1 - coverage}}$$

- During transitions, use a conservative, but safe, estimator (an mentioned before)

The complete framework

(To be presented at SAC'08)



Example for exponential distribution

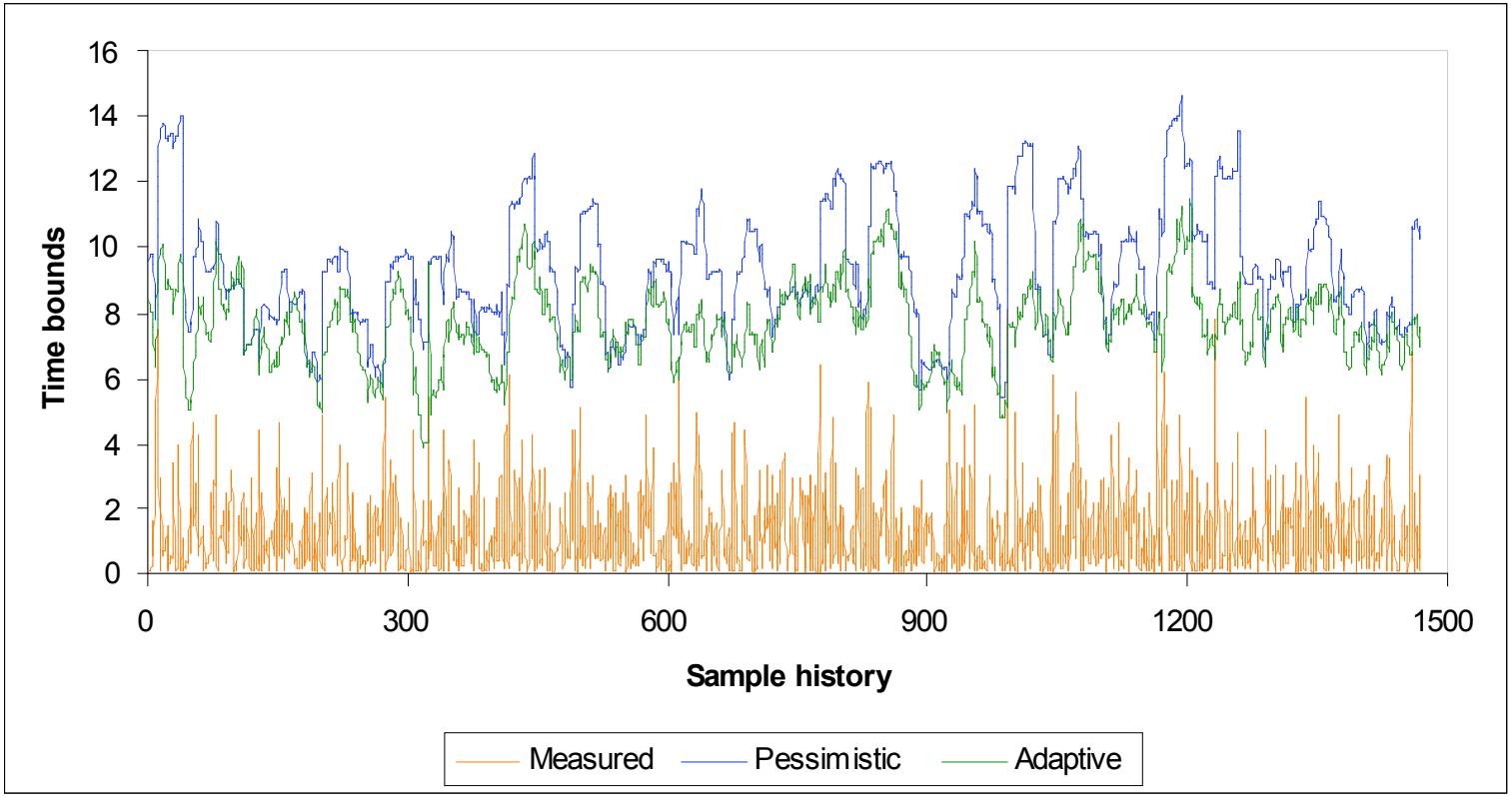
- Stable phase detected when $|\sqrt{V(\hat{D})} - E(\hat{D})| < \tau$, with τ being a tolerance bound
- From statistics, we know that for an exponential distribution, a 100(1-α)% confidence interval of E(D) is

$$\frac{2h}{\chi^2_{2h;\alpha/2}} E(\hat{D}) < E(D) < \frac{2h}{\chi^2_{2h;(1-\alpha)/2}} E(\hat{D})$$

