All Electrical Vehicles Connected to the Internet: Implementation and Experiments

Franco Callegati, Aldo Campi, Walter Cerroni,
Matteo Marano, Claudio Rossi
D.E.I. "G. Marconi"
University of Bologna, Italy
Email: name.surname@unibo.it

Giovanni Pau
Computer Science Department
UCLA, Los Angeles, CA, USA
Email: gpau@ucla.edu

***Abstract*—In this paper we present a test-bed implementation of an autonomic vehicular network infrastructure to configure and execute vehicular network services for All Electrical Vehicles (AEV). The manuscript reports a field experiment implementing the bridge between the AEV control network and the Internet as well as the signaling infrastructure which allows the autonomic configuration of the communication services that may be implemented exploiting such connectivity.**

***Index Terms*—Vehicular Networks, All Electrical Vehicles, CAN, SIP, Arduino, Smart Electrical Vehicle Recharge Systems.**

I. Introduction

Vehicular Networks (VANETs) have been subject of research studies for more than a decade [1], as a key enabling technology for Intelligent Transportation Systems (ITS). However the deployment of such infrastructures has been limited and ITSs mainly progressed by means of other infrastructures such as GPS, dynamic road panels, automatic speed and traffic enforcement cameras etc.. Moreover the most significant ITSs to date find application to travel information, traffic and demand management, smart ticketing or urban logistics, and focus much less on issues related to vehicle usage, user experience etc.

This paper tackle the All Electrical Vehicles area of application, as one of the most significant innovations in the automotive market of the next decade [2], and aims at showing that there is a whole bunch of innovative services that can be deployed, well beyond traditional ITSs, and at discussing what kind of infrastructure may effectively support them.

A practical experiment is reported, focusing on the smart management of the battery usage and recharging phase. This was chosen as a case study since the new "fueling medium" has characteristics which are inherently different from the traditional ones and the user should be assisted as much as possible in the transition.

The enabling technology is a communication system which is able to talk with the AEV power and engine management system and, at the same time, is able to exploit vehicle to infrastructure (V2I) connectivity to upload/download data to/from external databases and applications, exploiting existing wireless networks, namely GPRS, 3G, Wi-Fi hot spots and Wi-Max.

Although these networks are used "as they are", the communication characteristics and the methodology to exchange information must be adapted to the specific goals and geographic area of operation. For this latter reason, besides the pure connectivity, the network infrastructure must support *service management* functionalities to configure the proper *service profile* based on the *knowledge* of current location, type of vehicle, connection quality, etc. This *autonomic* capability of the network must allow proper communication service configuration without any need for specific actions by the end-user. To this end a proper signaling infrastructure is required, which in the following will be called *Transport Service Layer* (TSL).

This manuscript reports the results of the research activity developed in collaboration by three different research groups at the University of Bologna and at the University of California, Los Angeles, which addresses the aforementioned issues, with implementation and practical demonstration of the required functionalities as well as of their performance.

In Section II the results of the implementation of a flexible network interface for an AEV based on a microcontroller system are reported . This system enables the communication between the AEV and the Internet and is the basic building block to implement advanced communication services for the vehicle. In Section III a possible architecture for the TSL is presented and some experiments showing its effectiveness exploiting a VANET test-bed are reported in Section IV. Finally in Section V some conclusions are drawn.

II. Bridge the AEV control network and the Internet

The first result is the implementation of a network interface for an AEV which acts as a bridge between the vehicle control network and the Internet and is the key element to enable new services related to the management of the vehicle characteristics and functions. This system is the motivation of the second part of the work described in Sec. III.

978-3-901882-56-2

A. The LEMAD Power System

The AEV considered in this work is equipped with an electrical engine and battery pack managed by a powertrain which is developed by the Laboratory of power electronics, electric drives an electric machines (LEMAD) at the University of Bologna. This powertrain is designed for the propulsion of lightweight vehicles, below 1 ton of maximum mass, i.e. city cars and leisure vehicles as side-by-side.

The LEMAD powertrain is an integrated system made of two parts, the former for the control of the power stage which is shown in Fig. 1 and the latter implementing the internal vehicle control and communication system whose architecture is shown in the dashed line box on the left of Fig. 2.

