OBD-II vehicle data capture and monitoring system prototype

Claudio Aciti[1,2], Mauricio Urraco[1], and Elías Todorovich[1,3]

[1] Universidad Nacional del Centro de la Pcia. de Buenos Aires, Tandil, Argentina, caciti@exa.unicen.edu.ar
[2] Universidad Nacional de Tres de Febrero, Departamento de Ciencia y Tecnología, Caseros, Argentina
[3] Universidad FASTA, Facultad de Ingeniería, Mar del Plata, Argentina

Abstract. In this work, a prototype of a system for capturing vehicle parameters according to OBD-II technology was designed and built. The prototype is able to read different magnitudes generated by a car through an ELM-327 USB interface that allows communication with the vehicle on-board computer, and the data obtained through a USB GPS. A 3G modem and a USB Wifi adapter were added to the developed prototype. The system was implemented on a Raspberry Pi B platform with the Raspbian operating system. The 3G modem was used to transmit information collected over the Internet to a remote device or exchange. The USB WiFi adapter allowed the Raspberry Pi to be configured as an Access Point. In this way, it could be connected to a mobile device that supports the Android operating system to visualize the data read. The tests were conducted on a Chevrolet Corsa Wagon Life Gls 1.4 4p model 2009.

Keywords: OBD-II, Monitor, Raspberry Pi

1 Introduction

The on-board diagnostic system (OBD) in vehicles is an integrated system that, by means of sensors, provides the driver and technician with information about the vehicle's engine and operation [11]. The advance in the electronic technologies embedded by the manufacturers in motorized vehicles has brought a series of benefits associated with motor performance. Numerous vehicles have used electronic control systems to increase the efficiency of both the fuel injection system and the ignition system. At the same time, different ways to diagnose the problems associated with these new electronic devices have been developed, and this is how an on-board computer currently controls sensors and actuators that keep the engine running under favorable conditions. The set of actuators, sensors and diagnostic software is called On-Board Diagnostic (OBD) System [8].

The US manufacturers, government, and entities concerned about the environment began the standardization process in the 1960s.In 1970, a series of standards and requirements of gradual emissions was set for the maintenance of vehicles for prolonged periods of time. To comply with these standards, manufacturers turned to electronic controls of fuel supply and ignition systems. The sensors measured the performance of the motor and actuators adjusted the systems to provide optimum performance. In addition, these sensors could de accessed to obtain an early diagnosis of the vehicle [12]. In 1988, a standard connector and a set of diagnostic test signals were defined. This resulted in the first generation of on-board diagnostic system requirements, called OBD-I. The second version of the on-board diagnostic system, called OBD-II, is an improvement in both capacity and standardization with respect to the initial regulation of OBD. The OBD-II standard specifies the type of diagnostic connector and its pins, the available electrical signaling protocols, and the format of the messages [14,16].

In order to carry out this monitoring, a scanner is connected to the diagnostic interface and it allows access to vehicle information. There are basic and advanced scanners. The basic scanners show error codes and information stored by the vehicle, while the most advanced ones have an interface that displays the status of the sensors in real time, graphs and stores data, which greatly facilitates the diagnosis by the user. These advanced scanners have a high cost, which hinders access by mechanics and companies.

Nowadays, many companies have vehicles fleets for internal use or to provide transportation services. An information collection system allows them to control the use of vehicles to reduce investment and maintenance expenses, prevent failures and accidents, detect bad driving habits of their employees, reduce expenses associated with fuel, tires, etc. In South America, some companies provide services based on vehicle metrics. For example, Sitrack covers the so called bi-oceanic corridor and offers a fleet control system for monitoring vehicle use and drivers' actions. The system sends the information about what is happening in real time to a remote computer [5]. Another regional company based in Buenos Aires, Ful-Mar, provides a communication interface with the on-board computer that allows online reading and recording of the information captured from the vehicle in operation. [2]. Many companies develops fleet management software and hardware worldwide. For example, Fleetio [1], Plug-N-Track [3], and Titan GPS [6]. They offer devices and a cloud-based system for tracking services that provides satellite geo-localization, real-time alerts, driver behavior, vehicle health, maintenance, and reporting and analytics. Although it is a constantly growing market, there is still much to be done with this technology.

The main goal of this work was to develop a prototype of a data collection system for automobiles. This system registers what is happening in situ what happens and sends the captured information to a remote computer. In addition, an Android application was developed to view the captured data through a mobile device.

