symmetry

MDPI

Project Report

A Study on Big Data Thinking of the Internet of Things-Based Smart-Connected Car in Conjunction with Controller Area Network Bus and 4G-Long Term Evolution

Donghwoon Kwon [1], Suwoo Park [2] and Jeong-Tak Ryu [3,*]

[1] Department of Computer Science, Texas A&M University-Commerce, Commerce, TX 75428, USA; donghwoon.kwon@tamuc.edu
[2] Rovitek Inc., Gyeongsan-si, Gyeongsangbuk-do 38479, Korea; swpark@rovitek.com
[3] School of Electronic and Communication Engineering, Daegu University, Gyeongsan 38453, Korea
* Correspondence: jryu@daegu.ac.kr; Tel.: +82-53-850-6635

Academic Editor: Ka Lok Man
Received: 19 May 2017; Accepted: 2 August 2017; Published: 9 August 2017

Abstract: A smart connected car in conjunction with the Internet of Things (IoT) is an emerging topic. The fundamental concept of the smart connected car is connectivity, and such connectivity can be provided by three aspects, such as Vehicle-to-Vehicle (V2V), Vehicle-to-Infrastructure (V2I), and Vehicle-to-Everything (V2X). To meet the aspects of V2V and V2I connectivity, we developed modules in accordance with international standards with respect to On-Board Diagnostics II (OBDII) and 4G Long Term Evolution (4G-LTE) to obtain and transmit vehicle information. We also developed software to visually check information provided by our modules. Information related to a user's driving, which is transmitted to a cloud-based Distributed File System (DFS), was then analyzed for the purpose of big data analysis to provide information on driving habits to users. Yet, since this work is an ongoing research project, we focus on proposing an idea of system architecture and design in terms of big data analysis. Therefore, our contributions through this work are as follows: (1) Develop modules based on Controller Area Network (CAN) bus, OBDII, and 4G-LTE; (2) Develop software to check vehicle information on a PC; (3) Implement a database related to vehicle diagnostic codes; (4) Propose system architecture and design for big data analysis.

Keywords: IoT; smart car; V2V; V2I; V2X; CAN bus; big data

1. Introduction

According to Klynveld Peat Marwick Goerdeler (KPMG)'s Global Automotive Executing Survey 2016 [1], automotive trends are rapidly changing every year. For instance, connectivity and digitalization were ranked tenth in 2015, but were ranked first in 2016. This has a significant meaning due to the fact that vehicles may transform to mobile data rooms, which can lead to virtual product features and services [1]. One of the well-known technologies regarding this trend is the Tesla autopilot feature which is composed of autosteer, autopark, driver assistance visualization, etc. [2]. Additionally, Tesla vehicles regularly receive over the air software updates.

All these emerging technologies are core and key for smart connected cars, and Figure 1 below depicts the full range of connected car technologies and services [3].

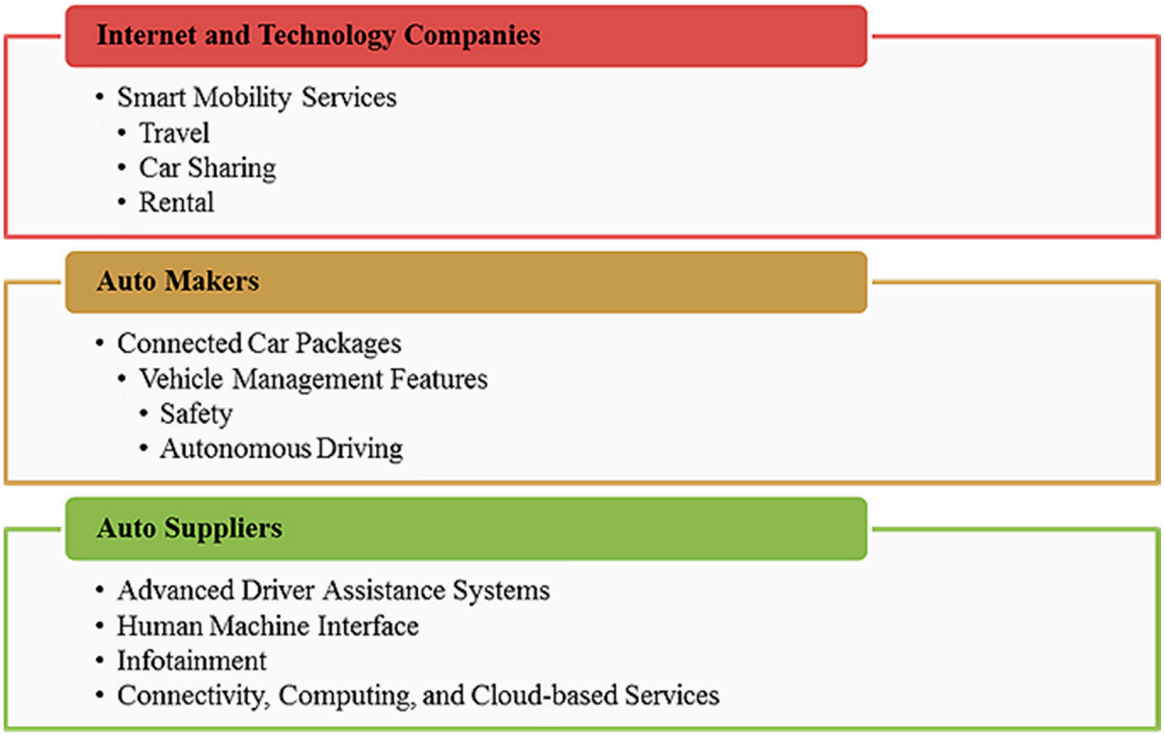


Figure 1. The range of connected car technologies and services.

Among connected car packages provided by auto makers, the vehicle management feature is selected for further discussion. Additionally, the vehicle management feature in terms of the smart connected car in this paper means the collection and transmission of the information via wireless communications. However, before discussing all the details, the first and most important thing that readers should understand is a fundamental definition and concept of the smart connected car. The most significant criterion to be a smart connected car is connectivity. This connectivity can be provided by a third-party system (smartphone) or the vehicle's own transmitter/receiver unit [4]. In this work, however, both connectivity ideas are employed for the system architecture. The first connectivity type refers to Internet of Things (IoT). The main concept of IoT is that objects, sensors, and everyday items generate, exchange, and consume data through network connectivity and computing capability [5]. The second connectivity type results from Controller Area Network (CAN) bus communication developed by Bosch (Gerlingen, Germany) [6]. Since most modern vehicles are composed of a huge number of electronic sensors, which communicate through CAN bus, this point exactly matches up with the definition of the second connectivity type. Thus, the IoT and CAN bus-based system architecture and actual module development will be proposed in this paper.

This paper is organized as follows: Section 2 concentrates on describing relative technologies and communications in terms of CAN bus, Vehicle-to-Vehicle (V2V), Vehicle-to-Infrastructure (V2I), Vehicle-to-Everything (V2X), and connected cars; Section 3 proposes research directions and a system architecture; Section 4 introduces newly developed CAN bus and data transmission modules with testing results, database (DB) design, and implementation, as well as proposing the design of a mobile application and cloud-based Distributed File System (DFS); Section 5 summarizes the research work and describes research limitations and our future work.

