A plateform for road experiments

B. Ducourthial and S. Khalfallah
Laboratoire Heudiasyc UMR CNRS 6599
Université de Technologie de Compiègne
Compiègne, France 60205
Email: firstname.name@hds.utc.fr

***Abstract*—The Intelligent Transportation Systems (ITS) currently attract a lot of attention. It is expected that such systems could improve the road safety, offer a better resource usage, increase the productivity, reduce the impact of transport on the environment... Besides the standardization effort regarding protocols and architectures, experimentation of new solutions is important, especially in the so-called VANET (Vehicular Ad hoc NETworks). Indeed, many issues are still open in these networks, and many research teams are involved over the world. The road experimentations allow to validate new ideas and to guide new theoretical development. They offer a better understanding of the VANET.**

This paper deals with experimentations on the road. A simple architecture is sketched; it is composed of on-the-shelves hardware and a software suit. Experimentations are reported and examples of results are given. This work is intended to help in the design of new road experimentations.

I. INTRODUCTION

A. Motivation

These last years, Intelligent Transport System gain attention of the research community. It is expected that ITS will reduce the road fatalities (around 40,000 deaths per year in USA or Europe), increase the productivity and the profitability of the infrastructures, avoid traffic jams and reduce the impact of road transports on the environment. ITS involve the road infrastructure, the car manufacturers, the telecommunication infrastructures... They often rely on embedded systems and wireless networking facilities.

Regarding the on-board applications that could be available in future vehicles, four families can be sketched. The first family is composed of the *infrastructures oriented applications*. These applications rely on the embedded sensors in the vehicles and vehicle-to-infrastructure (V2I) communications in order to collect data from the vehicles. The information will be used by the infrastructure-side for optimizing the management of freeways, the goods' transportation, the emergencies' organization.... The second family concerns the *vehicles oriented applications*. They rely on embedded sensors and infrastructure forecast to give accurate information to the vehicles by means of infrastructure-to-vehicles (I2V) communications and vehicles-to-vehicles (V2V) communications. In this family, we find road safety applications such as vehicle behavior adaption, collision's prevention, inter-vehicles security distance management... The received information is processed by the embedded computers, and may affect the vehicle behavior. The applications are critic. The third family concerns the *drivers oriented applications* to ease the road usage. We may find applications regarding traffic jam avoidance, road work warning, ride duration estimation *etc.* Here, this is the driver that analyze the information. The information does not concern critic application but it is expected to help the driver. Finally, the last family concerns infotainment applications for passengers. These applications may offer new on board services such as Internet access, distributed games, chats, tourist information, city leisure information, movies announces downloads [1].

Research projects regarding ITS can be found in the USA (VII, CICAS, IVBSS...), in Europe (CVIS, SAFESPOT, COOPERS, PReVENT, GST, HIGHWAY, FLEETNET...), in Japan (SmartWay, VICS...), in India (ITSIndia), in Germany (NOW), in France (PREDIT) *etc.* Standardization of the vehicular communication is now ongoing in major international organizations (IEEE, IETF, ETSI, ISO, SAE, ASTM), industrial consortia such as the Open Mobile Alliance (OMA) and the Car-to-Car Communication Consortium (C2C-CC) and national ITS authorities.

ITS is extensively studied by both theoretical and experimental researchers. For example, the study of Inter-vehicle communication (IVC) networks in [2] exhibit characteristics that are dramatically different from many generic MANETs. The authors elicits these differences through simulations and mathematical models. It is our opinion that experimentation and prototyping is important for guiding theoretical studies, and validating technical solutions. In this paper, we describe a testbed dedicated to the study of vehicular applications.

B. Related work

Some architectures have been studied by consortia. The architecture of the GST project (which concerns the communications vehicles-infrastructure) is based on Linux, OSGi, IPv6, HTTP, SOAP and the protocol OMA DM [3]. The architecture of the CVIS project [4] is based on Linux, OSGi, IPv6 and CALM. To the best of our knowledge, no performance measures on the road relying on these architectures has been published for the moment.