Thus, the objective was to develop a prototype that collects data from cars and is able to:

1. Monitor the on-board system on vehicles that comply with the OBD-II standard to capture engine parameters such as speed, RPM, throttle position, coolant temperature, torque, etc.
2. Obtain the geo-localization of the vehicle through satellite monitoring.
3. Connect to the Internet using telephone networks and send data remotely when the connection is available; otherwise, store the data in the internal memory until connection is available.
4. Design and implement an Android application that allows the user to connect to the data capture system to visualize the data that are being collected.

1.1 Limitations

The GPS has the limitation that in some areas it may not work, or it may work incorrectly, with considerable delays in obtaining the coordinates. A similar problem may occur with the 3G modem, because Argentina still does not have good coverage signal. Therefore, in certain areas the Internet connection may fail. In addition, the devices connected to the Raspberry Pi consume more current than the one the computer can deliver through its USB ports. For example, when the 3G modem is trying to establish the connection, it can consume peaks of up to 800mA, which can cause electrical problems that consequently end up causing a permanent failure in the Raspberry Pi. One solution to this problem is the use of a USB hub with independent power.

2 Related work

There are a number of references on OBD-II applications related to this work. For example, [7] presents a vehicle data acquisition system for the automatic management of automobile fleets using OBD, GPS, and WiFi technologies. The authors develop algorithms for decision making and suggest interesting applications for traffic control. In addition, the methods for remote diagnosis of the state of the vehicles were useful for the development of our work.

The positioning method presented in [13] corrects and replaces the inertial speed provided by a positioning system such as GPS by means of the speed value of the vehicle OBD-II system. This work makes positioning available in multiple conditions within a city. Although the work presented in our article also relates measurements from a GPS and an OBD-II system, the focus here is on monitoring.

The results of a system such as the one proposed here could serve as an input in works such as [9]. Such work proposes a method to calculate a route in order to save fuel and reduce the emissions of a vehicle. The method takes into account the driving style of the driver, the characteristics and components of the possible routes, and traffic information in real time. To evaluate the system, data sets were used, including OBD-II data of taxis with associated GPS positions. [17] also focuses on fuel savings by taking into account car driving styles. Consumption and environmental impact is obtained from OBD-II parameters. The authors develop a system that finally offers suggestions to drivers on how to save fuel. To reduce the emission of gases, in [18] they had already presented a survey of the factors that influence the emission of pollutants through experimental studies using OBD.

In [15] the objective was to monitor and analyze land vehicle fleets. This system measures speed, distance, and fuel consumption by integrating OBD-II and GPS data. Then, it transmits the information via WiFi to a remote server.

3 System development

The objective of this paper was to develop an ad-hoc system capable of communicating with the vehicle on-board OBD-II computer in order to obtain different metrics related to its operation. These data collected from the vehicle's engine are complemented with the geographical position provided by a GPS-USB and pre-processed to be sent to a remote computer. Internet connectivity is provided through a 3G modem. The functional requirements for this vehicle OBD-II data capture and monitoring system prototype are as follows:

- Setting up the connection with the OBD-II system on board.
- Sending messages and processing the obtained OBD-II answers.
- Reading a series of PIDs and decoding the answers according to the formula corresponding to each magnitude.
- Obtaining the coordinates of the vehicle position from the GPS.
- Connecting to the Internet through mobile networks.
- Storing the collected information and sending it by mobile networks to a remote computer when the Internet is available. Otherwise, the information must be stored until it can be sent.
- Providing a graphical interface through a mobile device (tablet or smarthphone) to visualize the data that is being collected.

A Raspberry Pi was used since this computer has limited hardware capacities, which allows it to be produced and sold at low prices. Its size and energy consumption are also small. A version of Linux Debian runs On this platform, called Raspbian [4], but Raspberry Pi supports other OS. These characteristics favor projects like the one presented here. We used a Raspberry Pi B, which has two USB ports and 512 MB of RAM.

Fig. 1 shows an overview of the system and the way in which the Raspberry Pi interacts with the USB devices to capture the vehicle's engine magnitudes and the GPS data, sets up and maintains the Internet connection, and uses the WiFi interface in order to become an access point. In addition, it can be seen that the microcomputer invokes a web service to send the captured data to a remote computer. In addition, a user can use an Android mobile device to visualize the captured data in real-time.

