Networking Needs and Solutions for Road Vehicles at Imara

Olivier Mehani, Rodrigo Benenson, Séverin Lemaignan, Thierry Ernst

Networking Needs and Solutions for Road Vehicles at Imara

Olivier Mehani, Rodrigo Benenson, Séverin Lemaignan and Thierry Ernst
Inria Rocquencourt, Imara team
Domaine de Voluceau, BP 105
78153 Le Chesnay Cedex, France
Email: {firstname.lastname}@inria.fr

Abstract: Trying to provide safer and more efficient transportation solutions, the need for reliable and efficient communications is increasing in ITS (Intelligent Transportation Systems) use cases such as collaborative road data collection, driver information systems, infotainment or fully driverless vehicles. Applications that rely on vehicle-infrastructure (V2I) and vehicle-vehicle (V2V) communications have different requirements and would benefit from a widely deployed common standard for the exchange of data. IP (Internet Protocol) is thus advocated as the convergence layer between the various available wireless technologies available on the market. This paper surveys the methods employed in the Imara team to provide a stable communication environment allowing driverless vehicles to interact with each other in several scenarios where information sharing is required. An account of how the solutions are integrated and how well they behave together in live experiments is given.
Areas of interest: In-Vehicle, Inter-Vehicle and Infrastructure to Vehicle Communications; ITS Architecture, Interoperability and Standards; Mobile IP and Network Mobility in IPv6.
Keywords: V2V, V2I, Mesh networks, MANET, OLSR, Wireless networks, IPv6, Mobility, NEMO, Service discovery, Zeroconf, Multicast, Semantic Networking, Ontology.

I. Introduction

The Imara team has been working on transport systems innovations for more than 10 years and is involved in several european projects in this field (CyberCars 2, COM2REACT, CVIS, etc.). Trying to provide safer and more efficient transportation solutions, several axes are explored, such as driver information systems, collaborative road data collection or even fully driverless vehicles.

In this context, the need for reliable and efficient communications between the vehicles (V2V) themselves or with the infrastructure (V2I) is increasing. There exist several wireless communication media, either proprietary of off-the-shelf (Wi-Fi, GPRS/3G, DSRC, etc.), each having their own specificities (coverage radius, delay, bandwidth). As a result, various techniques are used to achieve communication between the involved entities and interoperability becomes an issue. As discussed in [1], a communication system based on IPv6 (Internet Protocol) is advocated as the convergence layer between the various available physical technologies. IP convergence is the direction taken by some ITS cooperative systems project such as CVIS and by the Imara team itself.

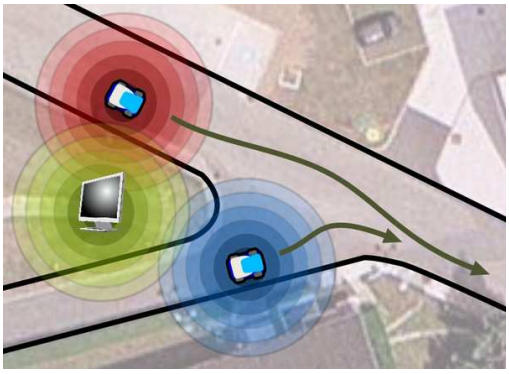

Figure 1. V2I: Two vehicles communicate with the infrastructure to determine when they can pass a crossroads.

The aim of this paper is to survey methods employed at Imara to provide a stable communication environment allowing driverless vehicles to interact with each other in several scenarios where information sharing is required. Section II details three use cases on which the team is working and Section III is presenting the common communication architecture allowing all three use cases. In section IV, we overview the ongoing implementations of this communication architecture and the results of live experiments. Our future works are outlined in section V before concluding this paper.

II. ITS networks use cases

A. Crossroads manager

Crossroads passing is a problem which notably requires communication in order to ensure safety and efficiency. On a first approach, it seems reasonable to rely on the infrastructure to ease the process [2]. In this scheme, vehicles arriving at a crossroads first contact the crossroads-regulating part of the infrastructure (Fig. 1), which they have previously found to be in charge of the current

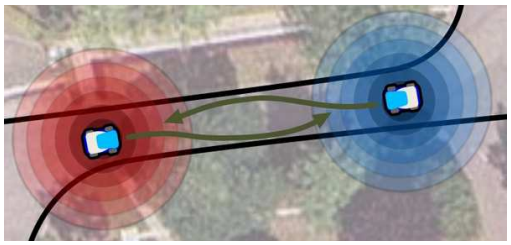

Figure 2. V2V: Two vehicles exchange their trajectory information.

segment of the road, and negociate when they can pass the intersection safely.