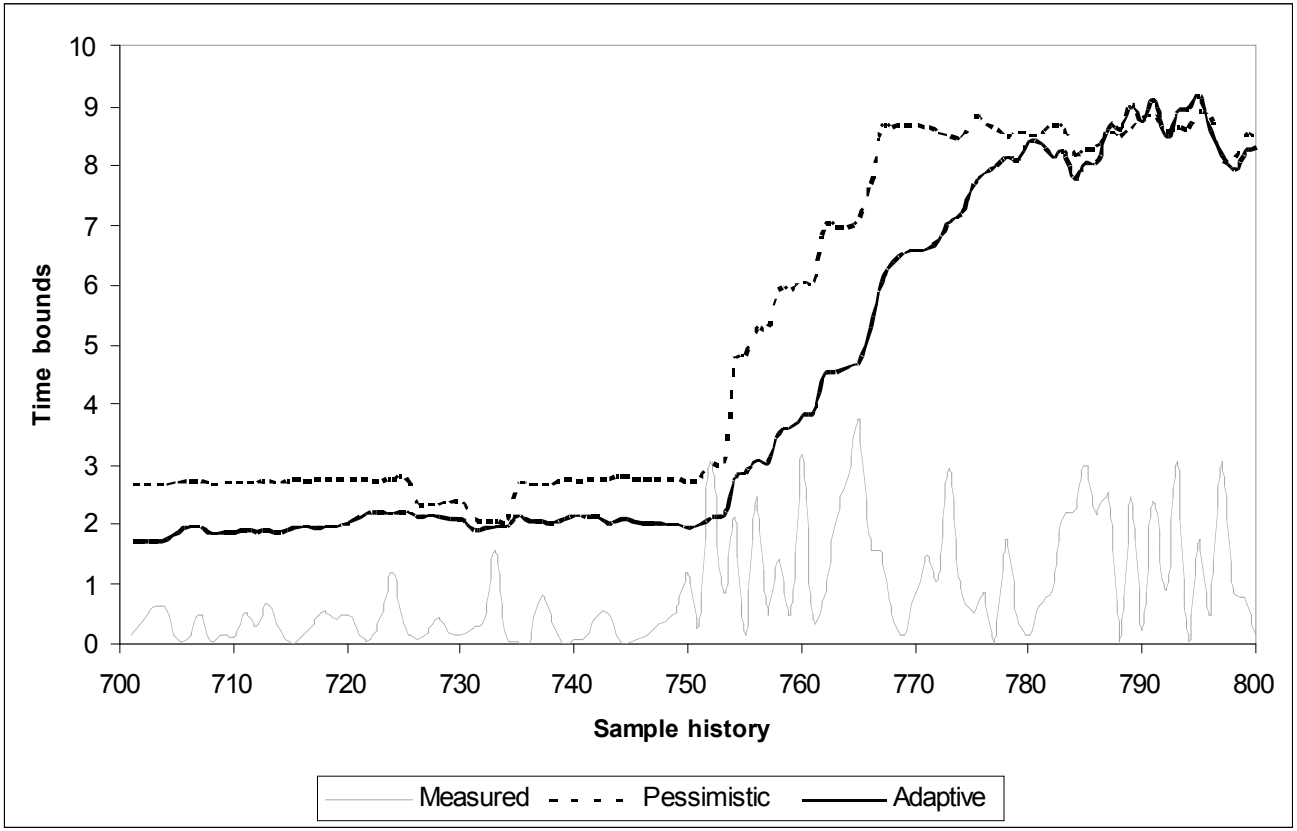
where h is the sample size

- If $\sqrt{V(\hat{D})}$ is inside the confidence interval, then stability is detected and the upper bound of the interval is used for $E(\hat{D})$