2. Related Work

2.1. Controller Area Network Bus Network

As briefly described in the previous section, CAN bus was developed by Bosch in the early 1980s due to the needs of many automotive makers [6]. The main objective of early CAN bus was to define a standard for network communication between sensors, actuators, controllers, and other nodes in real-time applications and to minimize redundant wiring [6,7]. Figure 2 below shows the general idea of CAN bus.

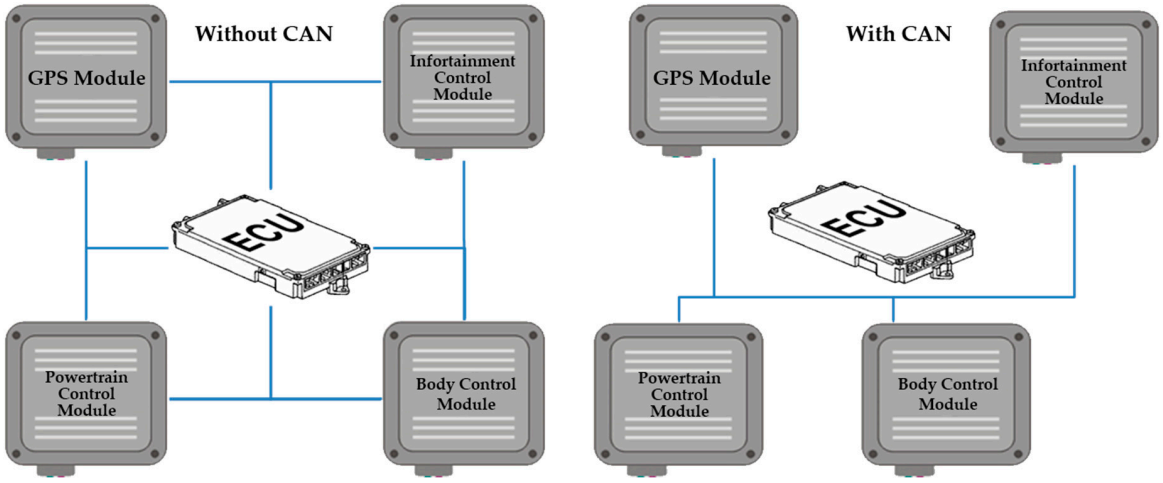


Figure 2. General idea of Controller Area Network (CAN) bus. GPS: Global Positioning System.

One of the well-known technologies regarding this in modern vehicles is embedded systems, also known as Electronic Control Units (ECUs), which consist of more than seventy microprocessors communicating over networks to control a vehicle infotainment system, powertrain, etc. [8]. However, microprocessors cannot directly communicate by themselves so a standard protocol has to be defined for transmitting and receiving data packets [6,8]. The protocol refers to the Open System Interconnection (OSI) seven layers' model consisting of a physical layer, data link layer, network layer, etc., and the CAN protocol standardizes two lower layers, which are the physical and data link layer [6,8]. In addition, the targeted applications in automobiles using CAN bus communication are an anti-lock brake system, driving assistant system, engine control, etc. [9]. Note that there are various communication protocols used in the different categories of modern vehicles such as the Local Interconnect Network (LIN) for the body, FlexRay for X-by-wire applications, Bluetooth, etc. [10]. The reason for focusing on CAN bus in this paper is due to its speed and cost-effectiveness compared to other protocols.

Next, which CAN is used for collecting vehicle information needs to be examined. As mentioned earlier, a number of ECUs are connected on CAN bus, and two important concepts can be extracted. The first extracted concept is a CAN bus topology in which multiple ECUs communicate to each other through CAN bus [11]. The second extracted concept is that there are two different types of CAN which are connected through the gateway shown in Figure 3 below, i.e., high speed CAN and low-speed CAN [10].

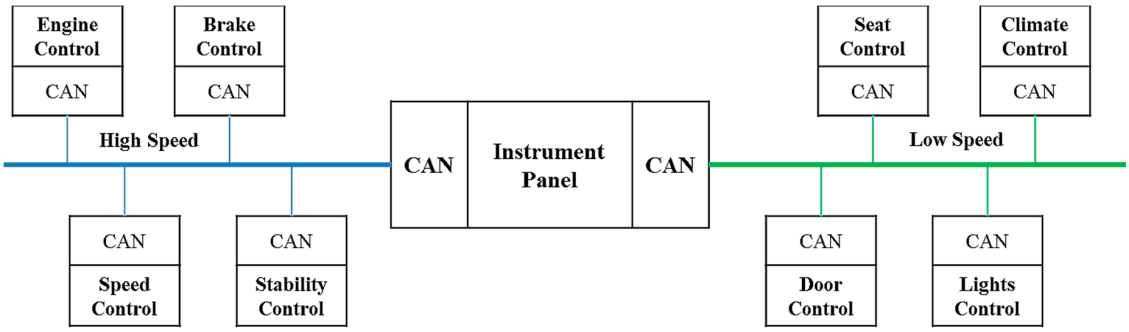


Figure 3. High speed and low speed CAN buses.

Two CAN types are categorized by the protocol features and international standards [12]. Class A is a low speed network which is used for sensors or executive management, and its speed rate is less than 10 KB per second [12]. As shown in Figure 3 above, seat control, door control, and lights control are the main applications of class A. Class B has a medium speed rate, and climate control is the main application of this class. The speed rate of class C is between 12 KB and 1 MB per second or faster than this rate so that high speed CAN is considered from this class [12]. Engine control, brake control, speed control, and stability control refer to powertrain management, and those are main applications of class C [12]. In addition, International Organization for Standardization (ISO) 11898, Society of Automotive Engineers (SAE) J2284, SAE J1939, etc., are relative standards for high speed CAN bus communication [12]. Class D has a more than 2 MB per second speed rate, and most infotainment systems are in this class [12]. The main goal of this research is to collect and transmit information such as the vehicle speed, Revolution per Minute (RPM), brake status, etc., so that class C is adopted in this work.

2.2. The Overview of Vehicle-To-Vehicle

A V2V communication technology, known as the connected vehicle safety model and Vehicular Ad-hoc Networks (VANETs), was introduced by the U.S Department of Transportation's (DOT) National Highway Traffic Safety Administration (NHTSA) in 2014 [13,14]. The primary idea of this technology is for the safety improvement of the light vehicles by enabling vehicles to share information or messages, e.g., the vehicle position, motion, size, etc., with each other through V2V protocols, e.g., Dedicated Short Range Communication (DSRC), on-board computers with sensors, and network interface cards such as Institute of Electrical and Electronics Engineers (IEEE) 802.11p, Bluetooth, etc. [13,14]. Note that safety improvement is associated with Forward Collision Warning (FCW), Blind Spot Warning (BSW), Lane Departure Warning (LDW), etc.

The in-vehicle components of the V2V system proposed by the U.S DOT NHTSA are the DSRC radio, Global Positioning System (GPS), memory, safety application ECU, driver-vehicle interface, and a vehicle's internal communication networks [13]. In addition to in-vehicle components, the V2V system can be expanded to the V2I applications based on Road Side Equipment (RSE), which may use other communications such as existing 3G and 4G-LTE cellular networks and Wi-Fi [13,14]. The overall concept of V2I will be explained in the following section.