Various experiments concerned the communications in VANETs. They use mainly the standard IEEE 802.11. In [5], the authors studied the feasibility of using the IEEE 802.11b standard for connecting a moving vehicle to an access point. In [6], the authors studied the behavior of the network connections (TCP and UDP) between a moving vehicle and an access point. The purpose was to understand the impact of the speed, the rate of transmission and the packet size. In these experiments, IEEE 802.11b devices have been used. The lack of accuracy of the embedded architecture appeared to be limitative.

In [7], the authors present some experimental results using a multimedia application in a VANET. Two vehicles have been used in two different environments (urban and freeway). The authors concluded that: i) the signal-to-noise ratio (SNR for *Signal to Noise Ratio*) is more important on freeway than in an urban zone, ii) the connection is more reliable on freeway than in a city and iii) it is better to use large data packets on freeway and small packets in urban zone (more fragmentation). The embedded architecture was based on PC under laptops, IEEE 802.11b PCMCIA cards and UDP. Similar results have been obtained in [8] for the SNR and the noise. Furthermore, the authors analyze the RTT, the TCP and UDP throughput. Three vehicles have been used with a static routing. According to the authors, the deployment of multimedia applications is difficult in a multi-hop network of vehicles. The embedded architecture was based on PC under Windows XP, UDP and TCP protocols, iperf [9] for the generation of the traffic, a GPS receiver, Netstumbler for the measure of the signals and a static routing on three vehicles.

In [10], the authors measured the link quality on freeway in urban and semi-urban environments. The results of the study showed that the semi-urban environment is the most favorable for the inter-vehicles communication. The architecture used in this study was based on Linux, a GPS receiver, the UDP protocol UDP and Netperf, a tool used for network performances evaluation. The purpose of the experiences in [5], [6] is to understand the performances in terms of connection duration and loss rate when a mobile car connects in points of access. The embedded architecture used in [5] relies on PC under Linux, TCP and UDP transport protocols, HTTP, iperf for the TCP traffic generation, wget and Apache for the HTTP traffic generation and finally tcpdump and kismet for the packet capture.

In [11] the authors perform evaluations carried out using the IEEE 802.11a protocol at 5.2 GHz between a moving vehicle and a fixed base station. They focus on realistic urban environment with speeds smaller than 50 km/h. According to the authors, the performances at very low speeds is degraded due to the presence of null zones on urban environment. Null zones are regions of poor reception due to a pronounced interferences. An 802.11b access point has been used; vehicles were equipped with 802.11 PCMCIA cards and omnidirectional antenna offering a gain of 7 dBi. Measurements have been done with iperf.

These experiments rely generally on a static routing and specific applications. The design of a simple architecture suitable to VANETs experiments and offering divers dedicated protocols for routing, diffusing or self-organization would help in the design of new road experimentations.

C. *Contribution*

In this paper, we describe an embedded architecture allowing a large set of road experiments. It allows to rapidly prototype protocols and distributed applications, including cross-layer protocols.

We first describe the hardware we tested. Then we summarize the software architecture. It is composed of a core program called *Airplug* that manages all the communications intra- and inter-vehicles. Every development is then easily made in user space processes, by using any programming language. The inter-applications communications are based on a simple text-based scheme, with an addressing adapted to dynamic networks.

To illustrate the interest of such an architecture, we explain how it can be used to study the performance of a given multi-hop protocol.

II. HARDWARE CONSIDERATIONS

For the purpose of our road experimentations, we used two equipped vehicles from our lab with (among other equipments) stable power supplying (Figure 1). However we also performed experiences using standard vehicles in order to perform experiences up to five vehicles. For this purpose, we tested several hardware solutions we summarize here.