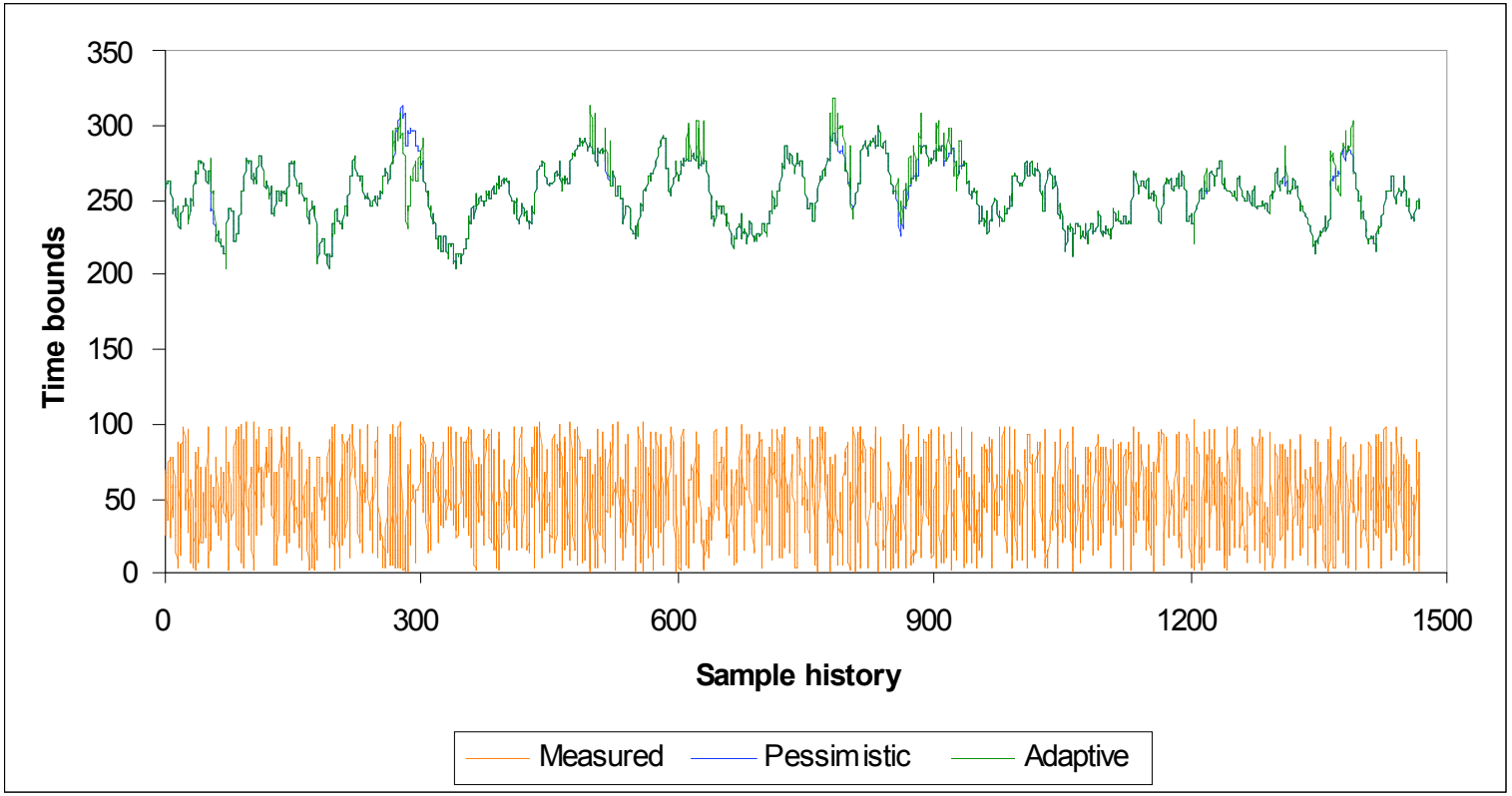
Pessimistic vs adaptive approach (Exponential)



Exponential variation



Pessimistic vs adaptive approach (Random)



Using real traces

- Trace set from Umass repository (http://traces.cs.umass.edu/index.php/Network/Network)
  - UPRM Wireless Traces
    - A collection of wireless traces from the University of Puerto Rico. Contains wireless signal strength measurements for Dell and Thinkpad laptops. Tests were performed over distances of 500 feet and one mile.
- Trace files:
  - We extract RTT values from tcpdump files using tcptrace utility.