2.3. The Overview of Vehicle-To-Infrastructure

The V2I technology is mainly focused on an Intelligent Transportation System (ITS). One of the well-known examples in terms of the ITS is an Active Traffic and Demand Management (ATDM) system which was launched by the Washington State Department of Transportation (WSDOT) in August, 2010 [15]. Note that the V2I technology is also based on V2V communications. However, since IEEE 802.11p and DSRC in V2V communications are the most suitable for short-range connectivity, heterogeneous wireless technologies such as 3G, 4G Long Term Evolution (4G-LTE), IEEE 802.11,

and IEEE 802.16e for long-range connectivity can be used for effective V2I communications [14]. For this reason, wireless access points, known as Road-Side Units (RSUs), are the key components of V2I communications [14].

2.4. The Overview of Vehicle-To-Everything

The overall concept of V2X is to communicate between a vehicle and all other entities including pedestrians, mobile devices, etc. Since V2X covers a wide range of aspects of the smart-connected car, the limitation of short-range connectivity results from IEEE 802.11p and DSRC motivates to explore new communication technologies. To the best of our knowledge, the fastest wireless network is LTE-Advanced (LTE-A), and 5G networks are an actively on-going research area [16,17]. Furthermore, 5G is expected to bring new capabilities to the smart-connected car due to extreme broadband, ultra-low latency, and edgeless connectivity [18]. The overall concept of V2V, V2I, and V2X is depicted in Figure 4 below.

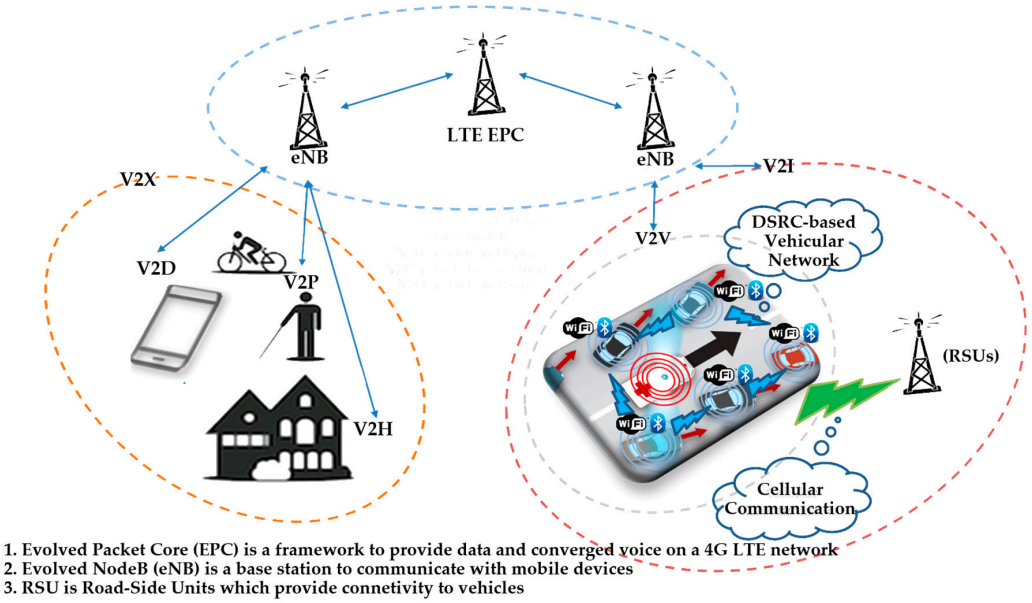


Figure 4. The overall architecture of Vehicle-to-Vehicle (V2V), Vehicle-to-Infrastructure (V2I), and Vehicle-to-Everything (V2X). LTE: Long Term Evolution; EPC: Evolved Packet Core; eNB: Evolved Node B; RSUs: Road-Side Units; V2D: Vehicle-to-Device; V2P: Vehicle-to-Pedestrian; V2H: Vehicle-to-Home.

2.5. The Overview of Connected Cars

Numerous studies have proposed an idea for connected cars related to our research. In [19], a Hadoop-based big data solution is proposed for processing vehicle data such as RPM, speed, GPS location, etc., which are obtained by a Bluetooth On-Board Diagnostics (OBD) scan tool. The actual android mobile application is developed and this application enables users to see diagnostic information on the cloud. In [20], the authors propose system implementation in conjunction with European On-Board Diagnostics (EOBD), GPS, a General Packet Radio Service (GPRS), and Universal Mobile Telecommunications System (UMTS) to assess the risk associated with vehicle usage as part of a Pay As You Drive (PAYD) insurance program used for the assessment of insurance premiums.

3. The Proposed Research Directions and System Architecture

According to the literature review in the previous section, the following research directions are determined.

1. Although powertrain-related vehicle information, which is used for driving information, is a primary source for big data analysis, both B-CAN and C-CAN are adopted.
2. DB, which is composed of a driver table, vehicle table, and diagnostics table, needs to be implemented.
3. The diagnostics table stores each automaker's diagnostic codes so that drivers are able to check vehicle malfunctions via a mobile application.
4. DB should be designed by normalization so that data redundancy can be avoided.
5. Cloud-based DFS implementation is required for big data analysis.
6. Since the IoT concept in this research refers to the V2I concept, which is also based on the V2V concept, short-range and long-range connectivity based on Bluetooth and 4G-LTE are employed as the main communication networks.

Based on the defined research directions, the following system architecture is established, as shown in Figure 5.

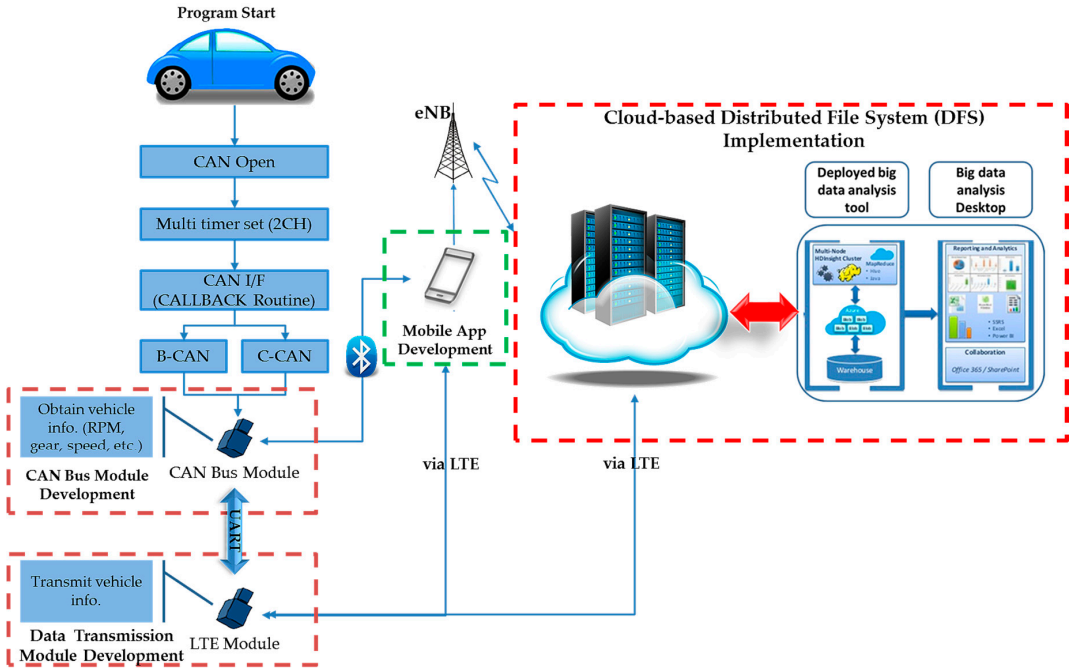


Figure 5. The overall system architecture. 2 CH: 2 Channel; CAN I/F: CAN Interface; RPM: Revolution per Minute; UART: Universal Asynchronous Receiver/Transmitter.