Laptops are adapted for experimenting in vehicles because they incorporate batteries. However laptops generally require a PCMCIA WiFi card in order to plug an external antenna on the roof of the vehicles. The connectors (MC) are too fragile and do not resist to multiple tests. Moreover such a platform is not robust and we encountered problems during intensive tests, especially with high temperature (eg PCMCIA card frozen). An alternative consists in using industrial PC (shoe-box) and WiFI PCI cards (Figure 2). This platform is much more robust (and not more expensive). It can be directly powered by 12V available in vehicles. However this requires a stabilize 12V output, or an additive battery to avoid any unwanted reboot.

Antennas should also be chosen with attention. A short communication range leads to small inter-vehicles distances. This is only feasible at low speed and in low traffic roads. On the other hand, if the antennas' range is too large, experimentations relying on multi-hops communications (such as alert diffusion in convoys) are difficult to coordinate because the inter-vehicles should be large to obtain useful tests. In fact, the range is never stable along the experimentation. Without obstacles on straightforward roads, the portability becomes very large, while it will be reduced in presence of trucks, trees, or building.

Regarding the operating system, it is important to reduce as many as possible the kernel and the running services. For instance, it is important that no network services (ntpdate, cups, dns...) disturb the measures. We used Debian GNU Linux.

To prevent any radio disturbance with the wireless device, an USB GPS receiver has been chosen (instead of a Bluetooth-based one). We choose a simple device (Holux) compatible with the Holux Linux kernel module. It is important to notice that, without any post-computing, all the positions given by such a device are not useful. While our experiments did not require embedded digital map, such application would certainly be useful.

As summary, the platform Caremba is established by some kits composed of industrial PC (shoe-box) 12V, a USB GPS receiver, a IEEE 802.11 wireless PCI interface card and a ominidirectional WiFi antenna for the outdoor communications 7dB-i. The operating system is Debian 4.0. (*cf.* figure 2).

Finally, it is important to notice that road experimentations require many persons and an important preparation. Talky walky and cellular phones are required to coordinate the participants. Tests are long and not always reproductible (traffic road varies), which is a drawback for computing average. Thus, a platform that allows to prototype and test the developments in lab, and to replay the road experimentations in lab is required. Moreover, road experiments involve generally too few vehicles and on the road experiments should be completed by simulation to study the scalability of the protocols. So, a platform which allows to switch from simulation to road easily is very useful. The platform we describe in this paper will soon own these capabilities (currently in tests) but their description is out of the scope of this paper, which focus on road experiments.

Fig. 1. Experimental vehicles.

Fig. 2. Embedded equipment.

III. The Airplug Architecture

A. Processes-based architecture

Embedded computers run Linux and the Airplug suit [12]. This software suit has been developed for experimenting in dynamic ad hoc networks. It is composed of a core program and a set of applications. It allows to easily experiment and prototype either on the road or on the lab.

The core program `airplug` manages the inter-applications communications, either local-to-the-host or inter-vehicles. The applications are plugged on top of the core program forked processes connected by pipes). The applications reach the network through `airplug`. All these processes run in user-space (Figure3). This processes-based architecture leads to a robust system where each application run separately from the others, and where only `airplug` is in charge of the network. The system is not affected by an application that may fail (both the network and the other applications are protected by `airplug`. Moreover, managing all the remote communications by a single user-space program allows many optimizations (*piggybacking*, scheduling, QoS policies *etc.*).

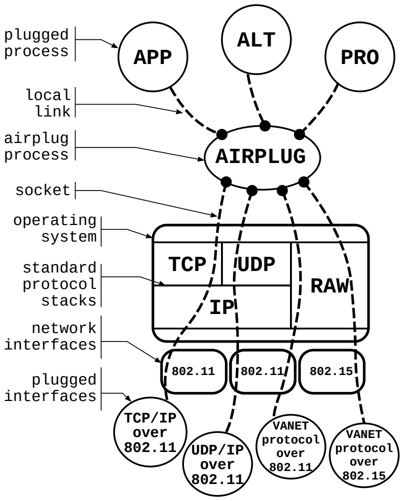

Fig. 3. Airplug architecture.