Aggressive selection logic

- Kolmogorov-Smirnov test
- If more than one distribution is detected, considers the distribution which presents the lowest bound

Total points: 7307
Exponential observation: 0
Shifted Exponential observation: 10
Pareto observation: 343
Weibull observation: 5932
Transient observation: 1022

Timing faults adaptative: 371
Adaptative bounds average: 54.50328473389455

Timing faults conservative: 5
Conservative bounds average: 83.66266612760802

COVERAGE WAS NOT SATISFIED (98%), we should have at most 146 faults

Conservative selection logic

- Anderson-Darling test
- If more than one distribution is detected, considers the distribution which presents the lowest bound

Total points: 7307
Exponential observation: 0
Shifted Exponential observation: 4508
Pareto observation: 295
Weibull observation: 1108
Transient observation: 1396

Timing faults adaptative: 106
Adaptative average: 63.252117644247726

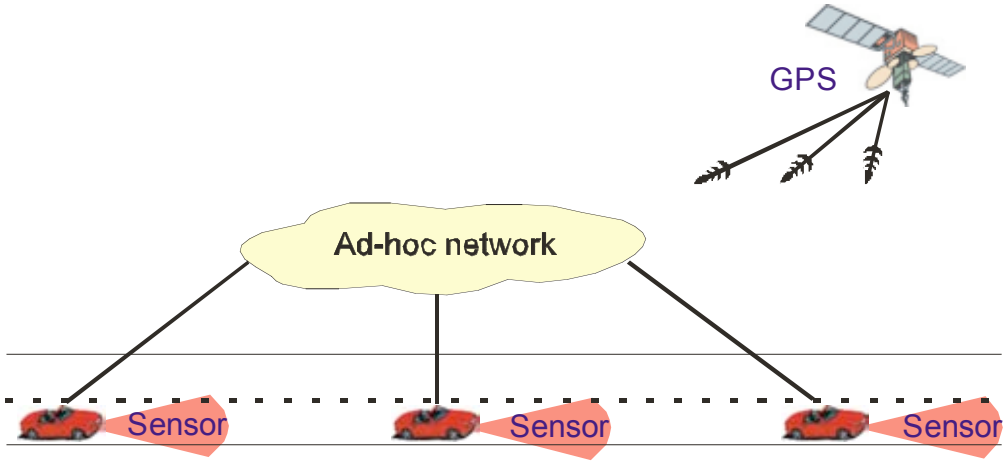
Timing faults conservative: 5
Conservative average: 83.66266612760802