Firstly, all devices are connected to the Raspberry Pi: the ELM-327 interface [10], the USB GPS, the WiFi adapter and the 3G modem. Once the vehicle is running or with its engine on, the power cable from the Raspberry Pi is connected

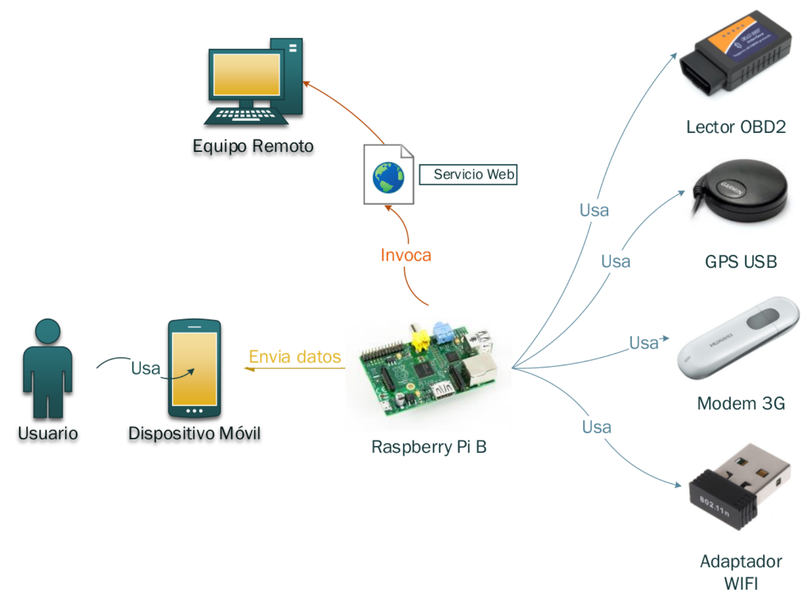


Fig. 1. System overview

to the vehicle power source. When the microcomputer has started the operating system, the data collector module, which is the main thread of the system, starts automatically. This module attempts to establish a communication with the previously mentioned ELM-327 interface to read the vehicle on-board computer. If an error occurs during this communication set-up or if the set-up fails, then the entire system will fail. On the other hand, if the connection is established successfully, then the data collector module initiates an infinite loop where the following tasks are performed:

1. The GPS reading module is implemented as a Python script. It is invoked and it obtains the geolocation and time information. If no data are received, then the system continues its execution normally but does not report any value for latitude, longitude and time.
2. The metrics of the OBD-II computer are read. If any of the readings fail, then values for that particular magnitude are not reported.
3. The data collected from the vehicle and the GPS are written in parallel to a data file and a specific TCP port.
4. Every 60 seconds the data sending module is invoked. This module is responsible for reading the data written in the file, sending it to a remote computer, and emptying the read file.

Along with this data collection process, the server module reads the data sent through the TCP port by the previous process and makes them available as a web page through another TCP port. A person with a smartphone or tablet with an Android operating system and WiFi capability can connect to the Raspberry Pi as an access point, and open the mobile application developed in this work to visualize the data.

Then, the system consists of the following modules:

1. Vehicle data collector module
2. Data sending module
3. Data server for the mobile application
4. GPS reader module
5. Mobile application

These modules are implemented in such a way that they are independent from each other to facilitate the testing and integration stages.

3.1 Data collector module

This is the central module of the system. On the one hand, it establishes communication with the on-board computer of the vehicle (OBD-II) to request the different parameters that are to be monitored. To do this, it initializes the ELM-327 embedded system using AT commands and then requests and interprets the different responses through Mode 1. In addition, it invokes the GPS reader modules and the client that sends data to a remote computer.

The ELM-327 embedded system detects and interprets nine different low-level OBD protocols. It also translates and returns the information in a unique format for any of them, allowing the independence with the internal protocol used by the vehicle. To configure the ELM-327 system, AT commands are used. Among other options, it is possible to deactivate the echo of the messages, deactivate the header in the response, or force the communication protocol to be interpreted. Below are some examples of AT commands:

- AT Z → Resets the device and shows its identification string.
  Type: at z
  Answer: ELM327 v1.5
- AT DP → Shows the current OBD vehicle protocol.
  Type: at dp
  Answer: AUTO, ISO 14230-4 (KWP FAST)

Although the OBD-II system has several modes of operation, this work focuses on mode 1 or data flow, which allows access through a PID (Parameter Identification Data) to the Engine Control Unit (ECU) analog and digital inputs and outputs. There are two sets of PIDs: the standard ones and those provided by the manufacturers. Most of the OBD-II PIDs in use are not standard, but for the most modern vehicles, have many more functions compatible with the OBD-II interface that are covered by the standard PIDs, and there is relatively less overlap between vehicle manufacturers for private PIDs. Although there are more than thirty standard PIDs, through which it is possible to read different metrics of the vehicle, in this work we consider only four to show the functioning of the system and the parameter reading method.