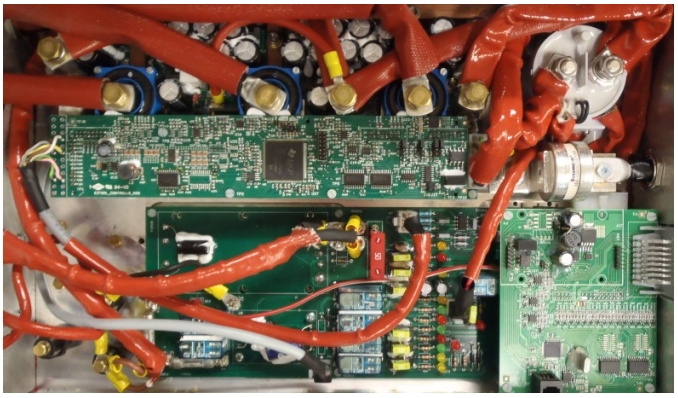

Fig. 1. Main box of the experimental powertrain.

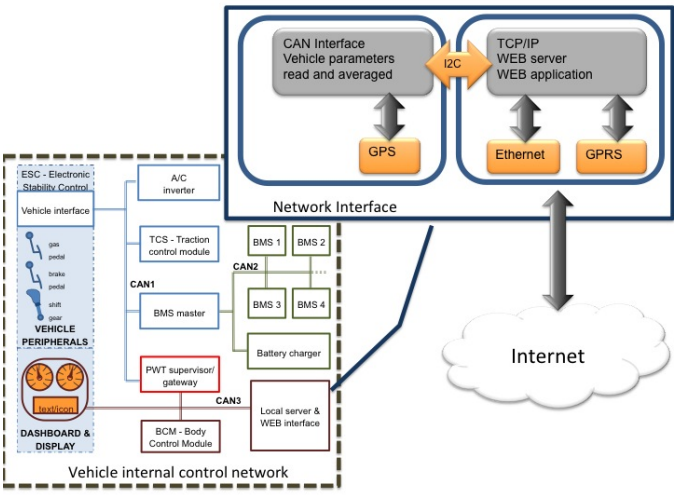

Fig. 2. The vehicle internal control network.

The power circuit is a traditional full-electric driveline. A battery pack, composed by n-cells in series is connected through the main DC bus to the traction inverter. This inverter supplies the traction motor, which is usually integrated with a mechanical reduction gearbox and differential. The main DC bus supplies a plurality of other power converter modules: the DC/DC converter which provides power to the PWT auxiliary circuits, the DC/DC converter which provides power to the vehicle loads, the inverter for the motor of the air conditioning compressor, the on-board battery charger.

The battery pack is conditioned by using a modular Battery Management System (BMS). Each module is able to monitoring temperature and voltage of 8 cells. The BMS modules implement the passive equalization of cells during recharging by dissipating a small portion of the recharge current.

Each device is controlled by a microcontroller and all devices are interconnected through a high speed CAN bus network. On each device, the control system architecture is based on a multilayer structure. The control algorithm is split in low level, high level and communication level. In particular the high level control algorithm is interconnected through the CAN bus with other peripherals for data sharing, while the low level take the references from the high level and controls the hardware device (power electronics, signal conditioning, etc..).

B. The Internet Interface

The vehicle is connected to the Internet by means of an embedded system providing networking functions based on the Arduino hardware and software platform [3]. The Arduino UNO [4] mother board was used as the basic building block for the prototyping phase reported in [6]. Starting from the first prototype an ad hoc integrated board was developed, which is used for the tests reported here.

The overall block diagram is shown in the continuos right left box of Fig. 2. The described system is made of two logical sub-blocks; the former communicating with the LEMAD powertrain via the CAN bus, the latter implementing the network interface. Data are exchanged between them using an on-board I2C serial bus [5].

The network interface module is equipped with an Ethernet interface (which can be replaced by a Wi-Fi module) and a UMTS/GPRS module can be added, so that several networking opportunities are available at once and can be opportunistically used depending on the current conditions. Typically wireless technologies able to guarantee a wide area coverage (GPRS/UMTS) are suitable to provide con nectivity when the vehicle is in motion, while wired or wireless technologies (Ethernet, Wi-fi) providing larger band but shorter reach are suitable to provide connectivity when the vehicle is standing still parked and/or re-charging.

The choice of using the Internet standard protocols to present the information to the final user guarantees a very wide accessibility, while the modular hardware architecture may easily accommodate new networking technologies that may become available by replacing the GPRS module with new interfaces when available (for instance the LTE).

The system is also equipped with a large capacity storage unit (SD card). The relevant vehicle data may be stored in the storage unit for backup reasons or as a sort of buffer for periods when the connections are lost. In the latter case the stored data may be uploaded to the server by the web application when the connection is re-established.