This scenario unveils several needs for the communications layer. A reliable physical network layer is a prerequisite, in order to ensure an acceptably small packets loss which would not burden the communication. This, however, is not sufficient. The embedded computers also need a way to find peers able to fullfill their requests (*e.g.* find a supervisor able to provide the upcoming crossroads' geometry). Only when the computers have identified each other regarding the services they provide can they begin running the actual algorithms and negociation to pass the intersection.

B. Trajectories exchange

In order to avoid collisions, driverless vehicles need to perceive they environment and make a prediction of it (theoretically, up to infinity [3]). Since it is not possible to have an accurate prediction of what the surrounding obstacles (vehicles, pedestrians, dogs) are going to do, the driverless vehicle needs to make a worst case assumption of their possible movement. Given that the vehicle will try to avoid a collision under any circumstance using a worst case assumption forces the vehicle to keep large enough distances from moving obstacles or to limit his maximum speed. This behavior is desired, since it is the only way to ensure a collision-free trajectory in an uncontrolled world.

This behavior, however, is suboptimal when two driverless vehicles are interacting. The future behavior of one such vehicle can be known with a better precision than that of a human driver. Thus, if driverless vehicles where able to communicate their plans (Fig. 2), the worst case scenario would become much more limited than when a human driver is assumed. Having tighter bounds on the future behavior of surrounding vehicles allows to reduce safety distances, and augmente displacement speed.

In order to exchange their trajectories, the vehicles need to communicate data such as arrays of position and speed periodically. The required bandwith is of a few tens of kilobits per seconds. These data need to reach all the surrouding vehicles that could possibly interact with the emitter (including opposite lanes) in the time horizon covered by the exchanged data. Technically, geometric information with a road network, existing walls and the like should be required to precisely define the set of "possibly interacting", but a simple geometric distance criterion can be considered as a good enough approximation. The most important requirement for this application is the latency, which needs to be known and bounded in order to be included in the safety margins computations. A latency of a few tens of milliseconds is good enough for most urban scenarios.

C. Semantic networking

An important issue emerges as soon as communication is desired in an heterogenous environment: the unambiguous representation of concepts to be shared amongst vehicles. By heterogenous we mean automatic vehicles (cybercars) possibly from different manufacturers, with different sets of sensors and actuators, and thus different internal way of representing the world. They only share the same communication system. What code (*i.e.* language) to use inside which context must be determined. A common way to describe (*i.e.* conceptualize) and represent (*i.e.* formalize) a context is required in order to share it. Ontologies have been designed for that purpose [4]. They are graphs which link concepts in both the taxonomic axis (*inheritance*: a cybercar `IS-A` car) and semantic axis (a cybercar `IS-POSITIONED` at some coordinates).

Systematically defining relationships between concepts provides a robust formal layer for reasoning. For instance, in the context of two vehicles willing to exchange informations on their respective environments, Vehicle A can sense the surroundings through a laser device while the vehicle B is relying on stereovision. Different sets of data are generated. A laser generates an array of tuples $(\text{angle}, \text{distance})$. Stereovision may generate various outputs like disparity maps, depth maps, etc. If the vehicles' respective environments are overlapping (*e.g.* at a crossroads), it is desirable to match their perceptions (*i.e.* do collaborative perception). Environment model matching, however, is not trivial, especially when relying on different sources and representation models. Ontologies allow to abstract the sets of data by enforcing the projection of individual representations on a common semantic system. They also offer a robust first-order logic reasonning framework which opens opportunities for possible cognitive approaches of decision making.

From a networking point of view, the use of ontologies mainly relies on the availability of a storage and query engine at the vehicle level, as well as means of transferring data between ontologies and their clients.

