UNIVERSITY OF NAIROBI

School of Computing and Informatics

AN EMBEDDED FRAMEWORK FOR INTEGRATING SPEED, ALCOHOL AND WEIGHT MONITORING IN A PUBLIC SERVICE VEHICLE

By

Samue Karuga Mungai

P52/64696/2013

Supervisor

Dr Eng Lawrence Muchemi

A research project submitted in partial fulfilment of requirements for the award of Master of Science degree in Computational Intelligence, School of Computing and Informatics, University of Nairobi

DECLARATION

This research project is my original work and has not been presented for a degree or any other award in any other university.

Student Signature: .... **Date:** ....

Name: Samuel Karuga Mungai **Registration Number: P52/64696/2013**

This project report has been submitted as a partial fulfilment of the requirements for award of the degree of Master of Science in Computational Intelligence with my approval as the University supervisor

Supervisor Signature:........................ **Date:** ....

Name: Dr Eng Lawrence Muchemi

DEDICATION

This work is dedicated to my mother, Monica Wangari Karuga, whose constant support enabled me to pursue my education, my father Hiram Mungai Kiarie, who passed on the love of reading and respect for education and has been instrumental in my life by providing a pillar that propelled us as a family. I also dedicate this work to my brother Stanley Kiarie, and my close friend Elizabeth Wakonyo. The two of them were a force behind this achievement, especially when balancing between education, work and family became unbearable. I also dedicate this work to my grandmother Lydia Wambui Karuga. This old lady made me believe in the power of possibilities especially if you take control of your life. It is amazing that her life has been a case of you get what you want in life.

ACKNOWLEDGEMENT

I wish to convey my gratitude to my distinguished supervisor Dr Eng Lawrence Muchemi for his true leadership which was outstanding throughout the project. He provided detailed guidance and in-depth understanding of the research subject in the course of preparing for and conducting the research. I am grateful for the helpful comments he provided throughout the project period.

Special thanks to officers at Kenya National Highway Authority, Kenya Bureau of Standards, Dalcom Enterprises, Pinnacle Systems Limited , Power Gorvenors Limited, Saferider Management Systems, Papo Hapo Solutions, Control Tech Limited, Flex Communications Limited, Central Vehicle , Simba Corporation,Markmann & Company Limited, Control Tech Limited, Paymatic , Lab Technologies Innovations Limited, Comat Trading Co. Limited, Mvita Uto Diesel Systems,Cbet Matatu Owners Sacco Society Limited, Central Vehicle Services Limited Marts Logistices Limited and other persons who responded to the research questions, fed me with the much needed information and also gave the required project data.

For the panel Members, Dr E.T Opiyo, Dr A.N Wausi and Dr Eng Lawrence Muchemi, this work would not have been completed were it not for your constructive criticism during and after defence. This provided useful inputs which were incorporated in improving and polishing up the research work.

Lastly, special thanks to my family and friends for their belief that it was indeed possible to finish the project. They also provided fresh impetus every time and this kept me going. Indeed, this achievement was a team effort.

ABSTRACT

Road Carnage is a great problem in our Kenyan roads. To reduce this, the government have tried to introduce regulatory laws for drivers on the roads. For instance, all public service vehicles except taxi cabs and commercial vehicles exceeding 3.048 tonnes must be fitted with a well-functioning and inspected Speed Limiter. Heavy vehicles are also required to record their weights in weigh bridges in an attempt to reduce overloading. To curb drunk driving, traffic officers' takes impromptu alcohol level tests on roads that are notorious for drunk driving.

This research identified a problem with the introduction of these laws. There was no integration framework that could be used to monitor these 3 main causes of road accidents (Over speeding, drunk driving and overloading). The research then tried to come up with an integration framework that can do the monitoring

To solve the main research problem of lack of an integration framework, the research worked with 3 clear objectives. First, a suitable integration framework was designed. Secondly, a prototype based on the chosen framework was developed. Lastly, some data was simulated in order to test and evaluate the prototype

Through previous research works, this research project identified that the framework for on-board Network Level Diagnostics (Suwatthikul, 2008) could be used with a few modifications. This framework constitutes of 3 main parts; Network Interface Module (NIM), Diagnostics Module and a Fault and Knowledge database. For this research, NIM was used to collect data from Speed Limiters, Weigh Bridges and Alco blows. Diagnostics module was replaced by a rule based control system that acted as the central control. The fault database was used to record all the instances collected, while the Knowledge database was omitted.