Speed: The PID corresponding to the speed request is 0x"0d" and the answer is one byte wide. The valid range is from 0 to 255 km/h and the formula to decode the speed is A, with A being the response value.

RPM (Revolutions per minute): The PID corresponding to the motor RPM request is 0x"0c" and the response is two bytes wide. The valid range is from 0 to 0 16,383.75 rpm and the formula for decoding the RPM is $((A*256)+B)/4$, with A and B being the response's first and second byte respectively.

Throttle position: The PID corresponding to the throttle position request is 0x"11" and the response is one byte wide. The valid range is from 0 to 100% and the formula for decoding it is $A*100/255$, with A being the response value.

Mass air flow (MAF) rate: The PID corresponding to the MAF request is 0x"10" and the response is two bytes wide. The valid range is from 0 to 655.35 grams per second (gr/s) and the formula for decoding it is $((A*256)+B)/100$, with A and B being the response's first and second byte respectively.

3.2 Mobile Application

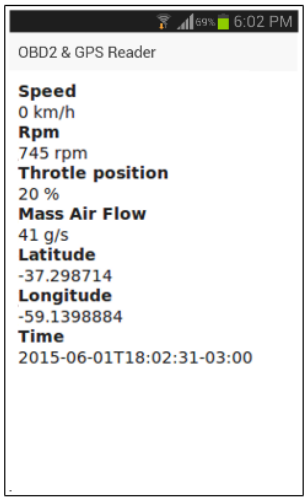
The server module ensures that the data read from the vehicle OBD-II computer and the GPS are available to any user in the vehicle with an Android mobile device with WiFi capability. The idea is that the user connects via Wi-Fi to the Raspberry Pi as an AP (Access Point). The user sets an IP address and port from the phone's browser or from an application with an embedded browser.

The Android application is basic. It has an embedded browser that is automatically routed to a TCP port assigned with the IP address of the connected Raspberry Pi. For the development of this application, a framework called Ionic-Framework is used which allows applications to be implemented for Android and iOS from HTML5 code and JavaScript code. It also supports the use of AngularJS, which facilitates the development. The application has only one screen which shows the values of the captured data: speed, rpm, throttle position, air mass flow, latitude, longitude and time, in this order.

4 Case study

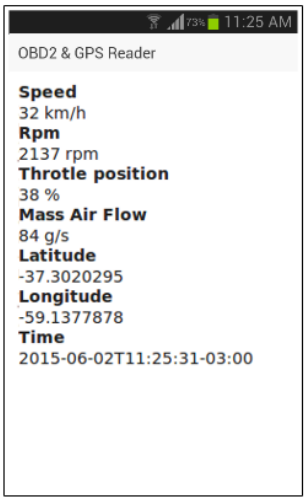
The developed system was tested on a Chevrolet Corsa Wagon Life Gls 1.4 4p 2009. Initially, the modules were tested independently, and then, the system test was performed. Several problems of an electrical nature were faced. The Raspberry Pi B can deliver up to 500mA of current through the USB ports and the 3G modem alone consumes peaks of up to 800mA. Thus, it is impossible to power the modem via the Raspberry Pi. Therefore, two solutions are possible: one of them is to use an independently powered USB hub, and the other one is to use the new computer, such as Raspberry Pi 2B, which can be configured to deliver up to 1.2A among all its USB ports. Therefore, with a source of 2A or more, it is possible for the 3G modem and the other connected devices to work properly without the need for an independent USB hub. In this work, we decided to use an additional power source.

The screenshot in Fig. 2 shows the values of the different parameters while the vehicle is stopped and when it is driven at low speed in the city. The engine is running at almost 2200 RPM, while the speed is around 30 km/h. The throttle position is a value greater than zero, but rather low, because during the driving the throttle is slightly pressed.


6:02 PM

OBD2 & GPS Reader

Speed
0 km/h
Rpm
745 rpm
Throtle position
20 %
Mass Air Flow
41 g/s
Latitude
-37.298714
Longitude
-59.1398884
Time
2015-06-01T18:02:31-03:00


11:25 AM

OBD2 & GPS Reader

Speed
32 km/h
Rpm
2137 rpm
Throtle position
38 %
Mass Air Flow
84 g/s
Latitude
-37.3020295
Longitude
-59.1377878
Time
2015-06-02T11:25:31-03:00

Fig. 2. Vehicle information when the vehicle is stopped and when it is driven at low speed

5 Conclusions and Future work

It was possible to develop and implement a prototype of a system that collects data and sends it to a remote computer through the Internet. This prototype can be extended both in hardware and software .