III. Communication architecture

A. Network hardware

The network hardware in the vehicles has been designed to be transparent. It is based on a small embedded router (a 4G Cube), with one ethernet interface for the intra-vehicular network, one Wi-Fi interface to connect to the VANET (Vehicular Ad-Hoc Network), and an optional

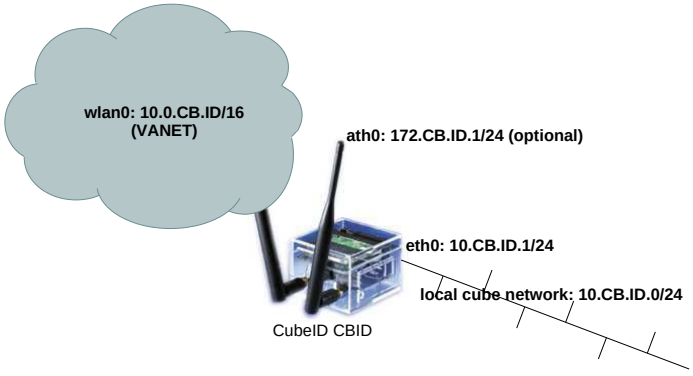

Figure 3. A classical 4G Cube has several interfaces which IP adresses and network ranges are determined from the router's ID for easy identification. The identifiers of the cubes are not below 1600 and have not yet reached 3199, thus giving the optional Wi-Fi network correct IP addresses in the 172.16.0.0/20 private range.

other Wi-Fi interface which usage can be adapted according to the needs (wireless intra-vehicular network, other VANET access, etc.). The router is based on a MIPS CPU and runs the Nylon Linux distribution. This operating system allows access to a wide range of network applications which only need be cross-compiled to work.

The network addressing is currently in IPv4 for historic reasons. It has been done in such a way that the topology is as simple as possible. Every subnetwork is clearly seperated from the others by using different IP address ranges. One of the main interest of this repartition is the *complete absence of NAT*. Every cube has an identifier, which is a 4-digit serial number. The IP addresses pools are derived from this identifier in order to simplify the determination of which router routes which part of the network (Fig. 3).

A cube holds all its host networks' parameters, and runs a DHCP server on each managed interface (ethernet and host Wi-Fi). A nameserver is listening on the hosted networks for request to be forwarded to a real DNS, if reachable. This setup allows to auto-configure the host clients and eases maintenance (a faulty cube can be quickly replaced by another one, as they share the same configuration).

B. Higher level specifications

From a vehicle-centric point of view, the network architecture can be summarized as in Fig. 4.

Optimized Link State Routing (OLSR): As a VANET is a higly dynamic network, it needs a network which allows for easy reconfigurations of the topology. In this respect, the OLSR protocol [5] is used to build a mesh network between the in-car routers, without the need for infrastructure. This protocol has the additional advantage of being able to advertise host networks, thus allowing for automatic update of the routing tables to include the in-car networks.

When the mesh topology is fully established, OLSR also accounts for nodes which are not directly reachable on the

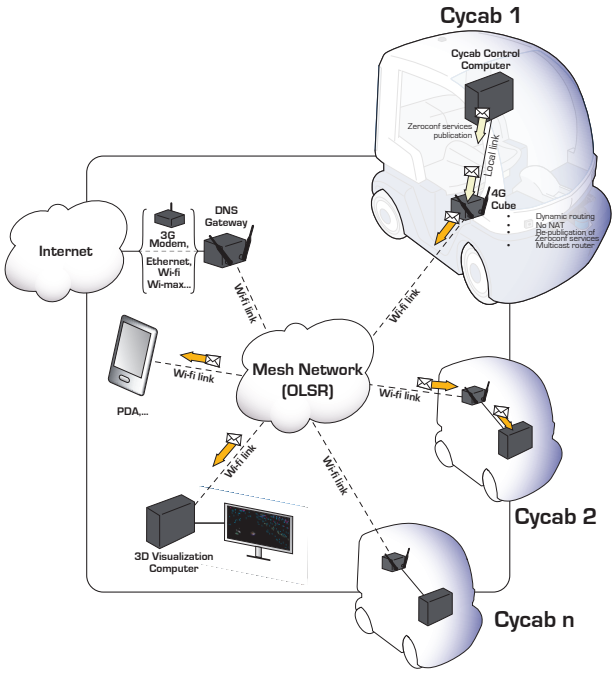

Figure 4. Connected through a mesh network based on OLSR, the in-car routers provide services like DNS-based services discovery or multicast routing. Thanks to the cubes, access to the VANET and other vehicles is almost totally transparent to higher level applications.