Overall this system fulfills a number of tasks:

- read relevant operating data from the vehicle power train;
- parse these data and covert them to an "Internet friendly" format;
- make the data available for consultation to the final users in a web format, meaning as a set of web pages which can be navigated with a conventional Internet browser;
- write the data on a remote server by means of a suitable Internet application so that this data may be processed by an off-line application.

The Internet interface may also be used to enable a sort of "social interface" which let the car directly sharing on social networks such as FaceBook (www.facebook.com), Twitter (www.twitter.com) information such as the working conditions of the vehicle, what is the state of the battery and so on. The car could upload on social networks even its own statistics on mileage, power consumption and speed, thus fostering a sort of "behavioral competition" between final users for better use of energy etc..

C. Experimental Results

The overall Internet connection system was tested in operating conditions, aside the experimental powertrain, designed for supplying an induction motor with an output power rating of 10 kW, by using a battery pack composed of 32 ion-lithium cells connected in series.

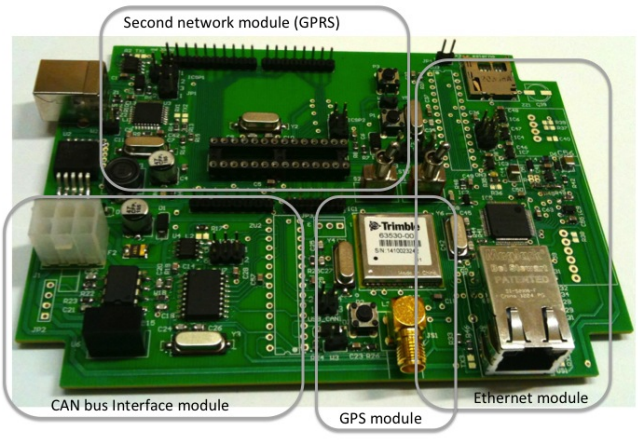

Fig. 3. The web interface board.

The picture in Fig. 3 portraits the custom Internet interface and the relevant logical blocks. It is equipped with 3 microcontrollers used to interface with the CAN bus, with the Ethernet interface and with the GPRS interface. The microcontrollers implement all required data parsing and exchange between modules by means of the I2C internal bus. In the figure the microcontrollers as well as the GPRS shield are missing to make the overall picture more readable.

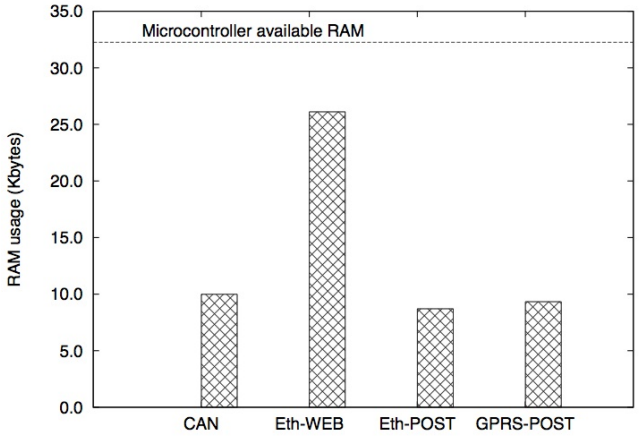

Fig. 4. ATMega328 system RAM usage for the various system options.

Fig. 4 shows the amount of ATMega328 programmable RAM used by the various software modules. Overall the RAM available is limited to 32256 bytes, given by the total RAM on board minus the space required by the boot loader necessary to program the micro-controllers. The module implementing the CAN interface and the web applications downloading the monitored data to a remote server via Ethernet, or GPRS, are indeed well below the limit and use about 30% of the overall capacity. The module implementing both the web application and a full functional web server with web pages which can be navigated from a remote browser for on-line monitoring of the state of the vehicle is more complex and requires 26 of the 32Kbytes available, for an overall RAM occupancy of approximately 80%, still acceptable for normal operating conditions.

The compatibility of the overall system processing capacity with the workload related to the amount of incoming messages over the CAN bus was tested by changing the rate of such messages read on the CAN interface. In fact sending data on the web at the speed they flow on the CAN bus for real time control of the vehicle is not a viable option. In the implementation the Arduino board implementing the CAN interface acts as follows:

- reads all the CAN messages;
- averages the parameters read over a subset of M values;
- stores the average in an array in its RAM (arrays are different for different ID-nodes, so that an array stores consistent data values);

- the arrays have a fixed dimension of N and when are full a packet is built and sent to the Arduino board implementing the Internet interface over the I2C bus.

