An Android solution for Car Monitoring and Alerting System

Hima.M[1], Angel Rose A[2]

[1]*Master of Computer Applications, College of Engineering, Trivandrum, Kerala, India*
[2]*Assistant Professor, Master of Computer Applications, College of Engineering, Trivandrum, Kerala, India*

---***---

Abstract - *At present one of the problems faced by the drivers is that most of the sensor data in the Engine Control Unit (ECU) is not visible to the driver. The ECU data is very helpful to the driver to understand the speed, rpm, voltage, temperature, etc. There is no proper application to show these values in an understandable way. Another problem is that accidents in a rural area will take a long time for the helping hands to reach the victim. There are still problems in finding relatives of the injured person. No proper application to alert the driver if he exceeds the speed limit. Still, it will take a long time for the police to detect the accident location and to the relatives, to reach the injured person on time.*

In this paper, we are introducing an efficient, low-cost android application based on the OBD-II Bluetooth scanner, as a solution to the above problems. Here we develop an Android application for giving alert to the driver if he is over speed and also there is a facility to send SMS to the relatives when an accident is detected. This app will inform the police immediately with location data in case of an accident. So that we can ensure safe car driving and can save lives.

Key Words: On-Board Diagnostics, Electronic Control Unit, Parameter Identification, Diagnostics Trouble Codes.

1. INTRODUCTION

OBD stands for On-Board Diagnostic System. This standard was developed in 1996 in the United States of America. The OBD standard is supported by all vehicles in the market developed after the year 1996. This standard support OBD port having 16-pin.OBD scanner is connected to the port to read data from the Engine Control Unit(ECU). Many sensors placed inside the vehicle are connected with this ECU. Data send from these sensors are accumulated at the control unit. Each sensor has an identifier associated with it. This unique identifier is called Parameter Identifiers(PIDs). In addition to this, manufacture can add additional PIDs as their own. Scanning tool can request data from the ECU by sending hexadecimal representing a PID. These scanners are portable and easy to use. Another type of code we can use with OBD is Diagnostic Trouble Codes (DTCs). These codes are used for troubleshooting purpose such as malfunction detection. The 16 pin OBD port can be used to connect the OBD scanner. It can then be used for reading data from ECU. It can read data such as speed, rpm, voltage, temperature, etc from the vehicle.

In this project, we use the OBD-II Bluetooth scanner for reading data from the ECU and pass the data to android application for further processing.

In this project, we develop an android application as an interface to the user for displaying the readings from the OBDII scanner. When speed is high it gives alert to the driver and sends message to authority with the vehicle number, GPS when the car's speed is over the limit. Another module included in the project is detecting the accident condition. When the condition has occurred an immediate message regarding the crash is sent to the driver's family members and to the concerned authority.

2. RELATED WORK

Some of the literature can be found which explains Android-based smartphones that support vehicle services. Hernandez developed a prototype of an On Board system which provides communication between driver and his vehicle. This provides intelligent transport services as a result.

Chen proposed Android/OSGi platform for vehicles, used for managing or diagnosing the status of a vehicular platform from a remote area.

Al-Ani proposed an Android-based terminal which replaces the stereo system of a vehicle that can provide high fidelity In-Vehicle Infotainment System. It also provides good driver experience with the use of a mobile operating system.

Joint work of Jorge Zaldivar, Carlos T. Calafate, Juan Carlos Cano, and Pietro Manzoni also proposes an idea about detecting accident in vehicles with the use of mobile phones. This paper discusses the use of OBDII Bluetooth scanner in vehicles. They also explain an android application which can read data from the scanner and how it can be used for further processing.

3. THE OBD-II STANDARD

The standard for OBDII include the different types of diagnostic connector and its pin details. It also includes the formats for messages.OBD scanner takes its power from one of its pin connected with the battery of the vehicle. This standard provides a predefined set of Diagnostic Trouble Codes.