After designing the framework, a prototype was developed on the Arduino framework. The Speed Limiter and a Weigh bridge data was generated by varying a potentiometer. To capture alcohol levels, an inbuilt Alco blow was developed. Since speed limiters store data in files, the values got from the potentiometer that was mimicking a speed limiter were saved in a file. Different formats were used in order to cater for various formats used by Speed limiter vendors.

Data was then simulated by varying the potentiometers and alcohol levels entering the device. This data was saved to a permanent log file for use in future .To verify that the device was working correctly, some limits were set and verifications were made to confirm that the device was correctly checking if those limits were exceeded or not. An example of a limit set used was Speed=80 km/h, weight= 1000kg and Alcohol Content=30 Micrograms.

The results of this tests showed that the device correctly identified when the limits were exceeded or not. This was verified from the stored logs and also a third party system that was a Ticketing/Monitoring system.

This project showed that this device could be used on the roads. With proper implementation it would be more effective to curb some of the social problems on the Kenyan roads like bribes, some culprits avoiding apprehension and many more. This technology would be better that using distinct devices since it monitors all the 3 main parameters contributing to roads accidents together.

This project was an achievement since it solved the problem that was currently there of lack of an integration framework for Speed, alcohol and Weights monitoring. Integration of these 3 factors in one device is therefore possible. This was the new knowledge that the project provided

This research project was similar to previous works on Hardware Integration frameworks. In all of them, a suitable framework was chosen, based on previous research, a hardware device was then developed , based on the proposed framework and lastly, the device was tested to verify if the research problem had been solved. It also discussed some of the issues in integration frameworks, notably, data formatting, data integration, system redundancy, data integrity and reliability.

Keywords- PSV, Speed Governors, Alco Blow, Weigh Bridge, On-board network level Diagnostic, Embedded device, Embedded system .

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

PSV ----------------------------------Public Service Vehicle

RTA ----------------------------------Road Traffic Accidents

BAL ----------------------------------Blood Alcohol Level

UAV ---------------------------------Unmanned Aerial Vehicles

WHO --------------------------------World Health Organisation

NTSA ---------------------------------National Transport and Safety Authority

EAC----------------------------------East African Countries

KeNHA ------------------------------Kenya National Highway Authority

HSWIM ------------------------------High Speed Weighing in Motion

NGSG -------------------------------New Generation Speed Governor

KEBS---------------------------------Kenya Bureau of Standards

ECU---------------------------------Electronic Control Unit

FD-----------------------------------Fault Diagnosis

FDI----------------------------------Fault Detection and Isolation

NIM---------------------------------Network Interface Module

DM ---------------------------------Diagnostics Module

Table of Contents

1. INTRODUCTION

1.1 Background

Road carnage is a great problem in Kenya and all over the World. The deaths and injuries caused by it are shockingly high. According to World Health Organization (WHO), Road Traffic Accidents (RTA) causes an estimate 13 million deaths and 20-50 million injuries in the world annually. World Report on Road Traffic Injury Prevention 2007 (WHO, 2007) , Global Status Report on Road Safety 2013 (WHO, 2013), Road traffic injuries.Fact sheet N°358 March 2013 (WHO, 2014), World Report on Road Traffic Injuries Prevention, Glossary of Terms (WHO, 2014), World Report on Road Traffic Injury Prevention (WHO, 2012), (Nantulya V.M, 2009), (Razzak J.A, 1998). In Kenya, PSV vehicles, especially Matatu, busses and cargo trucks, are the ones mostly involved in road accidents (Odero, 2003) hence this research targets them.

The main causes of road carnage in Kenya are over speeding, drunk driving and overloading of vehicles (Rosnek, 2013), (Police, 2007). Out of necessity, the Kenyan Government was forced to introduce laws that were to help regulate those three main causes. Even if these laws have been facing a lot of drawbacks, a lot, but not enough, have been gained by their introduction.