In the experiment here presented $N = 20$ and M is changed in order to vary the inter-arrival rate of messages to the network interface from 0.2 to 10 seconds. (i.e. the parameters related to a given vehicle function are averaged over a set of M values and then submitted in series of 20). We logged 1000 messages flowing on the I2C bus, with average message size of 150 bytes. In all measured cases the load on the I2C bus was well below its nominal capacity, but some messages were lost. In fact the processing load required for transmitting a message to the network interface sets a threshold. If a new message arrives when the microcontroller is still transmitting the previous one, it is lost. Fig. 5 shows the results of this test. Rather obviously the Ethernet interface which has a higher transmission speed and a dedicated layer 2 processor requires an average message processing time of approximately 0.6 seconds, and when the message inter-arrival time is above this value causes a very limited percentage of lost messages. On the other hand the microcontroller coupled with the GPRS interface requires a longer time to process and send a message, around 3.5 seconds and therefore the percentage of lost messages is higher. These data suggest the trade-off dimensioning rationale for M. A large M will provide a monitoring of the vehicle averaged over a larger time frame, which means less timeframe resolution, but very few messages will be lost guaranteeing a good reliability. On the other hand a small M will provide a finer time frame resolution but, depending on the networking conditions, many messages may be lost and monitored data missed.

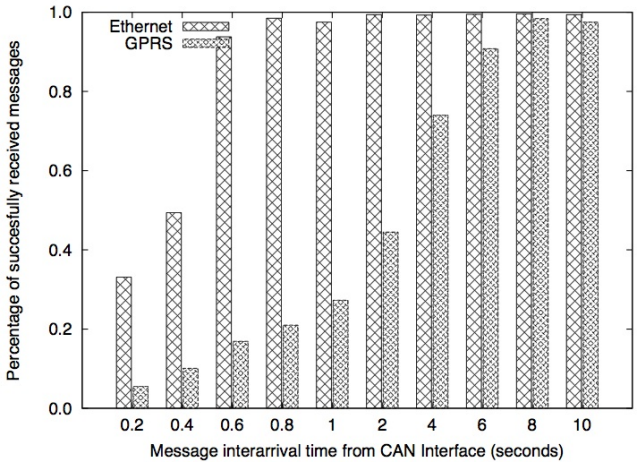

Fig. 5. Percentage of messages successfully forwarded by the network interface as a function of the inter-arrival time from the CAN interface.

Finally it is important to mention that the powertrain, equipped with the vehicle web interface system, is installed on the TGS RACE electric buggy shown in Fig. 6.

Fig. 6. Picture of the electric two-seat buggy TGS pirate equipped with the web interface system.

III. Autonomic Configuration of the AEV Networking Services

The AEV Internet interface provides manufacturers and users with the capability to access and implement V2I communication but, generally speaking, every specific network based service requires a proper configuration. For instance let's imagine a simple scenario where the vehicle uploads its operation data on a local municipality server to elaborate statistics about the AEV usage in the area. Such service can be run in different geographical areas but requires an ad-hoc configuration to know which is the address of the server to upload data etc.. Obviously it is not realistic to imagine a configuration run by the user when needed, but the vehicle and network must self adapt upon the connectivity and the service availability as the vehicles moves in space.

This requires an autonomic network capability which is supported by a signaling layer which will be called Transport Service Layer or TSL in the following. The TSL interacts with both user applications and network infrastructure to configure the proper communication service for a given application without user intervention. The topic of designing a TSL is not new to networking [7], [8], [9]. Nonetheless, at the best of our knowledge, no applications to the vehicular networking scenarios have been reported so far.

The research group at the University of Bologna proposed a TSL implementation for grid [10] and cloud computing [11], which has a very general architecture and can seamlessly be applied to VANET.

A. The Transport Service Layer

The TSL implementation used here decouples the signaling protocol, which defines the syntax used for service negotiation, from the descriptive language, which provides

the semantics to describe the service features (resources to be used, requirements to be satisfied, etc.).

The Session Initiation Protocol (SIP) is used to implement the service-independent signaling layer. SIP maps the communication service negotiation into a stateful end-to-end signaling flow.[1]

Then a semantic language oriented to the description of communication resources, the *Network Resource Description Language* (NRDL) [13], is encapsulated in the SIP message payload. NRDL is based on the Resource Description Framework and puts in place the capability to answer general questions such as: *"How should the application say what it needs for a given service instance?"* and/or *"How should the network service equipment say what sort of communication services are possible according to the network resources currently available?"*

In VANETs, the network service activation process is mainly based on an publish/subscribe paradigm [14] and, by exploiting the NRDL multilayer descriptive characteristics, it is possible to publish resource availability and issue resource requests, as well as to support subscription to the desired service. On the other hand, due to its session management capabilities, SIP guarantees the communication consistency and correct message delivery.