The prototype allows the user to monitor the on-board system in vehicles that comply with the OBD-II standard to capture different metrics. It also allows the user to obtain the geo-localization of the vehicle through satellite monitoring using a GPS.

To send the data, the system connects to the Internet using telephone networks when the connection is available; otherwise, the system stores the data in the internal memory until it is reconnected.

A mobile application was developed to allow the user to connect to the data capture system and to visualize the data that are being collected.

Particularly, in this work, the prototype must be electrically connected inside a vehicle with its engine on. Finding and adapting a reliable and sufficient power source can be a problem in the context of a vehicle. As explained above, in this work an additional independent power source was used.

While this work focuses on developing a prototype, from this basic platform, it is possible to implement extensions such as:

- improving the data visualization in the mobile application;
- capturing a greater number of metrics of the vehicle engine, at best, all standardized metrics for any vehicle that supports the OBD-II protocol;
- an independent battery that could also be used to power the prototype and thus replace the power coming from the car- this would prevent the prototype from shutting down if the vehicle is stopped or the power supply is cut off- and;
- developing a software that based on different algorithms, which analyze the captured information and infer more complex data such as car wear, average consumption, reduction of gas emissions, among others.

References

1. Fleetio. https://www.fleetio.com, accessed: 2018-06-22
2. Ful-mar sa. http://www.ful-mar.com, accessed: 2018-06-21
3. Plugntrackgps. https://www.plugntrackgps.com, accessed: 2018-06-22
4. Raspbian. https://www.raspbian.org/, accessed: 2018-06-27
5. Sitrack. https://www.sitrack.com, accessed: 2018-06-21
6. Titan gps. https://titangps.ca, accessed: 2018-06-22
7. Aljaafreh, A., Khalel, M., Al-Fraheed, I., Almarahleh, K., Al-Shwaabkeh, R., Al-Etawi, S., Shaqareen, W.: Vehicular data acquisition system for fleet management automation. In: Vehicular Electronics and Safety (IC-VES), IEEE International Conference. pp. 130–133 (20011)
8. Corvalán, R., Osses, M., Urrutia, C., Barrientos, V.: Control de emisiones de fuentes móviles, informe final para la comisión nacional del medio ambiente. Tech. rep., Gobierno de Chile, Santiago de Chile (diciembre 2000)
9. Ding, Y., Chen, C., Zhang, S., Guo, B., Yu, Z., Wang, Y.: Greenplanner: Planning personalized fuel-efficient driving routes using multi-sourced urban data. In: 2017 IEEE International Conference on Pervasive Computing and Communications (PerCom). pp. 207–216 (March 2017)
10. Elm Electronics Inc.: Elm327 datasheet. Tech. rep., Elm Electronics Inc. (2008)
11. Henderson, B., Haynes, J.: OBD-II & Electronic Engine Management Systems. Haynes Repair Manuals, Haynes Manuals N. America, Inc. (2006)
12. Landin, C.A.M., Jimenez, U.Y.F.V.: Scanner Automotriz Interfaz PC. Master's thesis, Instituto Politécnico Nacional, Mexico D.F. (2010)
13. Lim, J., Choi, K.H., Kim, L., Lee, H.K.: Land vehicle positioning in urban area by integrated gps/beidou/obd-ii/mems imu. In: 2016 IEEE International Conference on Intelligent Transportation Engineering (ICITE). pp. 176–180 (Aug 2016)

14. Lopes, J.C.O.: Diseño de un escáner automotriz OBDII Multi-protocolo. Master's thesis, Universidad de San Carlos de Guatemala (2014)
15. Malekian, R., Moloisane, N.R., Nair, L., Maharaj, B.T., Chude-Okonkwo, U.A.K.: Design and implementation of a wireless obd ii fleet management system. IEEE Sensors Journal 17(4), 1154–1164 (Feb 2017)
16. McCord, K.: Automotive Diagnostic Systems: Understanding OBD-I & OBD-II. S-A Design Workbench Series, CarTech (2011)
17. Meseguer, J.E., Toh, C.K., Calafate, C.T., Cano, J.C., Manzoni, P.: Drivingstyles: a mobile platform for driving styles and fuel consumption characterization. Journal of Communications and Networks 19(2), 162–168 (April 2017)
18. Ortenzi, F., Campbell, F., Zuccari, F., Ragona, R.: Experimental measurement of the environmental impact of a euro iv vehicle in its urban use. In: SAE Technical Paper 2007-01-0966 (2007)