B. Airplug IPC

The Inter-Process-Communication relies on text-based ASCII messages and very few rules. This simple message-based framework uses an addressing well adapted to dynamic networks such as VANETs. An application `APP` can send a message locally (keyword `LCH` for localhost), to the nodes in the neighborhood (keyword `AIR`), to both (keyword `ALL`) or to a given host using a name or any kind of address. The destination is either a specific application (known by a three-characters mnemonic such as `APP`) or all the applications (keyword `ALL`). In this case, only applications which have subscribed to the sender `APP` will receive the message. Such subscribing are managed by `airplug`; applications use the action keyword `BEG` to indicate to `airplug` that they want to receive the flow of messages of a given application, and the `END` keyword to end such a subscribing (Figure 4).

When a local application `APP` wishes to send a message to the neighbor nodes, `airplug` will broadcast a sixth field message composed of the sender application `APP`, its identity (if any), the keyword `ALL` or the name/address of the receiver and the receiver application of the keyword `ALL`. All messages have a *control* field used for piggybacking (eg. GPS data...) and a *payload* field used by the applications (Figure 5).

action	application	host	control	payload
SND		LCH		
BEG		AIR		
END		ALL		
		hostname		

Fig. 4. Format of a message for a local communication through a *local link*.

appl.	host	appl.	host	control	payload
sending	sending		AIR		
appl.	hostname		ALL		
	(if any)		hostname		

Fig. 5. Format of a message for a remote communication between two hosts.

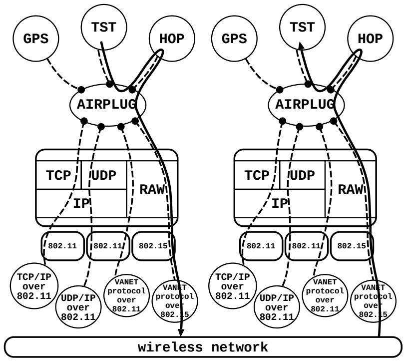

Fig. 6. Developing a new protocol (here HOP) with Airplug.

This architecture imposes no paradigm nor specific programming languages. The applications can be developed in any language, providing that they can send to `stdout` and receive to `stdin`. Indeed, when launching the applications, `airplug` redirect the pipes to the standard IO. Some light libraries have been developed to format the messages and process the fields at the reception. However this is very basic ASCII string processing.

C. Implementing new protocols

Remote communications rely on sockets. They generally consist in broadcast in the surrounding of the host. The sockets are based either on standard stack in the kernel (TCP/IP, UDP/IP...) or on raw sockets. This kind of socket is used to bypass the standard stack in the aim of experimenting new protocols. Moreover several different protocols can cohabit (*e.g.*, geographical routing for the alerts, source-oriented routing for the communications V2I).

As an example, the experiments reported in this article used the conditional transmissions, an efficient communication scheme for VANET that relies on conditions instead of addresses [13]. This protocol is implemented in a plugged Airplug application called `HOP` (Figure 6). The applications (here `TST`) that want to use this protocol instead of the standard stack send a local request to the local `HOP` instance. The message is then routed by the HOP protocol from nodes to nodes (that is, from remote `HOP` instances to remote `HOP` instances. When the message reaches the destination, the `HOP` instance delivers it to the local application, through Airplug and a local communication.

IV. APPLICATION: PERFORMANCE OF A MULTIHOP PROTOCOL

In order to illustrate the interest of the experimental platform, we summarize some tests on the road dedicated to the performance study of conditional transmissions, an efficient routing scheme in VANET [13].

A. Experimentation overview

The aim of these experimentations is to evaluate the performances of conditional transmissions, implemented through the `HOP` program (see Section III).