According to our proposed system architecture shown in Figure 5 above, there are three main components which need to be developed. The first component is the CAN bus module, through which the vehicle information mostly related to the powertrain can be obtained. However, this CAN bus module has hidden features which can remotely control a vehicle through a mobile application. Since these hidden features are out of the research scope, we will not mention them in the rest of the paper. Yet, once information is obtained by the CAN bus module, such information can be transmitted to a mobile device through Bluetooth and is checked by software on the PC. The second main component is to develop a data transmission module. This data transmission module is developed based on 4G-LTE. Data transmission heads to two targets such as a mobile device and server, but its primary destination is the server. The third main component is a mobile application targeted at two things: (1) to check vehicle malfunction codes by interacting with the server; (2) to transmit the obtained information to a cloud-based DFS via wireless communications. The last expected main component is to implement the cloud-based DFS.

4. Proof of Actual Development and Proposing System Design

4.1. Modules and Software Development

To achieve the first and sixth research directions described in the previous section, OBDII and Bluetooth were applied to develop the CAN bus module. It is important to note that OBDII is internationally standardized by ISO 15765-4, SAE J1939, ISO 9141-2, ISO 14230-4, SAE J1850 PWM, and J1853 VPW. The circuit and PCB board design included the function of the ELM327 chipset, which is a programmed microcontroller developed by ELM electronics (London, ON, Canada). Note that the ELM327 protocol is the PC-to-OBD interface standard. Figure 6 depicts a blueprint of the module.

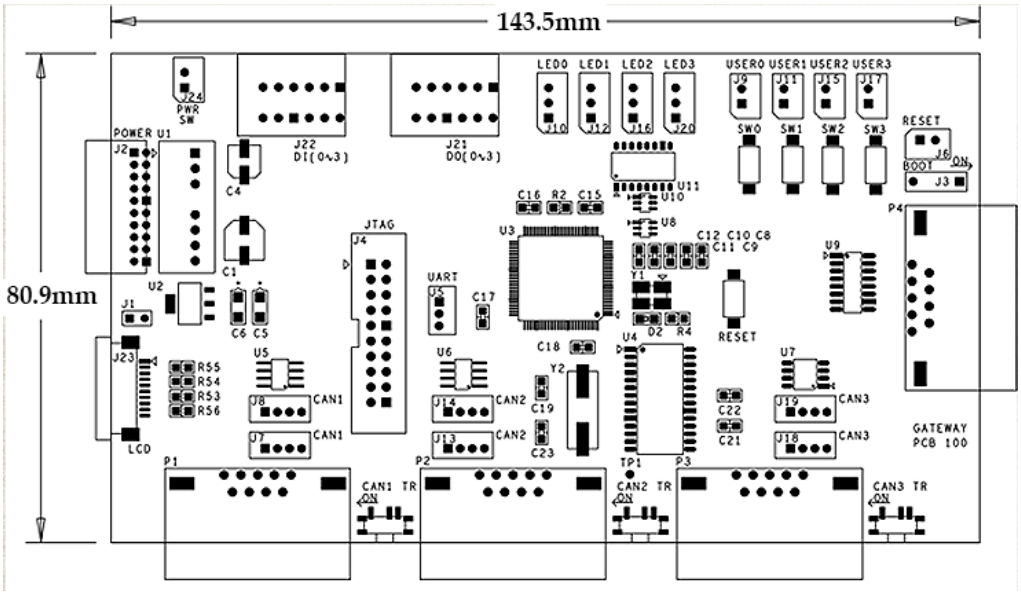
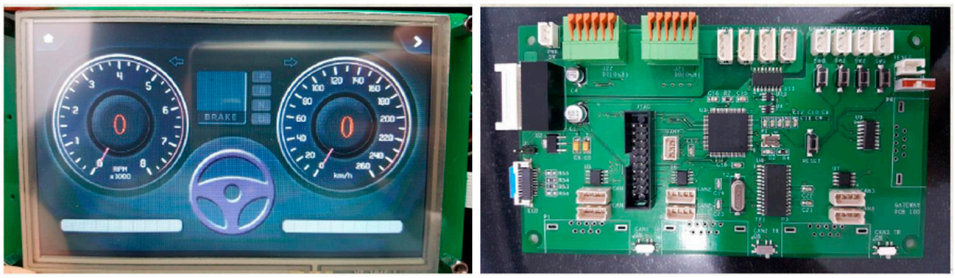


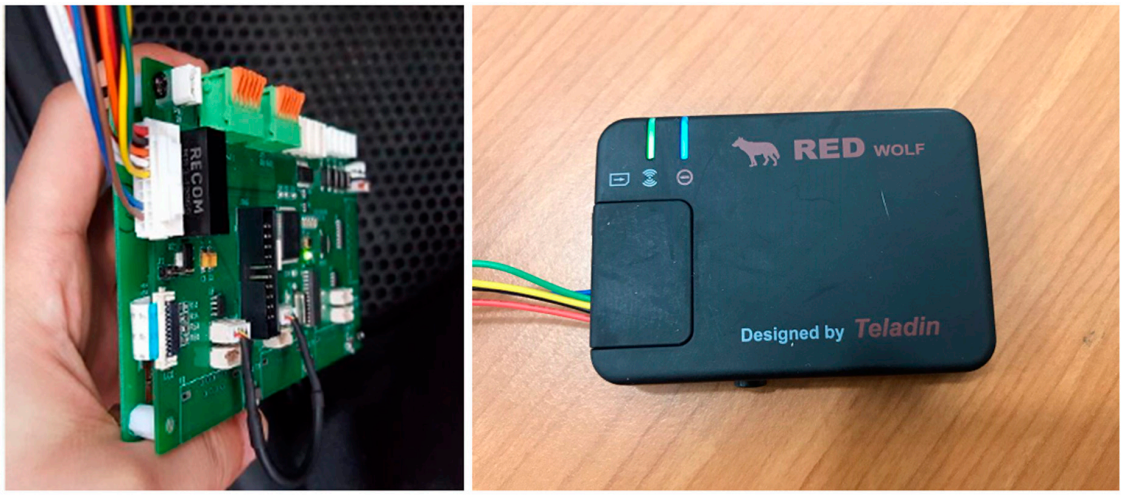
Figure 6. Blueprint of the module.