Wi-Fi link, due to the distance between two hosts. In this situation, a multihop path is added to the routing tables, transmitting the packets via one or more routers in the network until it reaches the recipient. This is interesting as it allows to build larger (in physical size) networks, and keep connections between vehicles for a longer period of time.

Additionally, a single masquerading router running OLSR and announcing the default route (0.0.0.0/0) as a host network is sufficient to provide internet access. This can be installed either as part of a minimalistic infratructure, or in some of the vehicles equipped, for example, with a GPRS/UMTS gateway. The other vehicles in the VANET can then get opportunistic access to internet without additional configuration.

Multicast DNS replication: In this automatically built network architecture using hand-maintained name-to-address mapping files would not be efficient. It is necessary to have a more automatic naming service. The works on Zero Configuration Networking (a.k.a. Zeroconf) have led to the development of Multicast DNS (or mDNS for short) [6]. This protocol is very similar to regular centralised DNS, except that every host runs its own DNS server and listens for queries it can answer on a multicast channel. The default domain in which resolve is made is `.local`.

One limitation of this protocol is that it is intended for small, routerless, networks and uses a link-local, *i.e.* non-

routable, multicast address. Fortunately, an opensource implementation of Zeroconf for Unix, Avahi[1], provides a *Multicast DNS reflection* facility. This system listens for mDNS requests or announces on its links, and forwards them on all the others. Running this daemon on a cube is sufficient for all its connected networks to have the same naming information. This way, names present on a local link are propagated to the OLSR mesh network and, in turn, replicated on the other routers' internal networks, thus creating a coherent naming space in which no IP address has to be fixed or remembered.

DNS-Service Discovery: Another work on Zeroconf has been focused on *service discovery* [7]. Based on a working DNS service, this protocol searches for specific records of service names in the `_tcp` and `_udp` subdomains (*e.g.* `_xmpp._tcp` or `_ntp._udp`).

Coupled with the mDNS system described above, this allows every computer in the VANET (either directly or through a router), to publish and dynamically update a list of services it can provide to the rest of the network.

Multicast routing: The classical method, when several clients need the same information, is to send it to them as a *unicast* stream, that is, replicate the whole transfer for every peer. This has two main drawbacks. First, the peers have to be known before being able to receive data. Then, replicating the whole data stream can have a big impact on the network availability, consuming more bandwidth for the same information.

To solve this problem, IP-level *multicast* is used. This allows several computers to listen to the same address (in the multicast address space 224.0.0.0/20). In such a situation, instead of duplicating the stream, a computer only has to send it once to the target multicast group and all the clients in need for this information will receive it. Multicast communication can be seen as a routable generalization of IPv4 broadcasting and, in most of the cases, seamlessly substitute to it.

Due to this one-to-many (or even many-to-many) scheme, it is not, at the time, really possible to maintain a connection-based communication, thus the UDP protocol is the default choice for multicasting. On the one hand, this allows vehicles arriving close to a multicasting peer to receive relevant traffic with (almost) no delay due to the connection handshake. On the other hand, this implies that no negociation on the format of transmitted data can take place. The vehicles should know in advance and be able to process the data format they will receive in a given multicast stream. This can be worked around, for example, by setting up a system where a new vehicle can get information about several streams giving the same information in differents formats and choose which one to listen to. The use of DNS-SD or ontologies for (resp.) passive or active discovery can provide an elegant solution to this problem.

[1] `http://www.avahi.org/`

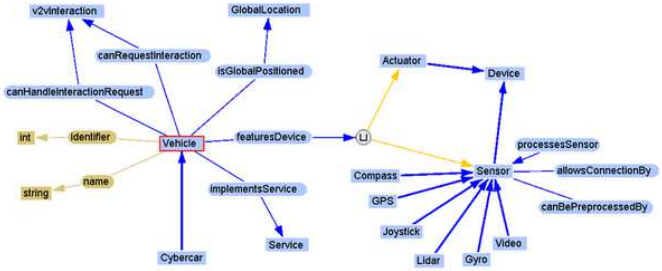

Figure 5. A small subset of the cybercar ontology. Ideas like "A vehicle features several devices, be they sensors or actuator" or "A cybercar, which is a vehicle, can request interaction with other vehicles" are logically formalized.