1.1.1 Over speeding

To help curb the issue of over speeding, the government introduced speed governors to all public service vehicles except taxi cabs and commercial vehicles over 3048 kg. These speed governors were intended to limit the speed of those vehicles to 80km/h. (Manyara, 2006) Even if this idea was a noble one; it did not take long for the PSV drivers to get ways to over speed without being noticed. These drivers had mechanical switches that they would switch off if they wanted to disconnect the governor and hence they were able to over speed unnoticed. In presence of police officers, all they had to do was to switch the governors on again. (Manyara, 2006) If apprehended, these drivers easily bribed their way out if the traffic officers involved were corrupt (Manyara, 2006)

This then led to introduction of the new generation speed governors. The National Transport and Safety Authority Board gave a legal notice that PSV vehicle owners had to ensure that their vehicles were fitted with digital speed governors in order to prevent human indiscretion and promote riddance of corruption in the roads (Nation, 2014). The main feature of these new generation speed limiters was recording the speed of the vehicles as it moves. The Benchmark 2013 (KEBS, 2013)

1.1.2 Drunk Driving

To curb the problem of drunk driving, the government introduced the use of breathalyzers. To implement this, the traffic officers had to pick spots that they thought were popular to drunk drivers, like near night clubs, and then motorists had to take Blood Alcohol Level (BAL) tests when they were passing these road blocks. (Manyara, 2006)

It did not take long before corrupt traffic officers started extorting bribes from drunken driving culprits, and hence most of these drivers went unpunished. (Manyara, 2006) These devices were also unhygienic. The mouth pieces were attached to the devices hence germs could be transmitted from one user to the other as they blow. Due to these two main fall-backs, the use of these Alco blows was suspended. (Manyara, 2006)

In November 2013, Alco Blows were reintroduced again. (News24, 2014) During their reintroduction, Transport Cabinet Secretary, Michael Kamau, stated that the version of the Alco blows that were to be used were better since they were modern, hi-tech and hygienic. The light, portable devices were using more accurate and up to date technology to determine the BAL. Their mouthpieces were also disposable, meaning each driver would use his/her mouthpiece, which would be disposed after use. (News24, 2014) These breathalyzers were be calibrated between 0-100 millilitres.

1.1.2 Overloading

Long distance tracks takes majority of the blame when it comes to accidents caused by overloading. (Manyara, 2006) The reason for this is that the truck is too heavy and it has more momentum making it more difficult to stop and more likely to get out of control when swerving to avoid an accident. Overloading also causes other problems with the truck such as brake failure or inability to steer. (chaikinandsherman, 2014) In 2009, in a meeting attended by Roads Assistant Minister Wilfred Machage, Permanent Secretary Michael Kamau, Public Works Assistant Minister Lee Kinyanjui, Transport Assistant Minister Harun Mwau, Director General

Kenya National Highway Authority Meshack Kidenda and his Kenya Rural Roads Authority counterpart Mwangi Maingi, it was noted that the more the trucks were loaded in excess of the 34 tonnes required by the Kenyan law, the more the roads were being damaged. (Media, Curbing Trucks Overloading, 2014) Damaged roads are another cause of RTA in Kenya today (Manyara, 2006)

In 2012, all EAC member states agreed on a prohibition of exceeding 56 tonnes of total loaded weights and a maximum of 7 axle vehicles (iwacu, 2014) , (ttcanc, 2014). In Kenya, this directive is regulated under The Traffic Act Chapter 403 and the EAC Vehicle Control Bill 2012. (iwacu, 2014)

Even if the two laws were to effectively regulate overloading in trucks, they have not been so effective due to widespread corruption and weak weighbridge management. (Daily, 2014)

According to research, conducted by CPCS Transcom Limited for the Northern Corridor Transit Transport Co-ordination Authority, 50-55 % of trucks plying Kenyan roads are still overloaded, some with cargo up to 60 tonnes. (Limited C. T., 2010)