To communicate between the CAN bus module and data transmission module, a Universal Asynchronous Receiver/Transmitter (UART) is used. Communication between the data transmission module and server is performed through 4G-LTE. Additionally, vehicle information, e.g., RPM, speed, gear, brake status, etc., can be visually checked by the developed software shown in Figure 7a. Development of the CAN bus module and PC software is based on C, IAR Embedded Workbench for ARM (IAREWARM) (IAR Systems, Uppsala, Sweden), and Visual C++ using MS Visual Studio 2010 (Microsoft Corporation, Redmond, WA, USA). Actual testing was conducted, and we confirmed that all the information was successfully received through our developed modules. Figures 7 and 8 show our actual deliverables.



(a) (b)

Figure 7. (**a**) Software to check vehicle information (**b**) Actual developed CAN bus module.



(a) (b)

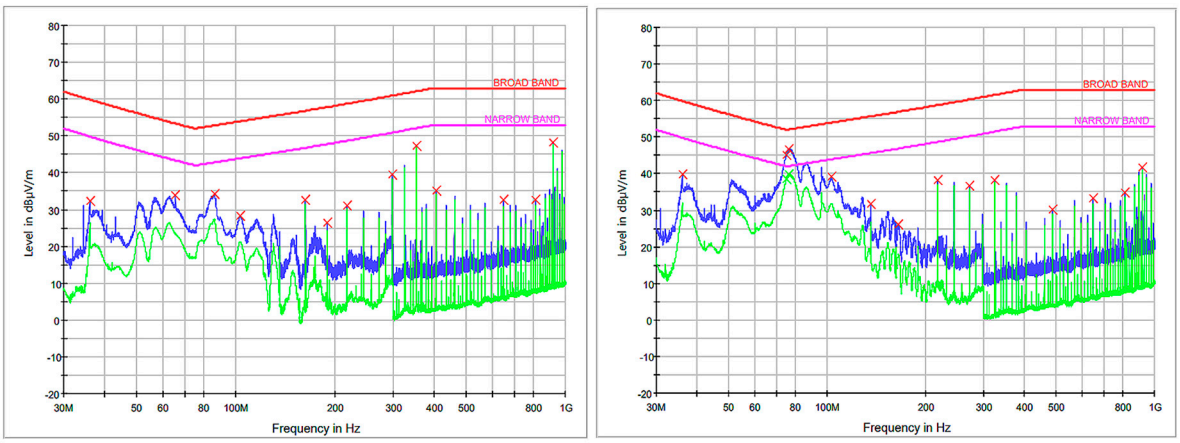
Figure 8. (**a**) CAN Bus module for testing (**b**) Actual data transmission module.

4.2. System TestingResults

We also officially requested system testing and an evaluation to see whether our developed modules met the criteria of both broadband and narrowband radiated disturbances in South Korea. Testing was conducted in the thirteen specific frequency ranges (30~50, 50~75, 75~100, 100~130, 130~165, 165~200, 200~250, 250~320, 320~400, 400~520, 520~660, 660~820, and 820~1000 MHz) and we confirmed that our modules fulfilled the testing criteria of broadband and narrowband radiated disturbances. Table 1 describes the testing criteria and Figure 9 illustrates the testing results.

Table 1. Testing criteria for broadband and narrowband radiated disturbances.

Category	Frequency (MHz)		
	30~75	**75~400**	**400~1000**
Broadband (dBμV/m)	62–25.13 log(f/30)	52 + 15.13 log(f/75)	63
Narrowband (dBμV/m)	52–25.13 log(f/30)	42 + 15.13 log(f/75)	53



(**a**) (**b**)

Figure 9. (**a**) Radiated disturbance result based on the peak detector mode only (**b**) Radiated disturbance result based on the peak and average detector modes.

4.3. Database Implementation

DB was implemented based on Microsoft Structure Query Language (MS-SQL), but as mentioned in the second research direction, our initial relational DB was designed by three DB tables such as the driver, vehicle, and diagnostics shown in Figure 10 below.

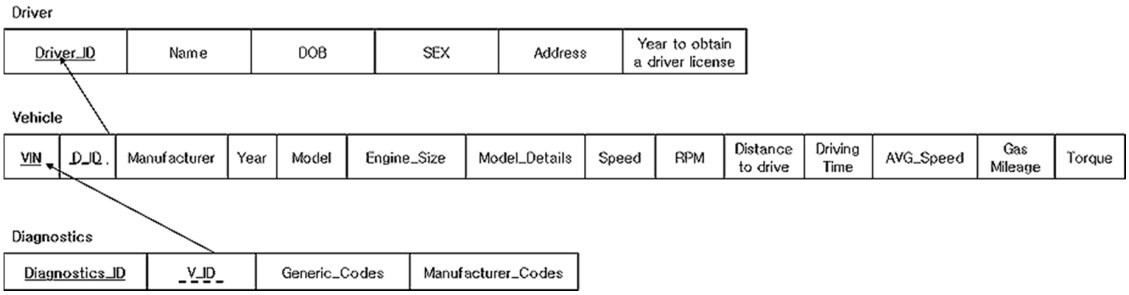


Figure 10. Relational database (DB) design.

The most significant problem of the designed relational DB is that it has a high chance to cause data redundancy. Due to such an issue, we realized the necessity to revise the relational DB with a consideration of the normalization concept so that we applied 1NF, 2NF, and 3NF to the relational DB. To do so, we achieved the forth research direction, and the normalized DB is shown in Figure 11 below. Note that we used a linear underline and dotted underline for a primary key and foreign keys, respectively.