COVERAGE WAS SATISFIED (98%), we should have at most 146 faults

Improving real-time control using a hybrid system architecture

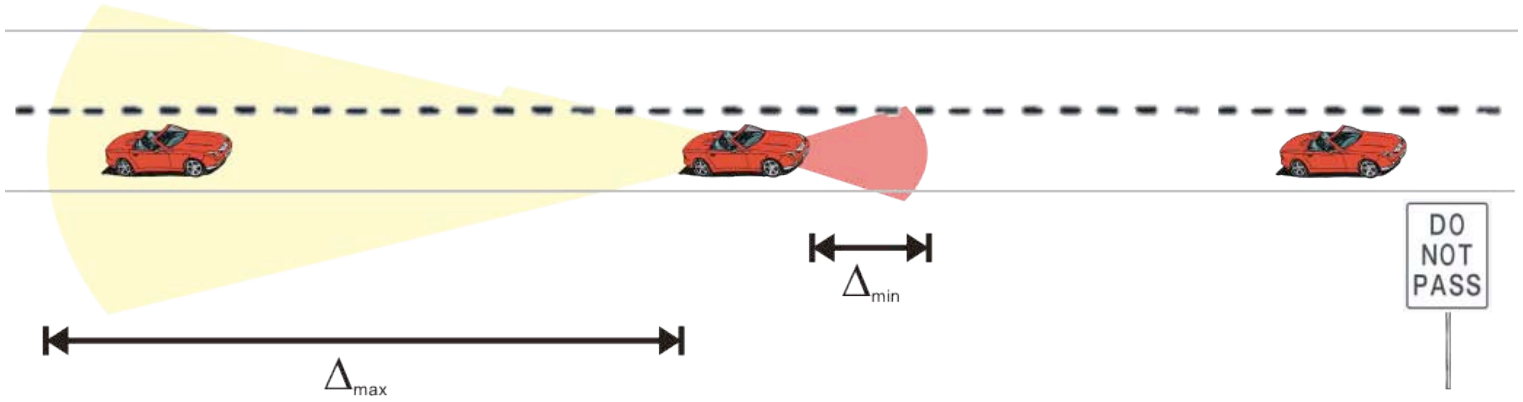
Platooning scenario

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

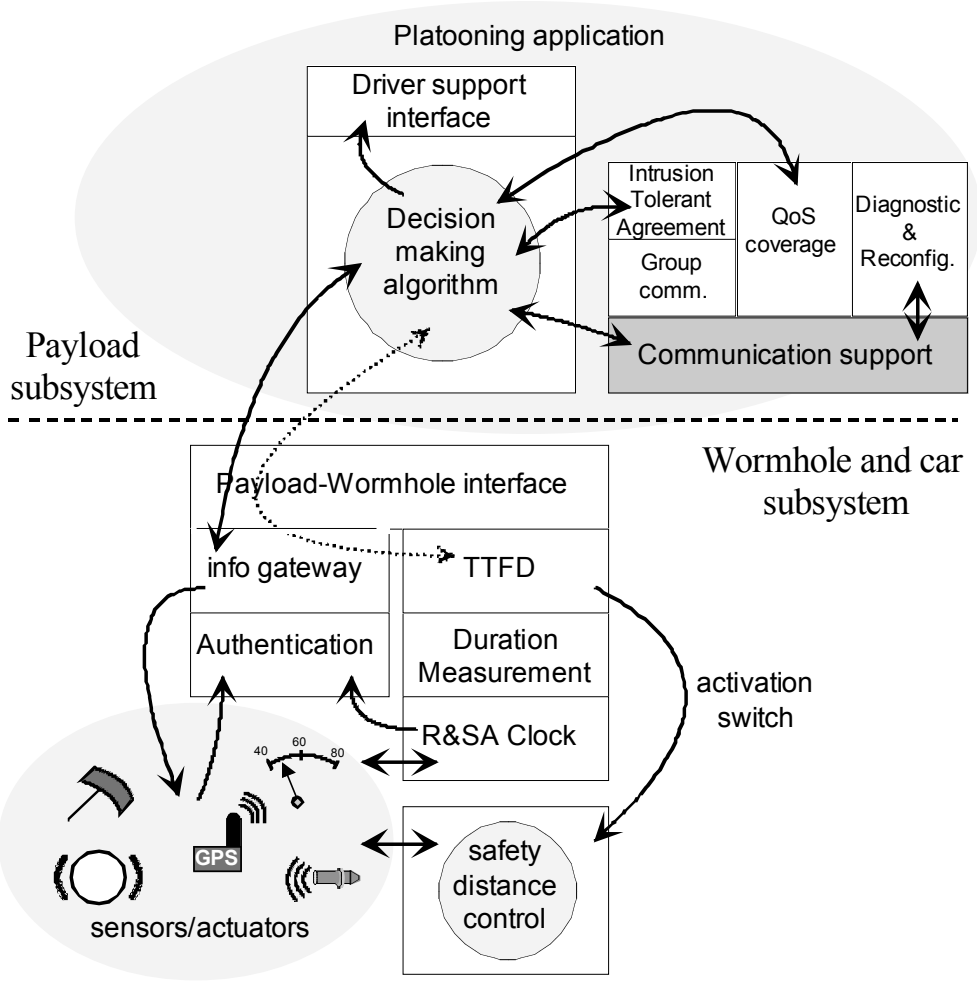


Requirements

- The fundamental requirements are:
  - Safety-critical: Δ_{min} to the front car
  - Non safety-critical: Δ_{max} to the car behind