3.1 Types of OBD-II scanner

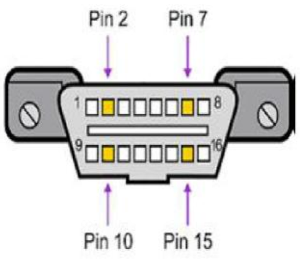
There are two categories of OBD-II scanners,

OBD-II Code readers

These are commonly low-cost devices which can be used with vehicles to read the PIDs.These PIDs are comprised of temperature, speed, rpm data. It can be displayed as real-time data in a user interface. One of the limitations of this reader device is that it does not support all type of data.

OBD-II Scan Tools

These tools are more expensive than code readers and they can offer a variety of different features. Scan tools are commonly used for advanced troubleshooting.OBD-II scanner functionality depends on whether it is a basic “code reader” or a more advanced “scan tool.” OBDII code readers are used for only read and clear codes. At the same time, scan tools are used for recording real-time data from the vehicle.



Figure(1)

Basic functionality of OBDII is its ability to read and clear codes. Scanners can also set up lists of parameter IDs (PIDs). OBDII standardization makes it relatively simple to use and they use the same connector defined by SAE J1962. Scan tools can be inserted into a universal plug into the OBD-II diagnostic connector in a vehicle.

3.2 OBD-II Protocols

An OBD2 supporting vehicle can use any of the five communication protocols: SAE J1850 PWM, SAE J1850 VPW, ISO9141-2, ISO14230-4 (KWP2000), ISO 15765-4/SAE J2480. The ELM-USB and OBD Tester can support all of them.

The CAN/OBD-II scanner uses an ELM327 chip based on SAE/ISO standards. This device can be used for the diagnostic purpose such as troubleshooting. The different modes of operation it provides are discussed below.

MODE 1: In this mode, some common values for some sensor such as speed, temperature, information about oxygen sensor are returned. Each sensor is uniquely identified by a number called PID (Parameter Identifier) used to identify the parameter. For example, speed has a PID of 12. The OBD standard includes 137 PIDs.

MODE 2: Mode 2 returns fault data at each instant. When ECM detects a fault, it records the sensor data at a specific moment.

MODE 3: Mode 3 return the stored DTC. All vehicles support the fault codes and are categorized into 4 :

P0xxx: for standard faults linked to the powertrain (engine and transmission)

C0xxx: for standard faults on the chassis

B0xxx: for standard faults on the body

U0xxx: for standard faults on the communications network.

MODE 4: In Mode 4 recorded fault codes are cleared and engine fault indicator switches off.

MODE 5: It gives a self-diagnostics result on the oxygen/lambda sensors. It is commonly used in petrol vehicles. For CAN bus ECU this mode is not used.

MODE 6: This mode used with systems not subjected to constant surveillance and gives results off self-diagnostics.

MODE 7: This fault codes which are unconfirmed. These codes are used to check that the fault code does not reappear without having to do a long test run, after a repair.

MODE 8: Mode 8 is used to return self-diagnostics on other systems and commonly used in Europe.

MODE 9: Gives information regarding the vehicle such as Vehicle Identification Number and calibration values.

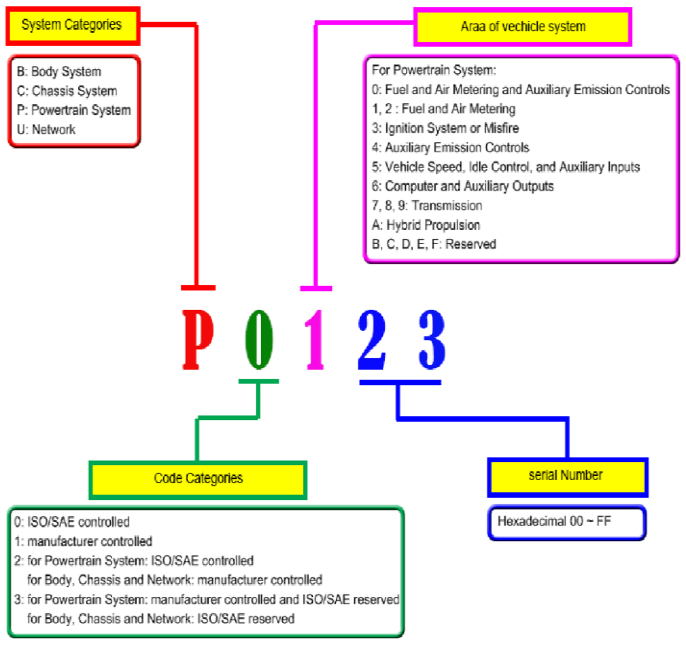
MODE 10 (OR MODE A): Mode10 is used to return the permanent fault codes.