The TSL operations can be summarized as follows:

- according to the user/application's preferences, a NRDL document is compiled describing the communication service request;
- a SIP session is started to carry the request and, by means of the opened SIP session, the NRDL document traverses the network from sender (vehicle) to receiver (service server in the infrastructure);
- by means of the SIP and NRDL negotiation capabilities, sender and receiver arrange to negotiate a communication service matching not only the user/application needs, but also the available resources necessary to activate it;
- once the service is negotiated the actual transfer of information may start and its characteristics may be managed (e.g., modified, suspended, re-routed etc.) via the TSL signaling.

These basic operations are performed in a self-consistent manner thanks to the session protocol which provides consistency in the various dialog phases. Obviously, since network services can be composed of several steps or needs keep-alive, the information exchange may continue until the required network service is completely established or finally canceled. Further technical details on the TSL architecture and implementation can be found in [12].

B. the C-VeT test-bed

The TSL previously described was tested in a real life VANET environment, which is the C-VeT infrastructure at the UCLA Computer Science Department. C-VeT provides both V2V and V2I connectivity [15] and was designed to be an open platform to support research on vehicular networking since the campus, with its 10 acres of urban development, reproduces many of the scenario, propagation, and communication challenges typical of a urban environment in a realistic manner, yet at a relatively small scale.

MobiMESH is the the C-VeT Mesh Network featuring a hybrid mesh network architecture built upon a Mesh Backbone composed of MobiMESH wireless mesh routers which provide the routing and mobility management infrastructure and an access network which can be used by standard Wi-Fi clients to get connectivity. MobiMESH is based on the ad-hoc network paradigm, where all nodes and mesh routers collaborate to route traffic. The backbone network is also responsible for the integration with the wired network, through devices called Gateways, that are equipped with a wired interface and that can route traffic to the Internet. The access network is rather flexible and operates in the infrastructure mode, so that clients perceive the network as a standard WLAN and behave accordingly.

IV. Field Experiments

The aim of the experiments run is to prove the effectiveness of the TSL in providing autonomic configuration of AEV communication services. The case study scenario is that of a smart recharge management service for AEVs. Starting from the assumption that the AEV is equipped with the Internet interface as described above, the scenario is a Smart Management System for recharging AEVs (SMS/AEV in the following), extending what proposed in [16] beyond peer-to-peer communication among vehicles. Here the service is enriched with a provider infrastructure, which may support both peer-to-peer message exchange between vehicles or communication with a centralized local service provider. Basically the SMS/AEV automatically collects information about availability and pricing of recharging stations in the area nearby the vehicle and matches the characteristics of such stations with those of the vehicle and the current state of the batteries, then presents to the user the available recharging options on the dashboard.

The service subscription logic is supported by the TSL. The basic assumption is that the user subscribes to a SMS/AEV active on a rather wide geographical area (nation/region). When the user wants to get information about recharging stations, he/she registers to the service by contacting the proxy of the SMS/AEV service provider and providing is current location available by means of the GPS module available in the Internet interface. The server is available on the Internet and may be contacted with whatever network connectivity is opportunistically available. The server will re-direct the user to an application server which has detailed and updated information about

[1]Note that the SIP session may consist of a set of user activities that can be logically correlated, with several exchanges of information (either in parallel or in series) as part of the same session.

the recharging options in the geographical area nearby the vehicle.

These two actions follow the logical rationale shown in Fig. 7. Again this happens with no need for specialized human intervention. The dialog deals with issues that span across different logical communication layers, the recharging-related issues being at the application layer, the setup of the communication channel with the local application server (with proper authentication and/or data protection) being a network layer issue.

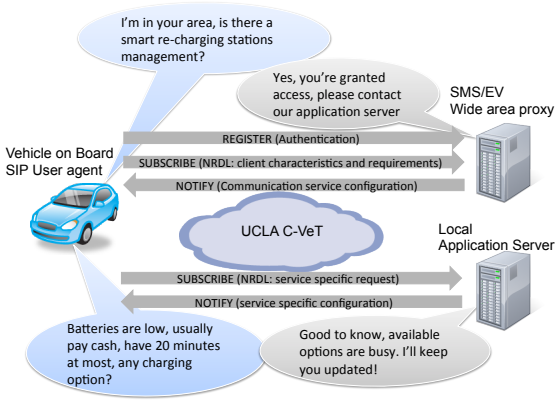

Fig. 7. Logical description of the TSL operations when a AEV enters a geographical area where the communication service needs to be secured by means of a VPN.

