Towards a cooperating autonomous car

Paulo Sousa
pjsousa@di.fc.ul.pt
FC/UL[*]

Paulo Veríssimo
pjv@di.fc.ul.pt
FC/UL

1 Introduction

The car of the future will be equipped with a large number of sensors, ranging from position and speed sensors to sensors indicating the presence of obstacles or indicating the road and weather conditions. All these sensors will be used by the car to perceive the reality and actuate according to it (e.g. braking, deviating from obstacles, horning). Furthermore, cars will communicate with other cars or entities in its proximity, over a wireless link, to achieve cooperation and coordination in certain occasions[1]. This paper describes the main challenges in engineering a cooperating autonomous car (we will refer to it as a *CoopACar*), presents possible approaches to the problem, and briefly describes a Cooperating Cars Simulator we are currently developing.

2 Challenges

The challenge is to achieve safe operation and proactivity in an open, unpredictable and non-deterministic environment. In terms of the construction of *CoopACars*, these challenges translate into two things:

- Correctly perceiving the reality
- Correctly actuating in response to a situation

[*]Faculdade de Ciências da Universidade de Lisboa. Bloco C5, Campo Grande, 1749-016 Lisboa, Portugal. Navigators Home Page: http://www.navigators.di.fc.ul.pt. This work was partially supported by the EC, through project IST-2000-26031 (CORTEX) and by the FCT, through the Large-Scale Informatic Systems Laboratory (LASIGE).

2.1 Perceiving the reality

From physics we know that it is impossible to a *CoopACar* (and to anybody) to perceive the **actual** reality. Without wanting to enter into a philosophical discussion, each of us has a different perception of the reality. So, the best approximation of reality that a *CoopACar* can have is one with a bounded error. However, to achieve a bounded error we must ensure timely communication. Unfortunately, it is very difficult to give timeliness guarantees in wireless networks.

To understand the relevance of ensuring that all cars perceive the same reality (with a bounded error), let us now discuss a concrete example. Consider *CoopACars* A and B approaching a cross with no traffic light. *CoopACar* A should stop because we assume *CoopACar* B has priority, but for some reason (e.g. some delay) *CoopACar* A doesn't timely perceive that *CoopACar* B is nearby. Since *CoopACar* A does not have a correct perception of the environment and since its decisions are based on that perception, a car crash occurs! Generically, if each *CoopACar* has different perceptions of the reality, car crashes can happen.

2.2 Actuating

Even if we assume that all *CoopACars* perceive exactly the same reality, can we say that car crashes will not occur and thus that safety will be achieved ? The answer is negative, because the car still has to react timely in accordance to each situation. For instance, consider the previous example, assuming that car A correctly and timely detects that car B is approaching. In this situation, car A knows that it has to stop and tries to signal the braking system, requesting its actuation. But if this internal communication suffers a delay, car A will stop too late and a car crash will still occur! Ensuring a timely actuation is therefore as important as ensuring a timely perception.

3 Towards a Constructive Approach

In the previous section we identified two fundamental challenges for a *CoopACar*:

- Timely[1] perception of the reality

[1]The perception has to be correct in both time and value domains, but in this paper we are only considering the time domain

- Timely actuation

Generically, a *CoopACar* has to perform timely actions in the presence of the uncertain timeliness characteristic of wireless networks.

A model that indeed addresses the challenges posed by uncertainty is the Timely Computing Base model (TCB)[4].

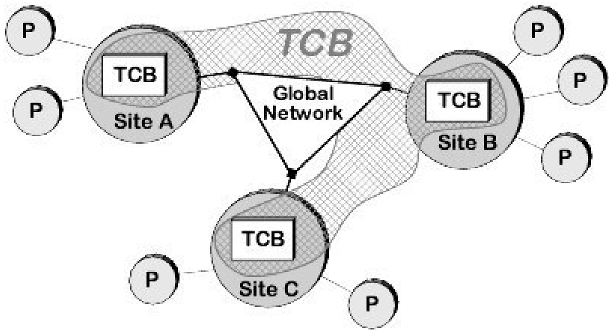

Figure 1. The TCB Architecture.

The TCB architecture is presented in figure 1. This model is one of partial synchrony. From an engineering point of view, it requires systems to be constructed with a small control part, a TCB module, to protect vital resources with respect to timeliness and to provide basic time related services to applications. The TCB provides simple support services, such as the ability to detect timing failures, to measure durations, and to execute *timely* timed actions. We will focus on the first one: timing failure detection.

The TCB timing failure detection service can be used by a *CoopACar* to timely detect and react to unexpected delays. For instance, it could be specified that a perception or an actuation was only valid within an certain time interval (to bound the error mentioned above). Then, if a timing failure occurs, the TCB would timely execute a specified fail-safety[5] handler (e.g. stopping the car).

The TCB can also be used to provide QoS Adaptation [2]. More specifically, the TCB can be used to allow *CoopACars* to adapt their speed according to the QoS allowed by the environment. This is very important, because if only the fail-safety mechanism described above was used, *CoopACars* would possibly stop more than necessary - it is better to move slowly without stopping than fast but constantly stopping. With QoS Adaptation they can gradually adapt their speed to environment conditions, and therefore reduce the probability of a timing failure to occur.