Using IGMP (Internet Group Management Protocol), clients can register to (or unregister from) multicast groups, informing their local router about this. When no client is subscribed to a group, the router can avoid forwarding traffic for this group, saving local bandwidth. On a wider network, the routers have to synchronize to properly set their multicast routing tables and send each other the multicast traffic their connected networks need. PIM-SM (Protocol Independent Multicast - Sparse Mode) [8] is a protocol which achieves this synchronisation task. The vehicular routers run a PIM daemon (`pimd`)[2] which listens to IGMP traffic on their interfaces, sets the routing tables appropriately and request traffic from other PIMd routers if needed.

Ontologies: A semantic layer of communication has been implemented through the use of a common ontology, written in Web Ontology Language (OWL) [9], an XML RDF specialization. This *cybercar ontology* stores a set of concepts attached to cybercars, ranging from datatypes to classes of possibly provided services. Fig. 5 illustrates on a small subset of the ontology some relationships around the concept of *Vehicle*.

It has been chosen to share an abstract version of the cybercar ontology (*i.e.* without instances) amongst vehicle, and let each of them instantiate (thus, specialize) it.

An ontology server has been introduced on each vehicle to ease the use of ontologies for clients. This server is written in Java and relies on the Jena library [10] to interact with OWL ontologies. The server is intended to be embedded in the vehicles, in parallel to the main vehicle control software. Interaction with it is done via by XML-RPC (through the Jetty Java webserver and the Redstone XML-RPC library).

When started, the ontology server registers a DNS-SD service (`_ontoserver._tcp`), and clients may then query the server through standard XML-RPC requests. The interface exposes four main operations: `query`, `add`, `updateObject`, `remove`, and a fifth one, for debug purposes, `owlDump`, which dumps the current in-memory on-

[2] `http://netweb.usc.edu/pim/`

```
SELECT ?position
WHERE {
VehicleA cc:isGlobalPositionned ?position .
}
```

Figure 6. A simple SPARQL request to get vehicle's position from the ontology. `VehicleA` contains the URI of the vehicle and the `cc:` prefix stands for the cybercar ontology namespace.

Figure 7. A 4G cube has two MiniPCI slots and two antenna sockets which are used to install two Wi-Fi cards. One is a 802.11b used for the VANET (to achieve longer communication ranges), while the other optional one can run in G mode to provide in-car wireless connections.

tology to an OWL file. The most used one is the `query` request: it wraps a SPARQL request [11] which is handled by the Jena library to perform high-level queries on the ontology. For instance, a query to get the current position of Vehicle *A* is shown on Fig. 6. The vehicle control software is in charge of instantiating the ontology, then updates it periodically by sending `add` and `updateObject` requests containing statements (*i.e.* a (subject, predicate, object) triplet).

As a demonstrator, the ontology-based architecture is used to visualize, on an external 3D engine, the live positions of the vehicles. The 3D visualizer can dynamically discover vehicles as soon as an ontology server starts (thanks to Zeroconf), and then draws the vehicles at their current locations. The 3D visualizer in under heavy development and will soon be able to take full advantage of the ontology by sending abstract requests to cybercars like "What sensors do you hold?" or "What obstacles do you see in front of you?". The results will be then integrated to the 3D environment.

IV. Implementation details and experiments

A. Hardware settings

The cube sports both 802.11b and 802.11g Wi-Fi cards (Fig. 7). The 11Mbps card is used to connect to the VANET. The other card, not present in all the cubes, can be used for versatile purposes, but is mostly used to provide wireless connection to other appliances in the vehicle like PDAs or laptop computers.