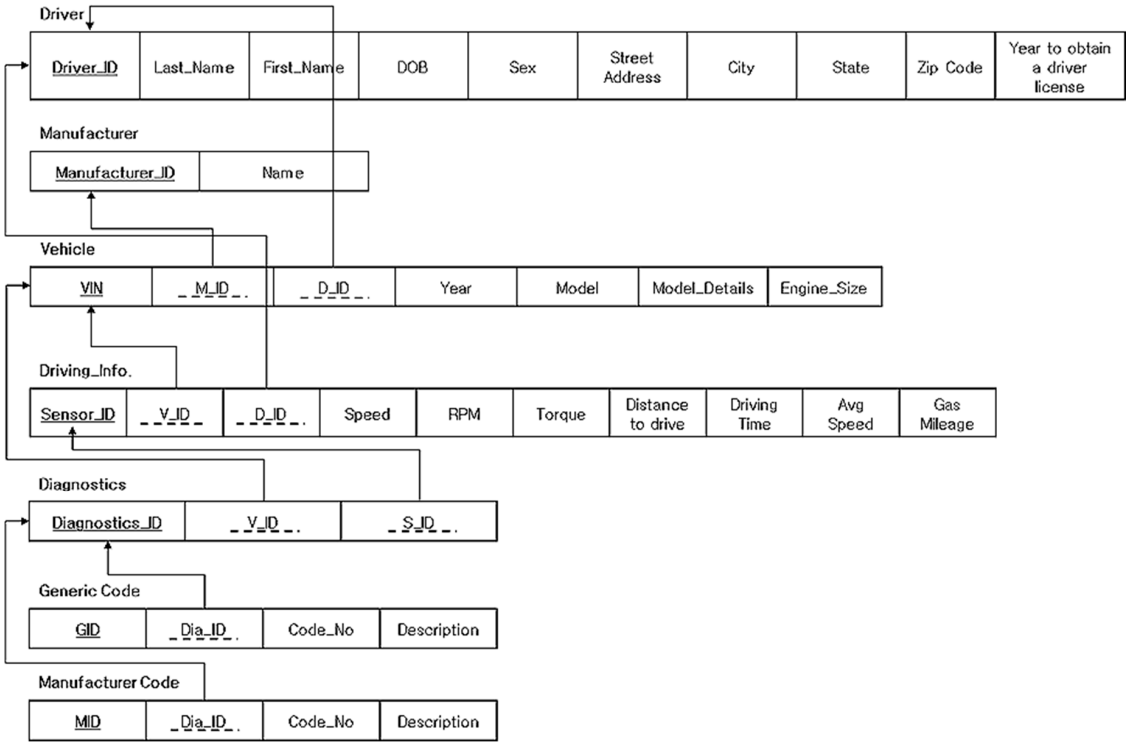


Figure 11. The normalized DB. DOB: Date of Birth.

Since we used the normalized DB, the vehicle powertrain-related information from now on belongs to the driving information table (Driving_Info. table). Drivers or users need to register their personal information, and such information is stored in the driver table. Yet, five categories of data (i.e., manufacturer name, vehicle year, model, model details, and engine size) need to be predefined

by a DB administrator. By doing so, users are able to easily select their vehicle information using a dropdown menu. The DB administrator is also required to define diagnostic codes based on generic codes and manufacturer codes. This is for achieving the third research direction. Moreover, diagnostic codes are classified into two types. For instance, if diagnostic codes start with P0XXXX, P0 means powertrain-related generic codes. If diagnostic codes start with P1XXXX, P1 means powertrain-related manufacturer codes. Note that there are four alphabets to indicate which part has a malfunction, i.e., B for body, C for chassis, P for powertrain, and U for user network. Since there are a huge amount of diagnostic data related to generic and manufacturer codes, we initially imported a total of 773 powertrain-related generic codes and 1433 manufacturer codes. The targeted automakers for manufacturer codes were Audi (Ingolstadt, Germany), Mercedes-Benz (Stuttgart, Germany), BMW (Munich, Germany), Hyundai (Seoul, Korea), Kia (Seoul, Korea), Chevrolet (Detroit, MI, USA), and Honda (Tokyo, Japan) because those automakers have the most market shares in Korea and United States. We are still working on importing the diagnostic codes for other manufacturers' codes and generic codes.

4.4. Mobile Application

From this section, we mainly focus on proposing system design because we have not developed actual deliverables. The intended mobile application is composed of seven functionalities, i.e., dashboard, diagnostics, graphing, restore, diagnostics logs, preferences, and diagnostics search. To design the mobile application, we referred to a variety of Android applications on the Play Google app (Google, Moutain View, CA, USA) market so that the name of the seven functionalities may change. The purpose of the mobile application is to check vehicle diagnostic codes and driving information in real-time. The planned development platform is OpenXC because it supports open source hardware and software.

The dashboard functionality is intended to check the vehicle speed, RPM, accelerating status, braking status, etc., in real-time. The diagnostics functionality enables users to check a vehicle malfunction in real-time by interacting with the server, and diagnostic results are automatically stored in diagnostic logs. In addition, the diagnostic search functionality helps users find diagnostic codes from the server. The graphing functionality is used for grasping driving habits so that the real-time gas mileage, average gas mileage, fuel consumption, etc., are provided to users. The restore functionality is employed for data backup, and the preferences functionality is related to the application settings. A mock-up of the application interfaces using the Balsamiq mockups (Balsamiq Studios, Sacramento, CA, USA) tool is depicted in Figure 12.

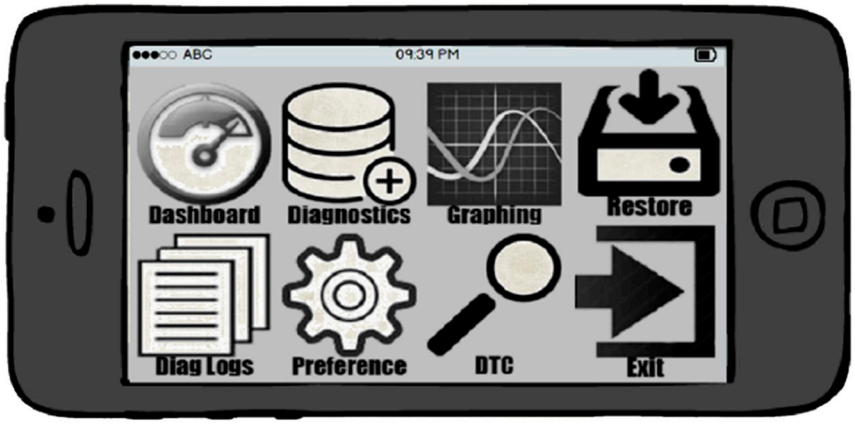


Figure 12. Application mock-up.

4.5. Proposing A Cloud-Based Distributed File System for Big Data Analysis

We have not started cloud-based DFS implementation for big data analysis, but our initial considerations are as follows:

- The main purpose is to share the information of diagnostics and driving.
- A variety of DFS' need to be examined.
- Protocols to share and store the information of diagnostics and driving are web service-based.
- The collected information goes through preprocessing, and a software framework needs to be employed for data processing, analysis, and reporting.
- A big data analysis tool using R, Python, Matlab, or other tools needs to be deployed [21–23].
- The analyzed result is provided to users through data visualization.

After our considerations were revealed, we realized one issue regarding the implemented SQL-based DB. Our initial plan was that the mobile application interacts with the server in accordance with the user's requests to obtain information on diagnostics and driving. Figure 13 depicts the flow of the diagnostics and graphing functionality using the sequence diagram.