In September 2013, KeNHA announced the introduction of new weighbridge technology. The centerpiece of the innovations is the High Speed Weighing in Motion (HSWIM) system, technology that measures axle weights as a vehicle travels along the road. This would in turn speed up freight, save taxpayers millions of dollars in road repairs and reduce the opportunities for parasitic corruption. Only those trucks suspected of being overweight would be diverted into the weighbridge station for static weighing. (Star, 2014)

The first bridge to use HSWIM was Mariakani Bridge. Currently, Athi River, Gilgil, Mtwapa, Isinya, Juja, Busia, Rongo and Webuye Bridges are also using this technology. (Star, 2014)

1.1.3 Variety Of Devices

With the laws regulating speed, drunk driving and overloading, there have been an emergence of different brands of devices that regulates them. However, all those brands must meet the laid standards in the Kenyan law. For instance, the speed limiters should be logging the speeds, the breathalyzers should be calibrated between 0-100 millilitres and the weigh bridges should be able to weigh 56 tonnes and above. The Benchmark 2013 (KEBS, 2013). Different brands play a major role when it comes to speed limiters. By 1st April 2014, there were 22 main manufacturers of speed limiters in Kenya, hence 22 different brands of devices, as reported by KeNHA. Approved Speed Governors and Vendors (KeNHA, 2014). A list of these entire devices is attached in the end of this document.

1.1.4 Possible Scientific Gap

A survey of available literature revealed that, over speeding, drunk driving and overloading are the main causes of deaths in Kenya (Nantulya V.M, 2009), (Manyara, 2006). The devises used to monitor these three vices worked in isolation and did not give harmonized reports.

From a computational perspective, there still lacked a proper integration framework with inbuilt intelligence able to monitor those three main factors contributing to road carnage. That envisaged integration framework would come up with a way that those three factors would be monitored at the same time and provide intelligent reports.

The research also noted that the integration framework could not do away with the current speed limiters and weighbridges since they had cost PSV owners and the government a lot in terms of finances and time to implement them. For instance, to install a speed limiter gadget in September 2014 cost vehicle owners an average of 30,000 shillings (Nation, 2014). Due to this , the framework was supposed to make it possible to interface and intergrate many main stream devices available in the local market , through a simplified plug-in framework..

1.2 Statement of the problem

The main research problem that this research project tried to solve was lack of an integration framework for speed, alcohol and weight monitoring.

This research project found that speed monitoring was done via speed limiters, alcohol monitoring done via Alco blows and weight monitoring via weighbridges. Clearly, those three factors were monitored separately and hence a need to have a framework that monitored all of them together.

The research also noted that the integration framework that was to be developed would need to interface all the existing brands of devices on top of integrating them

1.3 Objectives

To help the project solve the problem of lack of an integration framework for speed alcohol and weight monitoring, these three objectives were to be met:

The main objective was to design an integration framework for speed, blood alcohol level and weight monitoring based on previous research work.

After achieving the primary objective, two more secondary objectives were to be met.

First, a prototype based on the model that was designed was to be developed. This was to be an embedded hardware prototype.

Lastly, some data was to be simulated in order to test and evaluate the prototype.

1.5 Justification

By 1st April 2014, there were 22 brands of speed limiters in Kenya. Approved Speed Governors and Vendors (KeNHA, 2014). This large number of manufacturers was to be able to satisfy the high demand due to large number of PSV's. This number would go up as more manufacturers ventured into that lucrative business opportunity.

This research project interfaced all those devices in a single embedded system and also integrated them with BAL and weights monitoring. This meant that irrespective of the brand of speed limiter in use, the gadget was able to interface it and monitor it together with the values got from the Alco Blows and Weigh bridges.

An integrated device that monitors those 3 main causes meant that all the offences related to those causes are monitored. For instance, a road block could be used to monitor BAL of the driver. A driver may have been over speeding along the highway and then slow down when near that road block. This meant that, irrespective of whether he passes or fails BAL test, he wouldn't be penalized for over speeding. Since the device monitored all the three factors at the same time, then it meant that no offence would go unnoticed.