The conditional transmissions substitute conditions to addresses. By dynamically evaluating the conditions at the message reception, the protocol better fits to the dynamic than those relying on addresses. Indeed, using addresses in a dynamic network is a big challenge because many updates are necessary Two kind of conditions are used. The forward condition (CFW) determines whether a message should be broadcasted by an intermediate node or not. The upward condition (CUP) determines whether a message should be given to an application layer or not. Conditions are fixed by the sending applications, and are included in the messages. Examples of conditions are: time or delay, geographical position, distance, trajectory matching (eg. being behind the sender). For more details on the conditional transmission, refer to [13].

These experiments involved up to five vehicles. The first vehicle sends messages. The CFW conditions has been set in order that messages progress from the first to the last vehicle. In order to really experiment several hops while the transmission range varied (see Section II).

In order to generate a traffic and to measure the end-to-end delay of the first packet, the average end-to-end delay, the loss rate as well as the loss intervals, we developed the TST program. In order to obtain accurate measures, we combined the GPS dating (common to all the vehicles but with 1 Hz frequency) with the hardware clock of the embedded PC (accurate but not synchronized). The delay between the reception of the last NMEA frame sent by the GPS and the sending of the TST message is included in the message itself with the GPS clock. At the reception, the delay between the arrival of the last NMEA frame and the reception is also stored. A simple computation gives then a precision around 1 ms, which is sufficient to measure the transmission delay.

B. Results sample

A sample of results is given in Figure 7. By forcing the relay in each vehicle of the convoy, we obtained results depending on the number of hops. The delay is then proportional to the number of hops. Our implementation of the conditional transmissions requires about 246 ms per hop to forward a message. This delay has been measured at the applicative level. It includes the delay required to the MAC layer to obtain access to the wireless network.

# hop	**1**	**2**	**3**	**4**
packets lost (%)	9	32	41	58
delay 1^{st} packet (ms)	95	132	260	386
average delay (ms)	154	341	518	914

Fig. 7. Sample of results.

Since the experiences were performed on the road, the environment varied a lot during the tests (traffic and surrounding variation). Figure 8 displays the variation of the packets' loss along the experimentation for vehicles 2 and 5. We can see that even a stable convoy of vehicles does not lead to a stable data network.

This observation should be taken into account when designing routing relying on stable convoys, such as cluster-based routing. It also questions the studies by simulation of road traffic and VANET. Simulators are generally far from modelling a real environment, which varies a lot along the experimentation.

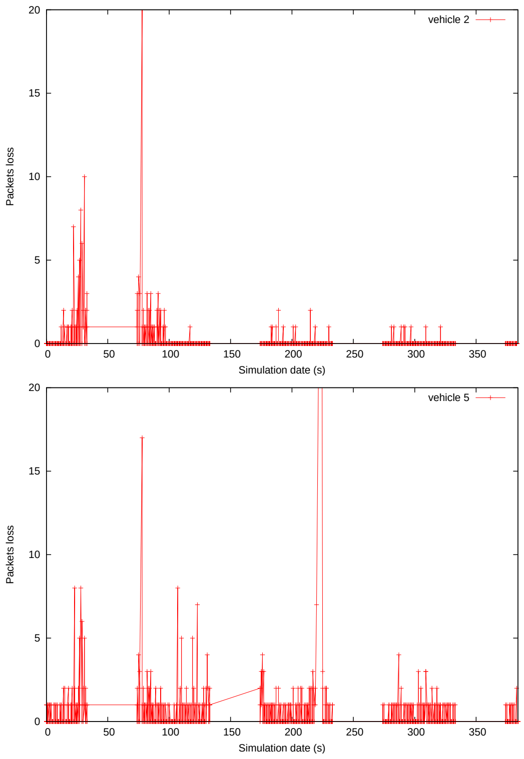

Fig. 8. Packets loss in the vehicle 2 (top) and 5 (bottom).