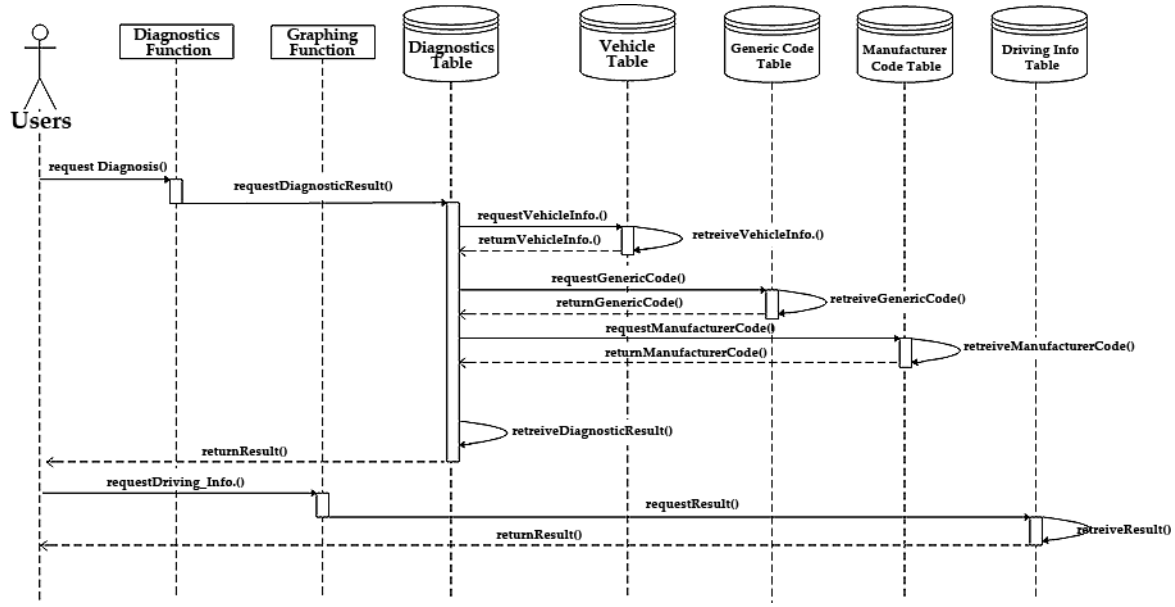


Figure 13. Sequence diagram of the diagnostics and graphing functionality.

The most significant issue with respect to our implemented DB is that DB design and implementation are against the requirements of the cloud-based DFS such as scalability, integrity, velocity, etc., so that we start to examine various DFSs for big data analysis.

4.5.1. Google File System

Google uses a self-developed file system (GFS) to provide a cloud service [24]. GFS is Linux-based and is composed of a single master and multiple chunk servers. The single master manages the metadata of the file system and control system operation. Chunk servers process the input and output of a client and store actual data.

4.5.2. Hadoop Distributed File System

Hadoop DFS (HDFS) was developed based on the GFS and is open source-based [25]. The HDFS architecture is very similar to the GFS because a single namenode plays a similar role to the single master, and datanodes are considered as chunk servers.

4.5.3. Amazon Web Service

The Amazon Web Service (AWS) is a cloud storage service and composed of an Elastic file System (EFS), Elastic Block Store (EBS), and Simple Storage Service (S3) [26]. The EFS provides the storage running on Amazon Compute Cloud (Amazon EC2) instances in the cloud environment. The EBS provides two types of block storage volumes, i.e., the Solid State Drivers (SSD) volume and Hard Disk Drivers (HDD) volume, in the AWS cloud. The S3 is an object storage in which it is possible to store and search data through the web service interface.

4.5.4. Microsoft Azure

Microsoft provides the Azure cloud platform service based on Infrastructure as a Service (IaaS) and Platform as a Service (PaaS) [27]. The Windows Azure platform consists of Windows Azure, SQL Azure, and Windows Azure platform AppFabric. The most remarkable difference between Microsoft Azure and Amazon EC2 is a cloud stack; in other words, while Microsoft Azure provides both IaaS and Paas, Amazon EC2 provides only PaaS. One interesting technology of Microsoft Azure is to support NoSQL. NoSQL, in contrast with the traditional Relational Database Management System (RDBMS), does not need to join database tables and simplifies data access.

4.6. Proposing Distributed File System Design

According to our examination of the DFS for big data analysis, Microsoft Azure would be the strongest DFS candidate for our system due to scalability and compatibility, and we are proposing the DFS design shown in Figure 14.

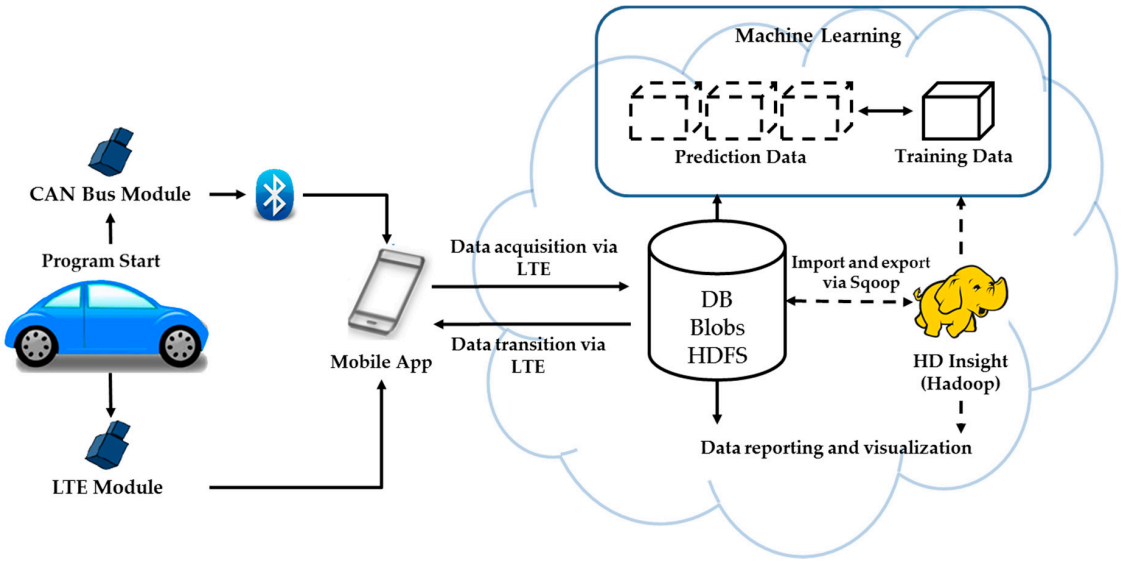


Figure 14. Overview of Distributed File System (DFS) design.

Once vehicle information is transmitted to and stored in the cloud server via LTE, the acquired data is loaded from the HDFS to Azure HD Insight via Sqoop for the purpose of data analysis. Note that Sqoop is a tool used to import and export data between a DB and HD Insight. There are two objectives of data analysis. The first objective is to provide data reporting and visualization in terms of driving habits, diagnostic logs, etc. The second objective, which is optional, is for prediction models through training data. For example, once enough data is collected to build a training model, we apply a machine learning technique to provide prediction models for giving maintenance events in advance, giving the best route information for saving gas, etc.

5. Conclusions

Governments and automakers in the world are actively seeking innovative technologies for V2V, V2I, and V2X connectivity, and the fundamental technology to fulfill such connectivity is IoT. This new technical paradigm motivated us to explore new technologies regarding the smart-connected car. In this paper, we successfully developed actual modules based on CAN bus and 4G-LTE to obtain and transmit vehicle powertrain information. The information obtained by modules was also visually checked by our developed software. However, although we implemented SQL-based relational DB for vehicle diagnostics and driving information, we realized that our implemented DB needs to be revised due to the conflict with the characteristics of NoSQL. This is a great asset for lessons learned and will be a part of our future work. We are still conducting this research, and successful cloud-based DFS implementation, as well as big data analysis, which will be the next main research topic in our future work.

Acknowledgments: This research was supported and funded by the Kyungpook Software Convergence Cluster Global Stage R&D Support Program (GB-RD2-03).