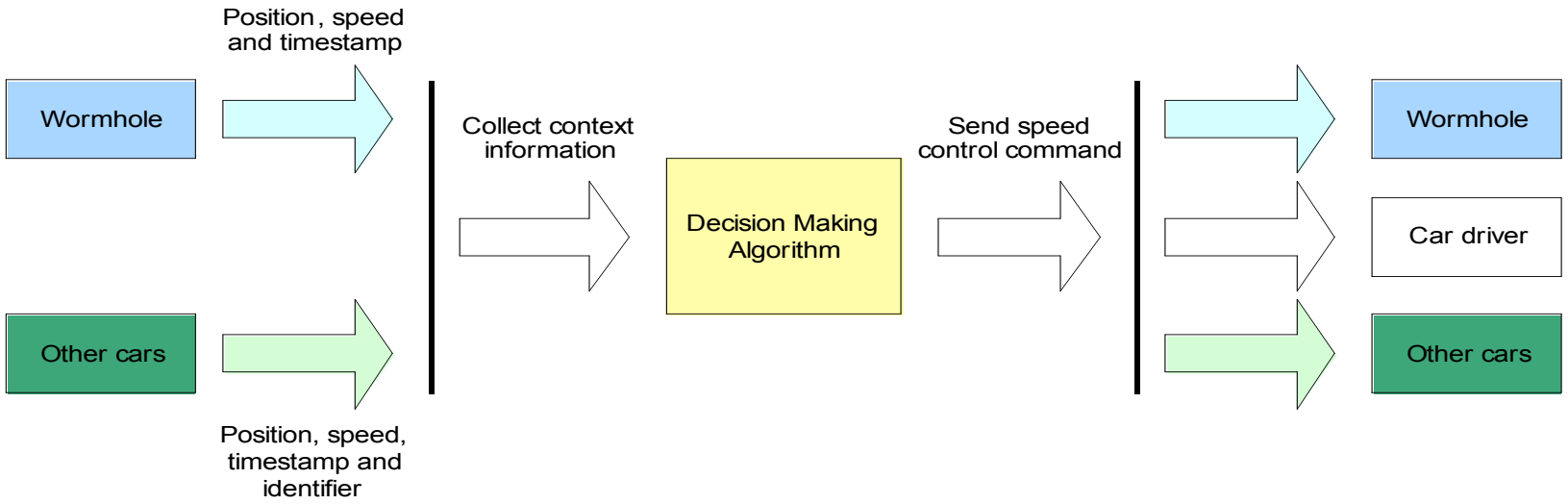


System architecture



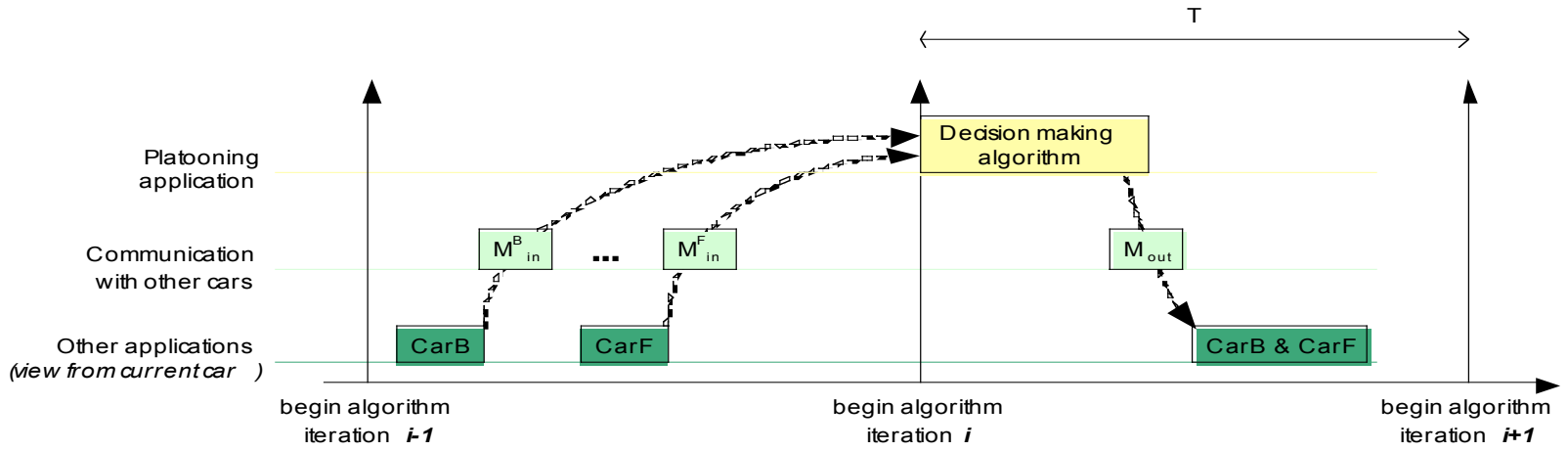
Platooning dataflow

- 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



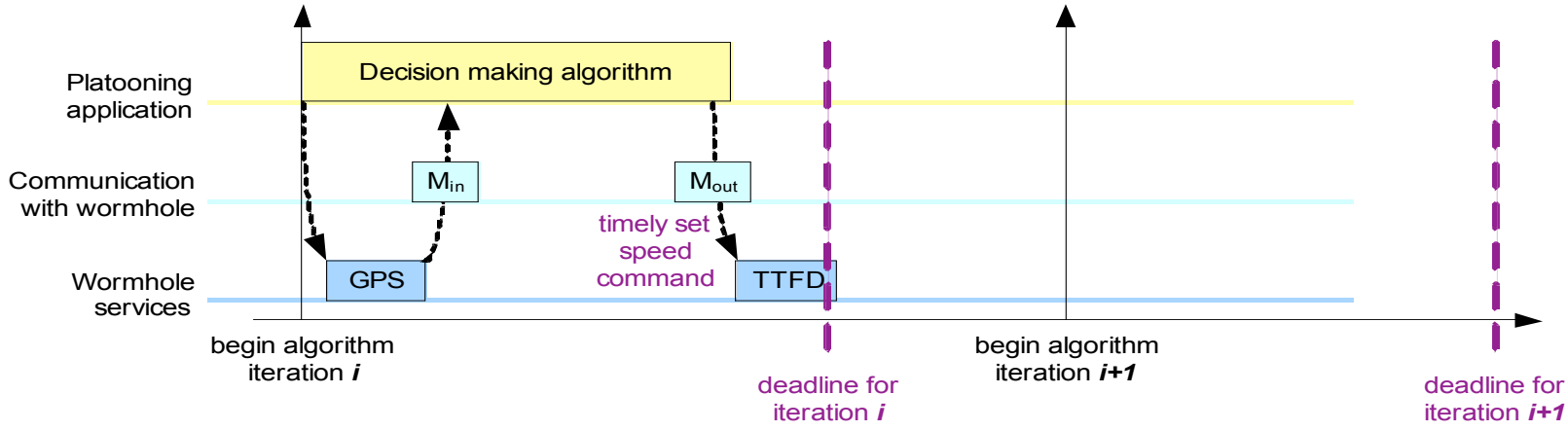
Decision algorithm

- Communication with other cars
  - task starts by collecting data, calculates a new speed for the car and sends its value to the neighbors (car B & car F)