Figure 8 shows with some detail what happens in the experiment. At first the user registers to the service by contacting the wide area proxy, which is a SIP server dealing with the authorization and authentication of the vehicle (user) accessing the SMS/AEV service. This is a server on the Internet which, as already pointed out, may be managed by a wide area service provider, for instance a mobile network provider or an energy provider. During the registration phase, the vehicle gets the information regarding the application server (or servers) dealing with SMS/AEV in the area where it is traveling and which it would not know otherwise. Then it contacts such server to get detailed information about the local service implementation. In the example the assumption is that the information about recharging opportunities may be retrieved from a specific municipality server but also from other vehicles, according to the paradigm presented in [16].

The overall scenario described above was experimented "in the field" using C-VeT. The details of the data flow of the experiment are plotted in Fig. 9, where the whole V2I signaling packet exchange is shown. In this specific example the network connection used the C-VeT Wi-Fi MobiMESH.

At first the vehicle registers (REGISTER: message # 57), then it provides information about its requests (SUBSCRIBE: message # 62). This subscription is at first

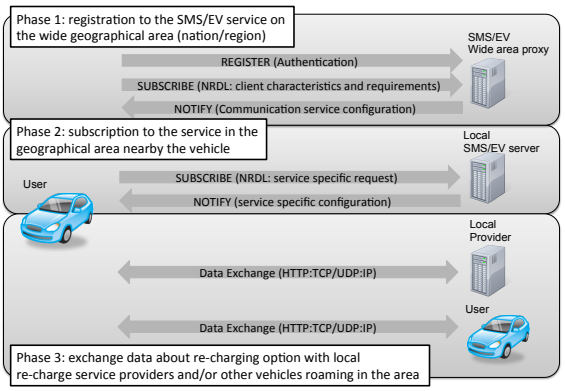

Fig. 8. The various phases of the user communication to retrieve data from the SMS/AEV: registration to the service, subscription to a local application provider, communication of data for recharging options with local providers and/or other vehicles.

refused by the proxy, which requires authentication to the user, which is then provided (SUBSCRIBE: message # 64). As a consequence the SMS/AEV server answers with the communication service profile configuration information (NOTIFY: message # 66). In this message it also says to which application server the vehicle has to connect to get the detailed information about recharging availability in the area. This signaling flow is fully compliant with the SIP registration and subscription specifications, no extensions and/or customizations to the protocol are required.

Now the vehicle subscribes to the application server (SUBSCRIBE: message # 70), possibly providing more detailed information about its characteristics and/or recharging capabilties. The server provides all the information to complete the service configuration in the subsequent reply (NOTIFY: message # 72) which is periodically replicated to keep alive, refresh and update the configuration as discussed above. Some of this NOTIFY messages may be lost if the connectivity is temporarily lost, without affecting the overall dialog.

As the figure shows the signaling phase is completed in just about 50 seconds with a very limited bandwidth usage, suggesting it may be well compatible with vehicular applications. Following message #73 the communication service is fully configured and the vehicle may start downloading data from the server.

Fig. 10 shows, as an example, the NRDL content in the SUBSCRIBE/NOTIFY messages exchanged between vehicle and municipality server (messages #64 and #66) and the SUBSCRIBE/NOTIFY messages exchanged between vehicle and application server (message #70 and #72). In the former the vehicle requests details about the server to be contacted for data upload, the protocols to be used etc. and receives answers about that. In the latter it receives information from such server, for instance

related to the recharging station in the area (for the sake of brevity the data referring just to one recharging station are presented here). This information may be presented to the final user in the AEV dashboard as soon as it is received or depending on the overall system configuration. For instance it could appear if and only if the battery level are below a certain level, an information which is available to the system already thanks to the interface with the CAN bus.

V. Conclusion

This paper presented a test-bed implementation of an autonomic vehicular network infrastructure to configure and execute network based services for All Electrical Vehicles. The paper presents an implementation of the interface of the electrical vehicle with the Internet based on a multi micro-controller architecture, which provides the bridge between the internal vehicle control network and the Internet.

The performance of such system were experimentally analyzed showing that the bridge must take care of averaging the parameters reported over the CAN bus over a suitable time window to keep the message flow to the Internet interface at a speed compatible with its packet processing capabilities.