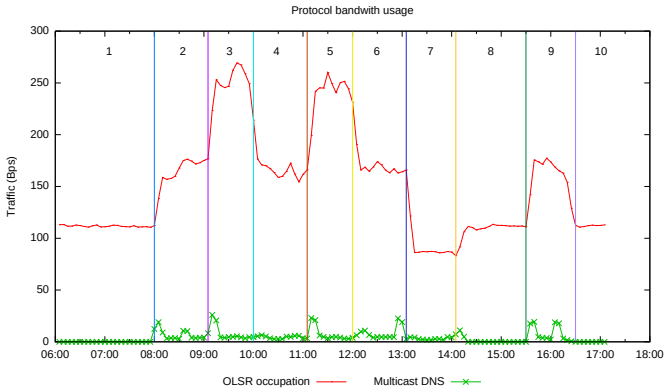

Figure 8. Bandwidth occupation on the mesh network by OLSR and mDNS control packets. OLSR nodes generate roughly 60 Bps per node of constant traffic, whereas mDNS is almost neglectable, generating peaks of less than 30 Bps when a new client is started.

B. Network and bandwitdh considerations

Bandwidth usage: Several of the protocols implemented in our VANET rely on periodic reemission of control packets, the first in line being OLSR itself, which has to keep track of peers and host networks. Another traffic-generating protocol (both periodic and bursts), is Multicast DNS and its reflections. As the bandwidth is limited on the mesh network, it is important to keep track of its usage by ever-running protocols for it not to collapse under its own weight. Series of tests have been run over a day, while measuring the occupied bandwidth (Fig. 8). The scenario has been as follows:

1) initially, two OLSR routers are present on the mesh network, without host computers;
2) a Zeroconf-enabled laptop computer running OLSR is added directly to the mesh network;
3) a new cube is connected to the network, with a host Windows computer running Apple's Bonjour mDNS implementation;
4) the latter cube is removed;
5) the same cube is readded to the network without its host computer;
6) one of the original cubes is removed;
7) another cube is removed;
8) the laptop computer is removed, and one of the cube is added;
9) the laptop computer is re-added;
10) the laptop computer is finally removed.

Multicast routing: Regarding the multicast routing tables propagation through PIM, tests have been run using the most typical case of multicasting: video streaming. The experimental setup features two cubes (single hop mesh network) with one computer on each host ethernet link. One of the hosted computers starts multicasting a video (say, a video from a camera watching the upcoming road) on a non-local multicast stream (say 225.0.0.1). The other computer then requests to join the multicast group.

Once the request to join has been sent, a delay of 5 seconds has been constantly observed before the group was available on the destination network. This may not be a problem in most of the cases, but can become an issue when early information is needed, as in the exposed use-case of trajectories exchange. To avoid this problem a (discutable) workaround may be to statically register routes to the router for this specific traffic to be always forwarded. This, however, goes against the argument of bandwidth saving, as this very traffic will always be routed, even on networks without clients requiring it.

V. Next plans

IPv4 fixed addressing is functional in the context of our small network of less than ten vehicles. However, porting this framework to real situations may prove impossible due to the limited number of addresses available in the IPv4 local pool. Collisions can be expected to happen between vehicles' identifiers, thus rendering communication malfunctionning or impossible. Taking into account the goal of giving internet access to the vehicles in the network worsens the problem as computers with only local IP addresses will try to send global traffic. This can be worked around, "as usual" with NATs, but in this context of mobile network nodes, it is inevitable that their NAT router will change, irremediably breaking their connections every less than one kilometer.

All these problems find an elegant solution in the use of IPv6 (and its Network Mobility extension [12]). The autoconfiguration and duplicate address detection (DAD) algorithms of plain IPv6 will remove the address collision issue while NEMO can provide for seamless internet communication has well as inter-vehicle communication over longer distances via internet routing. Moreover IPv6 is supported by several OLSR and mDNS implementations, making the migration of the current architecture quite straightforward. A Mobile IPv6 testbed is currently being developed at Imara to provide the necessary framework to ease the migration.

Regarding the specific routing problem for multicast in a mesh network, an enhanced version of OLSR, MOOLSR [13], takes care of synchronizing the routes between the mesh routers. This has not been implemented yet in the presented architecture, but is part of ongoing works towards a more functionnal multihop multicast mesh network.