The research project also noted that weights and BAL were recorded in physical forms only. This led to high levels of corruption as offenders could bribe their way out of trouble, leaving no evidence or record for their wrong doings. (Daily, 2014) This research project came up with an embedded device that recorded all the instances that the offenders broke the laws on Over speeding, Drunk driving and Overloading. The records would act as evidence when penalizing the offenders. There were a lot of cases that involved bribing the traffic officers if a driver was found to have exceeded the weights, speeds and BAL limits. Since this hardware product kept log of the offences and also involved automatic tickets creation, this would reduce cases of bribing. This is because the evidence is permanently logged in the hardware device and also because an automatic penalty given would now require the driver to follow the due process in orders to clear the fines.

1.6 Scope

The main research problem that this research project tried to solve was lack of an integration framework for speed, alcohol and weight monitoring. Any other factor apart from these three factors was never monitored in the integration framework.

1.7 Research Questions

To help the project solve the problem of lack of an integration framework for speed alcohol and weight monitoring, these five research questions were to be answered in the in the project work:

Could the research come up with a suitable framework for monitoring speed, alcohol and weight?

Could the research come up with an embedded hardware device that was able to do the monitoring?

Could the embedded hardware device be able to retrieve data from a speed limiter, weighbridge and an Alco blow?

Could the proposed device be able to log and record all the instances of speed, alcohol and weight collected?

Could the proposed device be integrated with third party systems which are meant to improve the user's experience?

1.8 Assumptions

To help the project solve the problem of lack of an integration framework for speed alcohol and weight monitoring, these three assumptions were done:

High speed weights in Motion (HSWIM) weigh bridges were in use in all the main roads and not the old generation manual bridges.
High speed weights in Motion (HSWIM) weigh bridges were available in all roads that we wanted to monitor the PSV's from.
Lastly, the research also assumed correct working of weigh bridges and speed limiters. The device got data from weighbridges and speed limiters and it assumed that this data was all true.

2. LITERATURE REVIEW

This chapter starts by giving a review of speed, weighing and body alcohol levels. In those three areas, a discussion is made of the existing technology in both a mechanical and computing view. After that review, the chapter discusses previous work on integrating frameworks, discussing in detail four suitable frameworks namely; On-board network level fault diagnosis in Automobiles Framework, Data Centre Physical Threats Monitoring Framework and Two stages Health Diagnosis framework. After that, a description of common problems facing integration frameworks is given. Lastly the chapter discusses the chosen Conceptual framework based on a modified On-board network level fault diagnosis in Automobiles Framework. This chosen Conceptual framework suits our research project best.

2.1 Speed review

Currently, speed limiter/ speed governor is used to regulate over speeding in vehicles. It measures and regulates the speed of a moving vehicle. A good example of a governor is the centrifugal governor. Centrifugal governor uses weights mounted on string loaded arms to determine how fast a shaft is spinning and the uses proportional control to regulate the shaft engine.