Author Contributions: Donghwoon Kwon conducted the literature review, established the overall research system architecture, implemented the database, designed the mobile application, conducted research about the DFS, and wrote the paper. Suwoo Park performed the circuit and PCB board design and developed the software and modules. Jeong-Tak Ryu guided the research direction.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. KPMG. KPMG's Global Automotive Executive Survey. 2016. Available online: https://assets.kpmg.com/content/dam/kpmg/xx/pdf/2017/01/global-automotive-executive-survey-2017.pdf (accessed on 9 November 2016).
2. Tesla. Model S Software Release Notes v7.1. 2016. Available online: https://www.tesla.com/sites/default/files/pdfs/release_notes/tesla_model_s_software_7_1.pdf (accessed on 9 November 2016).
3. Baker, E.; Crusius, D.; Fischer, M.; Gerling, W.; Gnanaserakan, K.; Kerstan, H.; Kuhnert, F.; Kusber, J.; Mohs, J.; Schulte, M.; et al. Connected Car Report 2016: Opportunities, Risk, and Turmoil on the Road to Autonomous Vehicles. 2016. Available online: http://www.strategyand.pwc.com/reports/connected-car-2016-study (accessed on 8 December 2016).
4. Mercedes-Benz Technology. Trend Analysis: Connected Car. 2015. Available online: https://www.mbtech-group.com/fileadmin/media/pdf/consulting/downloads/Trendanalyse_Vernetztes_Fahrzeug_2015_EN.pdf (accessed on 24 February 2017).
5. Karen, R.; Eldridge, S.; Chapin, L. Internet Society. The Internet of Things: An Overview. The Internet Society (ISOC). 2015. Available online: https://pdfs.semanticscholar.org/6d12/bda69e8fcbbf1e9a10471b54e57b15cb07f6.pdf (accessed on 14 March 2017).
6. Johansson, K.H.; Törngren, M.; Nielsen, L. Vehicle Applications of Controller Area Network. In *Handbook of Networked and Embedded Control Systems*, 1st ed.; Hristu-Varsakelis, D., Levine, W.S., Eds.; Birkhäuser Basel: Cambridge, MA, USA, 2005; pp. 741–765.
7. Currie, R. Development in Car Hacking. SANS Institute. 2016. Available online: https://www.sans.org/reading-room/whitepapers/ICS/developments-car-hacking-36607 (accessed on 14 March 2017).
8. Cook, J.A.; Freudenberg, J.S. Controller Area Network (CAN). 2007. Available online: http://www.eecs.umich.edu/courses/eecs461/doc/CAN_notes.pdf (accessed on 14 March 2017).
9. Wolf, M.; Weimerskirch, A.; Paar, C. Security in Automotive Bus Systems. In Workshop on Embedded Security in Cars. 2004. Available online: http://www.weika.eu/papers/WolfEtAl_SecureBus.pdf (accessed on 15 March 2017).
10. Suwatthikul, J. *Fault Detection and Diagnosis for In-Vehicle Networks*, 1st ed.; Zhang, W., Ed.; InTech: Rijeka, Croatia, 2010; pp. 283–306.
11. Ueda, H.; Kurachi, R.; Takada, H.; Mizutani, T. Security Authentication System for In-Vehicle Network. SEI Technical Review. 2015. Available online: http://global-sei.com/technology/tr/bn81/pdf/81-01.pdf (accessed on 16 March 2017).

12. Wang, Y.; He, L. The Application of Microcontroller MC9S08DZ60 in Automotive CAN Bus Electronic Control Unit. *Open Electr. Electron. Eng. J.* **2013**, *7*, 110–115. [CrossRef]
13. Harding, J.; Power, G.; Yoon, R.; Fikentscher, J.; Doyle, C.; Sade, D.; Lukuc, M.; Simons, J.; Wang, J. Vehicle-to-Vehicle Communications: Readiness of V2V Technology for Application. 2014. Available online: https://www.nhtsa.gov/sites/nhtsa.dot.gov/files/readiness-of-v2v-technology-for-application-812014.pdf (accessed on 16 March 2017).
14. Vegni, A.M.; Little, T.D.C. Hybrid vehicular communications based on V2V-V2I protocol switching. *Int. J. Veh. Inf. Commun. Syst.* **2011**, *2*, 1–18. [CrossRef]
15. Auer, A.; Feese, S.; Lockwood, S. History of Intelligent Transportation Systems. 2016. Available online: https://ntl.bts.gov/lib/59000/59200/59263/download1.pdf (accessed on 17 March 2017).
16. Araniti, G.; Campolo, C.; Condoluci, M.; Lera, A.; Molinaro, A. LTE for Vehicular Networking: A Survey. *IEEE Commun. Mag.* **2013**, *2*, 148–157. [CrossRef]
17. Le, N.T.; Hossain, M.A.; Islam, A.; Kim, D.; Choi, Y.; Jang, Y.M. Survey of Promising Technologies for 5G Networks. *Mob. Inf. Syst.* **2016**, *2016*, 1–25. [CrossRef]
18. Qualcomm Technologies Inc. Leading the World to 5G: Cellular Vehicle-To-Everything (C-V2X) Technologies. 2016. Available online: http://2pe5rtjld2w41m0dy17n5an1.wpengine.netdna-cdn.com/wp-content/uploads/2016/10/Qualcomm-Cellular-V2X-AESIN-Oct-2016.pdf (accessed on 17 March 2017).
19. Nkenyereye, L.; Jang, J. Addressing Big Data Solution enabled Connected Vehicle Services using Hadoop. *J. Korea Inst. Inf. Commun. Eng.* **2015**, *19*, 607–612. [CrossRef]
20. Boquete, L.; Rodríguez-Ascariz, J.; Barea, R.; Cantos, J.; Miguel-Jiménez, J.; Ortega, S. Data Acquisition, Analysis and Transmission Platform for a Pay-As-You-Drive System. *Sensors* **2010**, *10*, 5395–5408. [CrossRef] [PubMed]
21. The R Foundation. Available online: https://www.r-project.org/ (accessed on 8 August 2017).
22. The MathWorks Inc. Available online: https://www.mathworks.com/products/matlab.html (accessed on 8 August 2017).
23. Python Software Foundation. Available online: https://www.python.org/ (accessed on 8 August 2017).
24. Ghemawat, S.; Gobioff, H.; Leung, S. The Google file system. In Proceedings of the 19th ACM Symposium on Operating Systems Principles, Bolton Langding, NY, USA, 11–22 October 2003; pp. 29–43.
25. Borthakur, D. HDFS Architecture Guide. 2008. Available online: https://hadoop.apache.org/docs/r1.2.1/hdfs_design.html (accessed on 17 March 2017).
26. AWS Storage Services Overview. 2015. Available online: https://d0.awsstatic.com/whitepapers/AWS%20Storage%20Services%20Whitepaper-v9.pdf (accessed on 17 March 2017).
27. Chappell, D. Introducing the Windows Azure Platform. 2010. Available online: http://www.davidchappell.com/writing/white_papers/Introducing_the_Windows_Azure_Platform,_v1.4--Chappell.pdf (accessed on 17 March 2017).