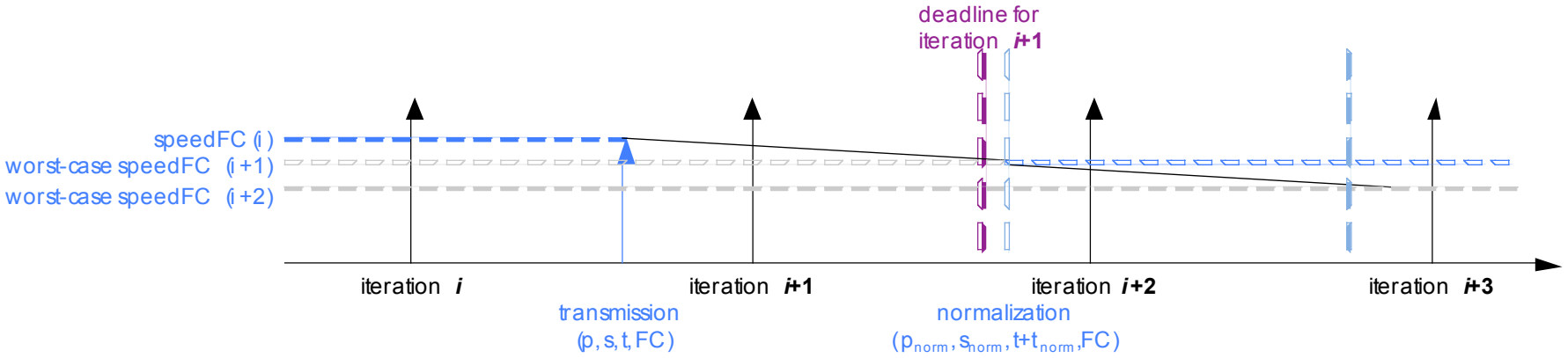
Decision algorithm

- Communication with the wormhole
  - task starts by collecting data from the wormhole, calculates a new speed for the car and sends the new speed value to the TTFD service



Decision algorithm

- Normalize all values to next iteration deadline
  - Use pessimistic assumptions about car behaviors (e.g. it stop right after sending its position)