Then a signaling architecture to provide autonomic control of the vehicle communication services was presented. A test-bed implementation of such architecture was tested showing that it can provide the required functionalities with a response time that is compatible with typical applications time frame.

Acknowledgments

This work was partially carried on while Prof. Callegati was visiting the UCLA - CSD supported by EINS, the Network of Excellence in Internet Science (http://www.internet-science.eu/) through European Union's 7th Framework Programme under Communications Networks, Content and Technologies, Grant Agreement no. 288021.

References

[1] K. L. Hannes Hartenstein, Ed., *VANET Vehicular Applications and Inter- Networking Technologies*. Wiley, 2010.

[2] A.Y. Saber, G.K. Venayagamoorthy, *One million plug-in electric vehicles on the road by 2015*, Proc. of 12th International IEEE Conference on Intelligent Transportation Systems, St. Louis, MO (USA), 2009.

[3] *Arduino*, `http://www.arduino.org`

[4] *Arduino UNO*, `http://arduino.cc/en/Main/ArduinoBoardUno`

[5] *I2C Bus*, `http://www.i2c-bus.org`

[6] F. Callegati, A. Pilati, M. Ramilli, C. Rossi, "Web management of electric vehicle fleets", Proc. EVS26 International Battery, Hybrid and Fuel Cell Electric Vehicle Symposium, Los Angeles, U.S.A., 2012.

[7] D. D. Clark, C. Partridge, J. C. Ramming, J. T. Wroclawski, "A Knowledge Plane for the Internet," Proc. SIGCOMM '03, Karlsruhe, Germany, 2003.

[8] A. Galis, H. Abramowicz, M. Brunner (eds.), "Management and Service-Aware Networking Architectures for Future Internet", Position paper, MANA Future Internet group, 2009.

[9] Keith Knightson et al., "NGN Architecture: Generic Principles Functional Architecture, and Implementation", IEEE Communications Magazine, Vol. 43 (10), 2012.

[10] F. Callegati, A. Campi, G. Corazza, D. Simeonidou, G. Zervas, Y. Qin, R. Nejabati, "SIP-Empowered Optical Networks for Future IT Services and Applications," *IEEE Communications Magazine*, Vol. 47 (5), pp. 48-54, 2010.

[11] F. Callegati, W. Cerroni, A. Campi "Application scenarios for cognitive transport service in next-generation networks", *IEEE Communications Magazine*, Vol. 50 (3), pp. 62 -69, 2012.

[12] F. Callegati, W. Cerroni, A. Campi, "Automated transport service management in the future Internet: concepts and operations", Journal of Internet Services and Applications, vol. 2(2), p. 69-79, 2011.

[13] F. Callegati, A. Campi, "Network Resource Description Language," Proc. 3rd IEEE Workshop on Enabling the Future Service-Oriented Internet, co-located with IEEE Globecom 2009, Honolulu, USA, 2009.

[14] M. Caporuscio and et al., "Design and evaluation of a support service for mobile, wireless publish/subscribe applications," *IEEE Transactions on Software Engineering*, vol. 29 (12), 2003.

[15] M. Cesana et al., "C-VET the UCLA Campus Vehicular Testbed: Integration of Vanet and Mesh Networks," European Wireless Conference, Lucca, Italy, 2010.

[16] M. Gharbaoui et al., "An advanced smart management system for electric vehicle recharge," Proc. IEEE International Electric Vehicle Conference (IEVC), Greenville, USA, 2012.