3.3 Diagnostic Trouble Codes (DTCs)

A DTC is made up of 5 digits. The below figure illustrates the working of a DTC. With this information, it is easier to troubleshoot a DTC without knowing the description of the code. Among the five letter code representing DTC, the first one is an alphabet in English representing system malfunction. Second is a digit that represents ISO/SAE malfunction. The DTC message has five codes. In this code, the first letter is an English alphabet which is used to represent the established malfunction system. The remaining four codes are digits. The second code represents the meaning of malfunction formulated by ISO/SAE or customized by the vehicle manufacturer. The third code is the area of the vehicle system. The last two codes are used to represent the definition of the subject malfunction.

Vehicle manufacturers use the same DTC across different vehicles. As an example: General Motors vehicle and the Diagnostic Trouble Code is B0566.

- The third digit of VIN is a "1", your DTC means Infl Rest System Indicator Circuit Malf.
- The third digit is "2", the DTC means Temperature Gage Data Out of Range


System Categories
B: Body System
C: Chassis System
P: Powertrain System
U: Network
Araa of vechicle system
For Powertrain System:
0: Fuel and Air Metering and Auxiliary Emission Controls
1, 2 : Fuel and Air Metering
3: Ignition System or Misfire
4: Auxiliary Emission Controls
5: Vehicle Speed, Idle Control, and Auxiliary Inputs
6: Computer and Auxiliary Outputs
7, 8, 9: Transmission
A: Hybrid Propulsion
B, C, D, E, F: Reserved
P 0 1 2 3
Code Categories
0: ISO/SAE controlled
1: manufacturer controlled
2: for Powertrain System: ISO/SAE controlled
for Body, Chassis and Network: manufacturer controlled
3: for Powertrain System: manufacturer controlled and ISO/SAE reserved
for Body, Chassis and Network: ISO/SAE reserved
serial Number
Hexadecimal 00 ~ FF

Figure(2)

3.4 OBD-II Parameter Ids

PIDs are used to request data from vehicles. It can be used for diagnostic purpose. The SAE standard J1979 defines many OBD-II PIDs and manufacturers also define additional PIDs created especially for their vehicles. Some commonly used PIDs are,

0x04: Engine load

0x05: Engine coolant temperature

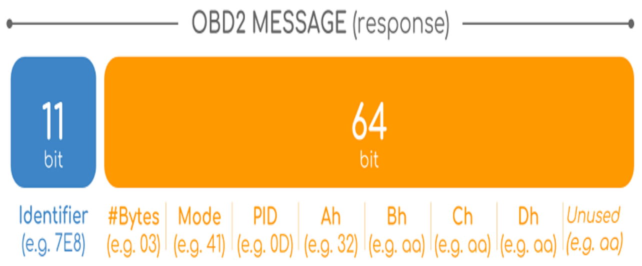
0x0C: Engine rpm

0x0D: Vehicle speed

0x10: MAF air flow rate

3.5 OBD Message Formats

Format of the message can vary depending on the protocol used. The response message has an 11-bit identifier. It also contains information regarding the mode of working, PID and the actual data from various sensors.



Figure(3)

OBD2 message consists of an identifier and data fields. Data field is split in Mode, PID and data bytes Ah, Bh, Ch, Dh. That fields are described as follows.

IDENTIFIER: Identifier is a standard 11-bit.It is used to distinguish between "request messages" and "response messages" to 7EF).

LENGTH: This field represents the length in a number of bytes of the remaining data (03 to 06).

MODE: 01-0A is used for requests and 0 is replaced by 4 for responses. SAE J1979 OBD2 standard defines 10 modes of operation.

PID: Each mode has a list of standard OBD2 PIDs .e.g. Mode 01 uses PID 0D for Vehicle Speed.

4. PROPOSED APPLICATION

The android application is used to connect with the OBDII scanner and also used to read data from it.