4 A Cooperating Cars Simulator

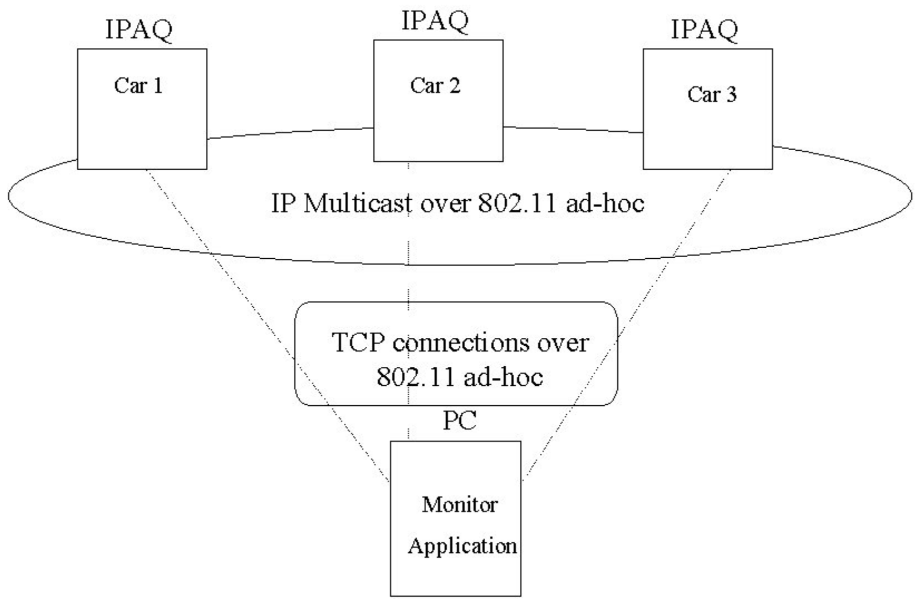
IPAQ
Car 1
IPAQ
Car 2
IPAQ
Car 3
IP Multicast over 802.11 ad-hoc
TCP connections over 802.11 ad-hoc
PC
Monitor Application

Figure 2. Simulator Architecture.

We are currently developing a *Cooperating Cars Simulator*. The simulator architecture is depicted in figure 2. Each *CoopACar* is implemented over a Pocket PC 2002 IPAQ. *CoopACars* communication is event-based using IP Multicast over 802.11 ad-hoc. In the current version, the position and movement of each car is simulated, although it could be possible to easily integrate GPS receivers with the IPAQs in order to allow each car to know its exact position. Beyond the cars, we have a monitor application running in a PC that is used to show the actual reality. This monitor application allows us to observe *CoopACars* behavior (e.g. if they collide). But the construction of this ”reality observer” raises some problems.

As we have mentioned before, reality is an instantaneous perception of the environment, and thus, strictly speaking, it cannot be reproduced in a monitor. The best we can do is to enforce better communication between the monitor application and *CoopACars*, guaranteeing much faster communication than environment dynamics, to give the impression that we are observing the actual reality. Again, the TCB can be used to help addressing this problem, but now using it as an ”accelerator”. How can we do this ?

As described in [4], a system with a TCB is divided into two well-defined parts: a payload and a control part. The generic or payload part prefigures what is normally ’the system’ in homogeneous architectures. It exists over a global network or payload channel and is where applications run and communicate (see figure 1). The control part is made of local TCB modules, interconnected by some form of medium, the control channel. This control channel has much stronger guarantees and can be envisaged as a *wormhole* [3]. We intend to use it as an ”accelerator” for the communication between the monitor application and *CoopACars*. One could think that this control channel could also be used for car-to-car communication, but this is not a feasible approach. In real life, the control channel will have limited resources and will not sustain heavy end-to-end communication. In the context of the simulator, we can manipulate car-to-monitor communication so that the control channel be efficiently used.

5 Concluding Remarks

This paper described some important challenges that arise when we try to engineer a cooperating autonomous car. A *CoopACar* will have to timely perceive the reality and actuate upon it. For that, a TCB can be used.

We also presented an ongoing work on building a cooperating cars simulator. This simulator will assist in demonstrating if the described challenges can be solved by the proposed approaches.

References

[1] V. Cahill (editor). CORTEX: Definition of application scenarios, Nov. 2001. Deliverable D1, EC project IST-2000-26031, http://cortex.di.fc.ul.pt/Deliverables/WP1-D1.pdf.

[2] A. Casimiro and P. Veríssimo. Using the Timely Computing Base for dependable qos adaptation. In *Proceedings of the 20th IEEE Symposium on Reliable Distributed Systems*, pages 208–217, New Orleans, USA, Oct. 2001. IEEE Computer Society Press.

[3] P. Veríssimo. Traveling through wormholes: Meeting the grand challenge of distributed systems. In *Proc. Int. Workshop on Future Directions in Distributed Computing (FuDiCo)*, pages 144–151, Bertinoro (Italy), June 2002.

[4] P. Veríssimo and A. Casimiro. The Timely Computing Base model and architecture. *Transaction on Computers - Special Section on Asynchronous Real-Time Systems*, pages 916–930, Aug. 2002.

[5] P. Veríssimo, A. Casimiro, and C. Fetzer. The Timely Computing Base: Timely actions in the presence of uncertain timeliness. In *Proceedings of the International Conference on Dependable Systems and Networks*, pages 533–542, New York City, USA, June 2000. IEEE Computer Society Press.