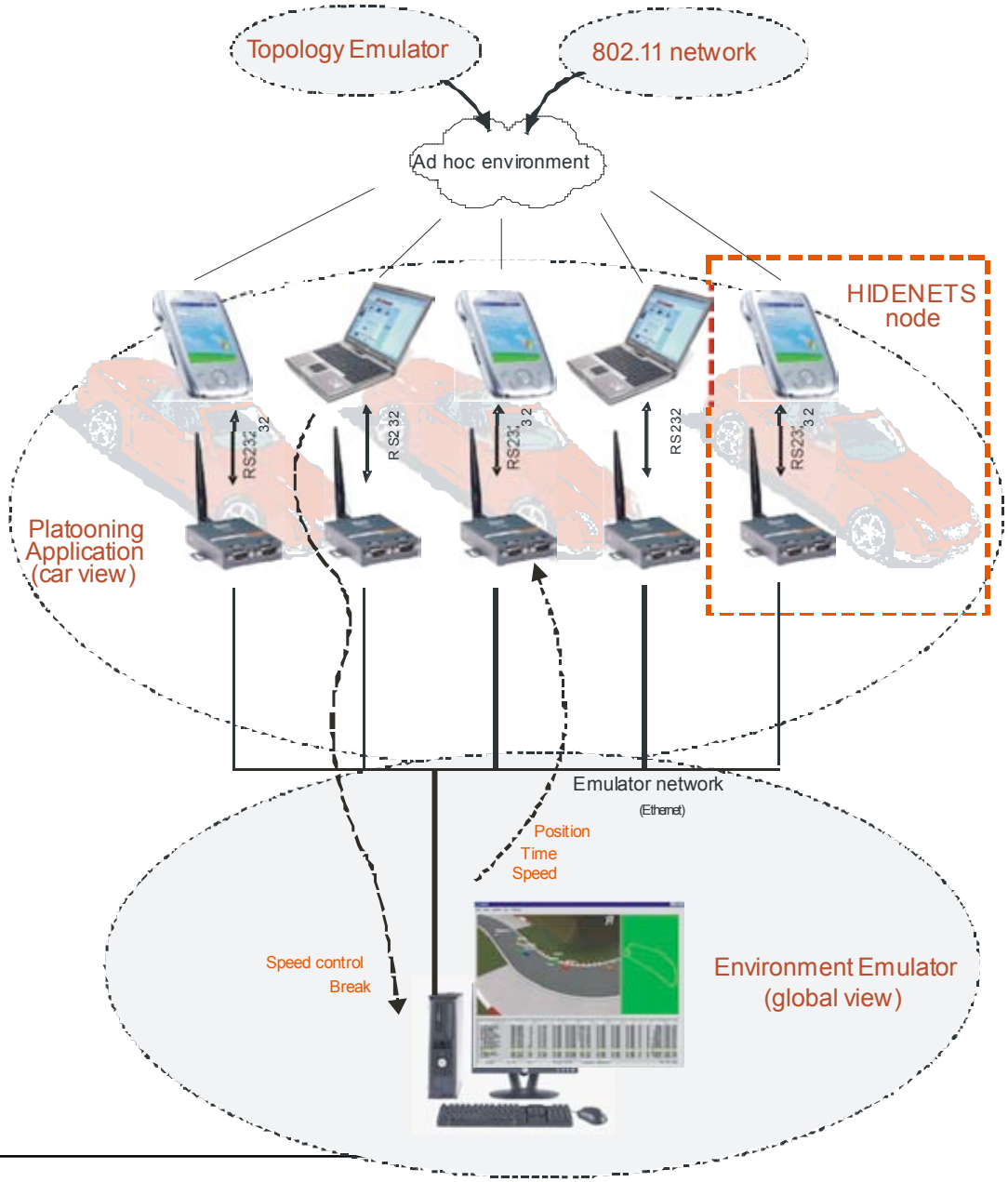


Timing Failure Handling

- **If the payload response is delayed**, the TTFD service detects the fault when the deadline is reached and a new speed value was not received.
- The TTFD service will immediately **activate the safety distance control componen**t of the car infrastructure…
- ... and, the **car will become controlled by a more pessimistic, but safe, control algorithm**.

- It is ensured that fail over will be done on time to secure safety (enough distance to break the car, if necessary)!

To conclude,
lets see some cars moving...


Topology Emulator
802.11 network
Ad hoc environment
HIDENETS
node
Platooning
Application
(car view)
RS232
RS232
RS232
RS232
Emulator network
(Ethernet)
Position
Time
Speed
Speed control
Break
Environment Emulator
(global view)

Thank you!

Visit us at
http://www.navigators.di.fc.ul.pt