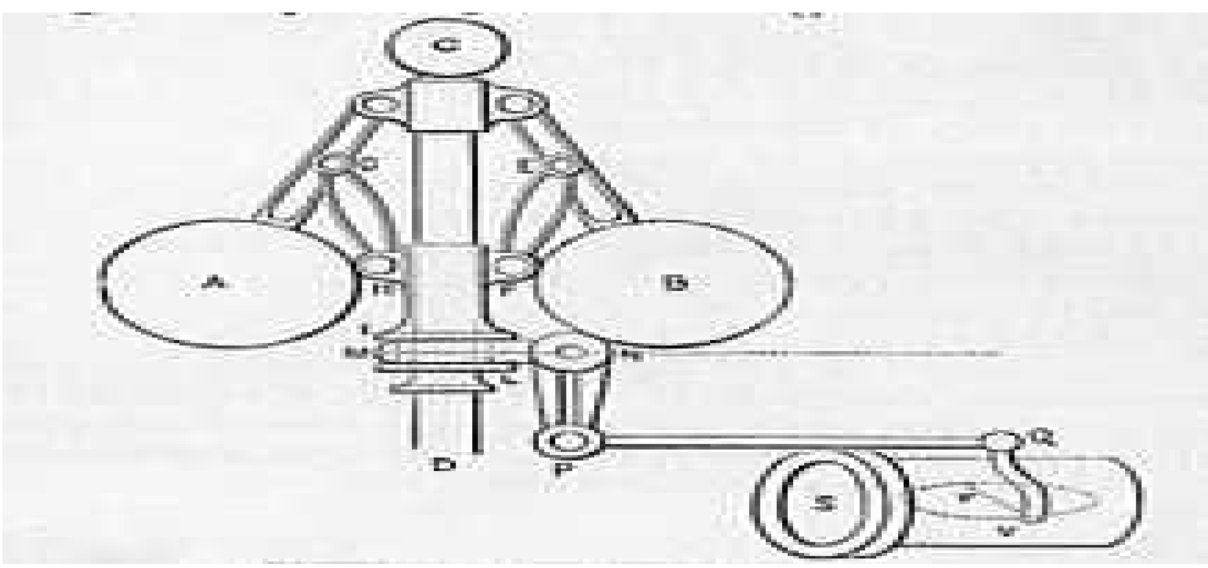

Figure 2.1: Schematic of an Engine Governor Speed Governors **(**Wikipedia, 2014)

Even if the traditional speed governors are very effective in speed regulation, they can be tampered with. For instance, after the Kenyan government introduced their use in the PSV vehicles, drivers soon found ways in which they introduced mechanical switches that could activate or deactivate the governor (Manyara, 2006). In the presence of traffic officers, the governors were switched on, in their absence, the drivers used to switch them off and hence they were able to go past the set speed limits of 80km/h (Manyara, 2006).
This led to the introduction of the new generation speed governors.

2.1.1 New Generation Speed Governors (NGSG)

These were introduced by the Kenyan Government to solve the shortcomings of the traditional speed governors. These new generation governors had to meet some specific standards as set out by Kenya Bureau of standards (KEBS), Maximum road speed Limiters for Motor Vehicles (KEBS, 2014). A full list of those requirements is given out as an attachment at the end of this document. However, their most important requirement for this research project was their ability to record the speeds as the vehicle moves.). Maximum road speed Limiters for Motor Vehicles (KEBS, 2014). NGSG has 3 main components: A Speed limiter, sensor and a valve.

2.1.1.1Speed Limiter

This allows maximum pulling power until a pre-set speed limit is reached after which it then starts to limit the speed. Once the speed drops below the pre-set speed limit, control is returned to the driver. As outlined by KEBS requirements, this process is done smoothly and does not affect the driving ability. (Dalcom, 2014) (Limited D., 2014)

Figure 2.2: Omata Speed Limiter (Dalcom, 2014)

2.1.1.2 Sensor

This is used in retrieval and monitoring of the recorded speeds. This speed data is the one used as evidence during prosecution of over speeding drivers. (Dalcom, 2014), (Limited D., 2014)

Figure 2.3: Omata Sensor (Dalcom, 2014)

2.1.1.3 Valve

This is the recording device. It is a digital, stand-alone speed data logger that records the data in real time. Data is recorded into the device by the introduction of the speed signal from the vehicle gearbox speed sensor attached to the drive shaft of the vehicle. (Dalcom, 2014), (Limited D., 2014)

Figure 2.4: Omata Valve (Dalcom, 2014)

The diagram below shows a screenshot of an already running NGSG, recording the speed, every five seconds