V. CONCLUSION AND PERSPECTIVE

This paper deals with road experiments, which are of great important for designing VANET applications. An experimental platform has been presented. It relies on common hardware and the Airplug software suit. This testbed allows easy prototyping of inter-vehicles applications and cross-layering protocols. To illustrate the interest of our platform, we presented the on-the-road performance study of a routing protocol called *conditional transmissions* [13]. Besides the performances evaluation, it is interesting to note the variation due to the environment along the experimentations. Future road experiments have been planed to complete this study.

A large set of applications have been developed for Airplug by different contributors. The Airplug software suit is available for research teams (contact the authors). We believe that it can help in studying VANET and designing efficient protocols.

Finally, all the developments (applications and protocols implemented in plugged processes) can be used without any change in the lab by replaying the GPS positions logged in the road and by filtering the out-of-range messages (currently in test). The developments can also be used in emulation mode without any change: the wireless communications are emulated in a computer. Finally, since any programming language is allowed by the platform, the developments can be reused in Network Simulator (with very few changes), providing they have been written in Tcl/Tk. We developed a Tcl/Tk library to ease such developments. These facilities help in studying the scalability of the protocols. Moreover, simulations can take benefit from road measures. These functionalities will be available in the next release.

ACKNOWLEDGEMENT

The authors wishes to thank Yacine Khaled for his contribution in the conception and the development of the platforms. We would also thank participants to our road experiments from France Telecom R&D and the Heudiasyc Lab (UTC).

REFERENCES

[1] K. Lee, S.-H. Lee, R. Cheung, U. Lee, and M. Gerla, "First experience with cartorrent in a real vehicular ad hoc network testbed," in *VANET MOVE'07*, ACM, Ed., Anchorage, Alaska, May 2007.
[2] A. E. J. Blum and L. Hoffman, "Challenges of intervehicle ad hoc networks," *IEEE Transaction on Intelligent Transportation Systems,*, vol. 5, pp. 347–351, 2004.
[3] "The GST project," http://www.gstforum.org/.
[4] "The CVIS project," http://www.cvisproject.org/.
[5] R. Gass, J. Scott, and C. Diot, "Measurements of in-motion 802.11 networking," In Proc. WMCSA, Apr. 2006.
[6] J. Ott and D. Kutscher, "Drive-thru internet: Ieee 802.11b for automobile users," In IEEE INFOCOM, 2004.
[7] P. Bucciol, E. Masala, N. Kawaguchi, K. Takeda, and J. De Martin, "Performance evaluation of H.264 video streaming over inter-vehicular 802.11 ad hoc networks," Proc. of 16th Annual IEEE International Symposium on Personal Indoor and Mobile Radio Communications (PIMRC), Berlin, Germany, Sep. 2005.
[8] F. Hui, "Experimental charaterization of communications in vehicular ad hoc networks," Thesis Master of Science in Computer Science, University of California Davis, 2001.
[9] "Iperf traffic generator." http://dast.nlanr.net/Projects/Iperf/.
[10] J. Singh, N. Bambos, B. Srinivasan, and C. D., "Wireless lan performance under varied stress conditions in vehicular traffic scenarios." *In Proceedings. IEEE 56th Vehicular Technology Conference*, vol. 2, pp. 743–747, 2002.
[11] N. Cottingham, J. Wassel, and K. Harle, "Performance of IEEE 802.11a in vehicular contexts," 2007.
[12] B. Ducourthial, "About efficiency in wireless communication frameworks on vehicular networks," Invited paper, workshop ACM WIN-ITS co-located with IEEE ACM QShine 2007, Canada, August 2007.
[13] B. Ducourthial, Y. Khaled, and M. Shawky, "Conditional transmissions: a communication strategy for highly dynamic vehicular ad hoc networks," *IEEE Transactions on Vehicular Technology, special issue on vehicular communication networks,*, vol. 56, no. 6, pp. 3348–3357, November 2007.