VI. Conclusions

This network setup is quite interesting for inter-vehicular communications. Basing itself on off-the-shelf hardware and protocols, used in regular networks, like multicast and Zeroconf, or more ad-hoc solutions, like OLSR mesh networks, it provides a functional working base for higher level applications. Moreover, it also sets the base for a fully autoconfigurable network, where the adresses of the embedded components may be arbitrarily or randomly determined, but always abstracted by way of user-friendly or semantically-rich names. Within this versatile framework, one can easily imagine different types of communications cohabitating, like planification, information exchanges between autonomous vehicles, smooth

forwarding of internet traffic or inter-vehicle "leisure-oriented" network traffic (chat, games, etc.).

Other solutions are, however, yet to be implemented. The network is functional as is, but other techniques (IPv6/NEMO) could greatly increase its possibilities and ease of use. Works will be led in this direction, with total transparency between a regular network and an in-car one as a goal.

References

[1] T. Ernst, "The information technology era of the vehicular industry," *SIGCOMM Comput. Commun. Rev.*, vol. 36, no. 2, pp. 49–52, 2006.

[2] O. Mehani and A. de La Fortelle, "Trajectory planning in a crossroads for a fleet of driverless vehicles," in *Computer Aided Systems Theory - EUROCAST 2007, 11th International Conference on Computer Aided Systems Theory, Las Palmas de Gran Canaria, Spain, February 2007*, A. Quesada-Arencibia, J. C. Rodríguez, R. M.-D. jr., and P. Roberto Moreno-Díaz, Eds., May 2007, in press.

[3] L. Bouraoui, S. Petti, A. Laouiti, T. Fraichard, and M. Parent, "Cybercar cooperation for safe intersections," in *Proc. of the IEEE Int. Conf. on Intelligent Transportation Systems*, Toronto, ON (CA), September 2006. [Online]. Available: `http://hal.inria.fr/inria-00126664`

[4] T. R. Gruber, "Towards principles for the design of ontologies used for knowledge sharing," in *Formal Ontology in Conceptual Analysis and Knowledge Representation*. Kluwer Academic Publishers, 1993. [Online]. Available: `http://citeseer.ist.psu.edu/gruber93toward.html`

[5] T. Clausen and P. Jacquet, "Optimized Link State Routing protocol (OLSR)," 2003. [Online]. Available: `http://citeseer.ist.psu.edu/clausen03optimized.html`

[6] M. K. S. Cheshire, "Multicast DNS," Internet Draft, `http://files.multicastdns.org/draft-cheshire-dnsext-multicastdns-06.txt`, February 2006. [Online]. Available: `http://files.multicastdns.org/draft-cheshire-dnsext-multicastdns-06.txt`

[7] ——, "DNS-based Service Discovery," Internet Draft, `http://files.dns-sd.org/draft-cheshire-dnsext-dns-sd.txt`, February 2006. [Online]. Available: `http://files.dns-sd.org/draft-cheshire-dnsext-dns-sd-04.txt`

[8] D. Estrin, S. Deering, D. Farinacci, V. Chinggung, and L. Liming, "Protocol Independent Multicast (PIM): Protocol specification," 1994. [Online]. Available: `http://citeseer.ist.psu.edu/deering95protocol.html`

[9] S. Bechhofer and al., "Web ontology language (OWL) reference," W3C, Tech. Rep., 2005. [Online]. Available: `http://www.w3c.org/TR/owl-ref/`

[10] J. J. Carroll and al., "Jena: implementing the semantic web recommendations," in *WWW Alt.* New York, NY, USA: ACM Press, 2004, pp. 74–83.

[11] E. Prud'hommeaux and A. Seaborne, *SPARQL Query Language for RDF*, W3C, July 2005. [Online]. Available: `http://www.w3.org/TR/rdf-sparql-query/`

[12] T. Ernst and A. de La Fortelle, "Car-to-car and car-to-infrastructure communication system based on NEMO and MANET in IPv6," in *Proc. of 13th World Congress and Exhibition on Intelligent Transport Systems and Services*, October 2006.

[13] A. Laouti, P. Jacquet, P. Minet, L. Viennot, T. Clausen, and C. Adjih, "Multicast Optimized Link State Routing," Inria, Tech. Rep., 2003. [Online]. Available: `http://hal.inria.fr/inria-00071865`