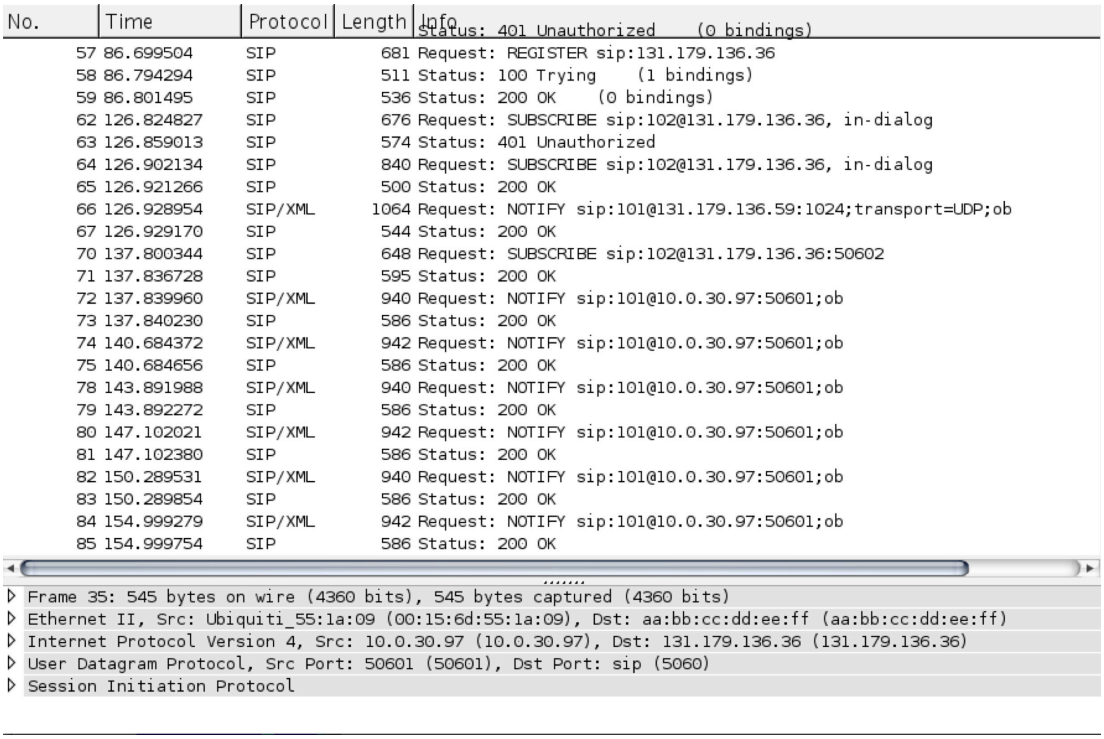

Fig. 9. A packet capture example of the exchange of SIP messages between the vehicle and the servers in the infrastructure.

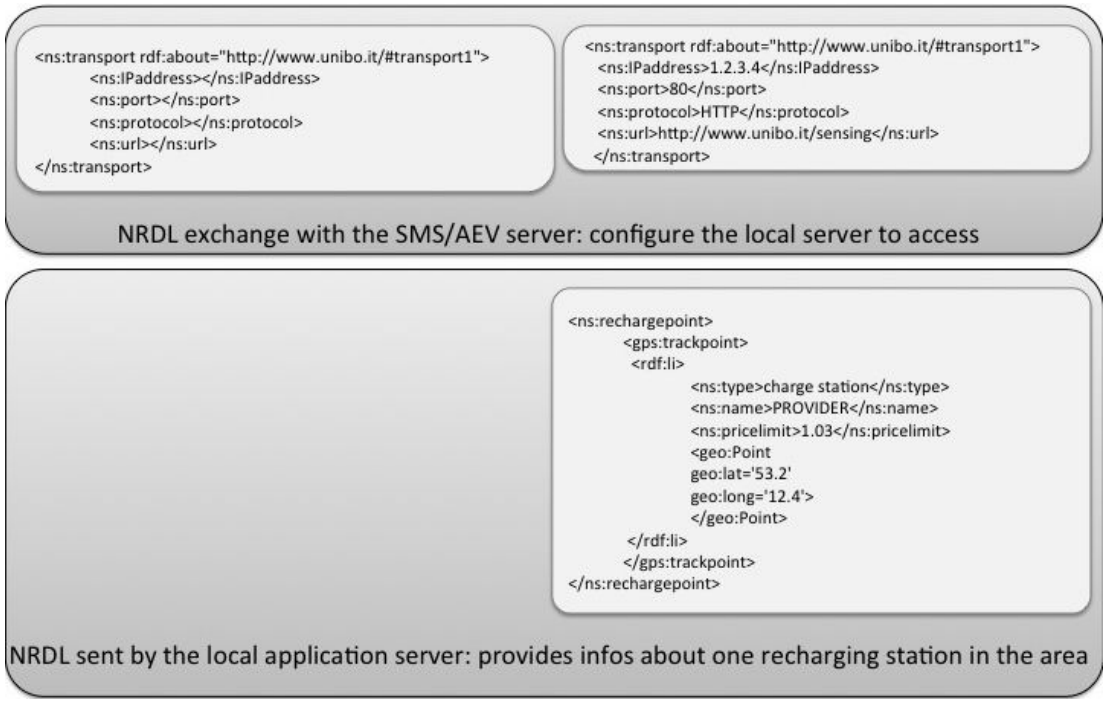

Fig. 10. Example of a simple NRDL document with data related to the application (provider and location of recharging station available in the area.