The OBD2 data reading comprises 3 steps,

1.connect to the OBDII scanner through Bluetooth

2. OBDII adapter is initialized with AT commands

3. Read data from the vehicle through PID codes

Basic features for the application are enabling Bluetooth, scanning for nearby Bluetooth devices, and connecting to a Bluetooth device. The Bluetooth chat allows devices to create a Bluetooth socket to communicate with another Bluetooth enabled device. Once the Bluetooth communication link was achieved, the application can receive data strings from the device. In order for the gauges and textViews to display data, we first needed a way to parse the incoming data from the Bluetooth device. The sample code below shows how we parsed the data for Intake Temperature and calculated the actual value. In order to create our gauges that will be used to display the vehicle data in real time. We created a custom gauge background with all of the parameters that we have chosen for the project with their appropriate data ranges for each. Once we had implemented the gauge dial backgrounds, the next step as they create the moving part, gauge needle. The system basically includes two components. One is the vehicle and another is the android application that can be used in connection with the car. The Android application can be used to give alert to the driver so that, we can decrease the impact of careless driving. An accident detection mechanism is also included as part of the system and emergency help can be reached to the driver by informing the authorities and to the relatives. Airbag status is taken to detect an accident condition. It can't be possible to check the airbag status in case of normal car driving.

At first mobile device is connected to an OBD-II device through a Bluetooth connection and start reading data from the ECU bus with the above-said methods. Then the speed data obtained from the OBD2 is used for giving alert to the driver if he is over speed. Another functionality is sending message to the authority with the location details of the car if the driver has overcome the limit, that the law permits. If an accident is detected based on the airbag status then SMS is sent to the relatives and to the authority.

In older days OBD-II connectors are based on serial connection such as RS-232. Now it has changed to a variety of mediums such as Bluetooth, WiFi, etc. These changes to technology have a dramatic evolution in the development of new applications.

To use Bluetooth features in the application, it is necessary to add Bluetooth permissions in the application's manifest file. They are BLUETOOTH, BLUETOOTH_ADMIN. Another permission need is ACCESS_COARSE_LOCATION ,which is used to access the location of the user.

```
BlutoothAdapter bluetoothAdapter = BluetoothAdapter.
                                    GetDefaultAdapter();
if (bluetoothAdapter == null) {
  // Device doesn't support Bluetooth
}
```

BluetoothAdapter is required for all bluetooth activity.For getting it call getDefaultAdapter() method. If it returns null ,your device does not support Bluetooth. For enabling bluetooth call the isEnabled() method.

```
if (!bluetoothAdapter.isEnabled()) {
Intent enableBtIntent = new Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE); startActivityForResult(enableBtIntent, REQUEST_ENABLE_BT); }
```

Then BluetoothServerSocket is created for connecting the two Bluetooth devices. Start listening for connection requests by calling accept() method. Initialize the BluetoothChatService to perform Bluetooth connections.

```
mChatService = new BluetoothService(this, mHandler);
```

The Handler is used to receive information back from the BluetoothChatService.

```
private final Handler mHandler = new Handler() {
```

Message can be handled in the following way,

```
  public void handleMessage(Message msg) {
    switch (msg.what) {
    case MESSAGE_STATE_CHANGE:
                .
                .
                .
     case MESSAGE_READ:
          compileMessage( msg.obj.toString());
```

To compile the message, msg.substring(0, 4) is checked with PIDs 0C,0D,05 etc and (msg.substring(4, msg.length()),16) is converted to suitable forms as a value to the above said PIDs.

The general workflow of the application is illustrated in the below figure:

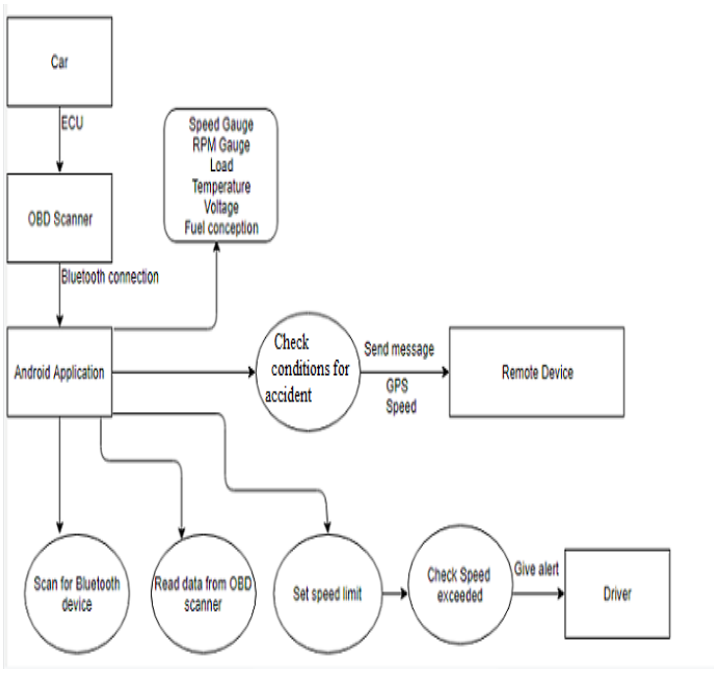


Figure (4)

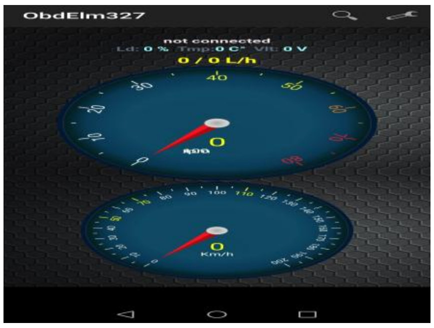
5. IMPLEMENTATION

By developing an android application we can communicate with the OBD scanner and can request data from it. The received data can be used for further processing and can be displayed to the user in a convenient format. Scanner here we use is a Bluetooth device ELM327. We need GPS and SMS functionality of our phone for the proper working of the application. In case of an accident detection, the application sends SMS with the location data to two mobile numbers.

First, scanning for Bluetooth devices occurs. Then a list of available devices is shown. If we select any one of them pairing code will be asked. By default, the code is 1234. Then the application is connected with OBD Bluetooth scanner. The data coming from OBD is shown at the interface. Speed and RPM are shown in gauges. The Speed limit is cross-checked at 3-sec intervals to avoid inappropriate alerting. Accident condition is checked with the airbag triggering condition and SMS manager is used to send SMS.

5.1 Graphical user interface (GUI)

At the main window, the user is informed with the speed, RPM, Voltage, mileage, Temperature, Fuel consumption, etc. The speed and RPM are shown in gauges. On top of the window, there are two buttons, one is for the activity to select the OBD scanner devices from the list. That activity also has a scan for device functionality. There are two buttons at the MainActivity. One is for Settings and another one for resetting the system. In the settings menu, we have another window to change the Engine displacement, Engine type, Face color. The Application looks like as follows.



Figure(5)

6. CONCLUSION

In this proposed system, an android application is developed for car monitoring purpose.It also include functionalities such as giving alert to the driver in case of over speed, sending message to authorities if he violates the speed limit and also sends message to relatives in case of an accident.The system uses Bluetooth connectivity with the OBDII device. Speed is checked with three-second intervals, for the better working results.The future work will include additional features such as real-time accident detection with the central server and automatic emergency helping integrated with ambulance service. A database for storing the data from OBD-II, for analyzing the behavior of driver will also be included in the future studies.

REFERENCES

[1] Jorge Zaldivar, Carlos T. Calafate, Juan Carlos Cano, Pietro Manzoni, "*Providing Accident Detection in Vehicular Networks Through OBD-II Devices and Android-based Smartphones*," in 5th IEEE Workshop On User MObility and VEhicular Networks On-Move 2011, Bonn.

[2] Mi-JinKim, Jong-Wook Jang, Yun-Sik Yu," *A Study on In-Vehicle Diagnosis System using OBD- Ⅱ with Navigation*", in IJCSNS International Journal of Computer Science and Network Security, VOL.10 No.9, September 2010

[3] https://en.wikipedia.org/wiki/On-board_diagnostics

[4] https://en.wikipedia.org/wiki/ELM327

[5] https://lembergsolutions.com/blog/how-guide-obdii-reader-app-development