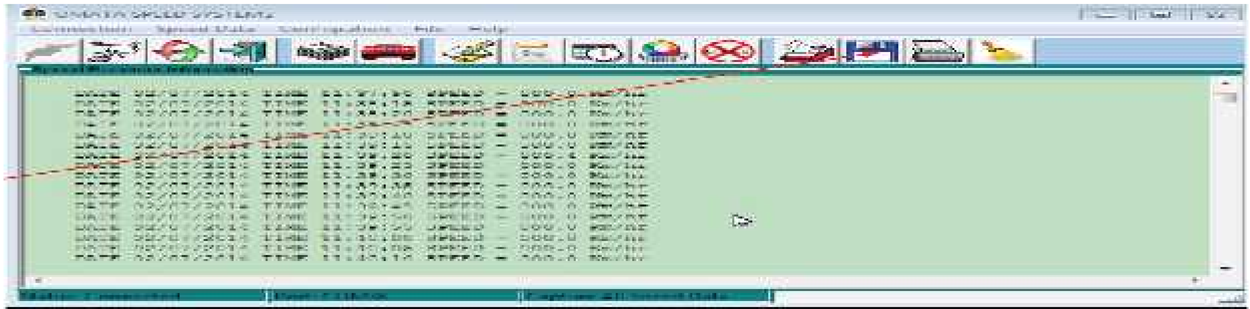

Figure 2.5: Omata Sample data set (Limited D., 2014)

2.1.2 Inputs, Processes and Outputs of New Generation Speed Limiter

Irrespective of the brand of NGSG in use, similar input, process and output exists. The speed sensors senses the speed in which the vehicle is moving and this is then transferred to the control unit an electrical pulse. This input is then processed by simple identifying if the limits have been exceeded or not. This processing is done by the control unit. The output from the control unit is then used to actuate as needed. If limits are exceeded the speed limiter actuates by restricting the amount of fuel been burnt in the engine. This is done until the speed falls to the one that is lower or equal to the maximum limit. To warn the drivers when the limits are exceeded, a sound output is given which continues the speed drops below or equal to the maximum limit. As a requirement by KEBS, the speed is then logged to a file as the vehicle moves.

2.1.2 Relevance of studying NGSG to our study

This project gets its speed data from the speed limiters logs. It would have been redundant and necessary to develop a mechanism for getting speed while the current speed limiters are doing that. However, the research found it necessary to keep a copy of the speed data as retrieved from the limiters in its new device.

2.2 Weight review

Vehicle weights are taken on weight bridges. These are large sets of scales, usually mounted permanently on concrete foundation that is used to weigh vehicles and their contents (Wikipedia, Truck Scale, 2014). In Kenya, long distance tracks takes majority of the blame when it comes to accidents caused by overloading. (Manyara, 2006) This is because they are too heavy and has more momentum making them more difficult to stop and more likely to get out of control when swerving to avoid an accident. Overloading also causes other problems with the truck such as brake failure or inability to steer (chaikinandsherman, 2014). Traditionally, there are 3 main technologies used in the weigh bridges:

2.2.1 Load Cell Systems

These are the most common and popular. The cells of these materials are made of strong materials like steel and a strain gauge is attached to it. This gauge has a wire that sends out an electronic pulse that is then used to know the weight of the vehicle (Ehow, 2014) (Systems, 2014), Weighbridges (KeNHA, 2014). That electronic pulse is directly proportional to the weight of the vehicle

2.2.2 Bending plate's machines

These machines replace load cells with metal plates that also have strain gauges. When a vehicle in placed on the plates, it exerts stress which is then converted to an electric signal that is directly proportional to the weight of the vehicle (Ehow, 2014) (Systems, 2014), Weighbridges (KeNHA, 2014).

2.2.3 Piezzo scales

These are uncommon. They use piezzo electric technology. They have sensors placed on conductors throughout the scale. When a truck pulls on this material, Voltage that is directly proportional to the trucks weight is created through the conductors. The piezzo sensors record these changes by creating an electric pulse that denotes the weight of the vehicle (Ehow, 2014) (Systems, 2014), Weighbridges (KeNHA, 2014).

The main disadvantage of weighbridges is the traffic jams and the delays that are caused when the trucks has to pull over in order to take weights measurements. (Star, 2014) Because of this, the Kenyan government introduced use of High Speed Weighing in Motion (HSWIM) system, on September 2013. Currently, Mariakani, Athi River, Gilgil, Mtwapa, Isinya, Juja, Busia, Rongo and Webuye Bridges are using this technology. (Star, 2014)

2.2.4 High Speed Weighing in Motion (HSWIM) system

These are designed to capture and record axle weights and gross vehicle weights as the vehicle moves over a measurement site. Weigh in motion (Wikipedia, 2014). Unlike the traditional static weights, HSWIM can measure the vehicle weight as it moves and hence only the overloaded vehicles are pulled over. The whole system overview is shown in the image below.

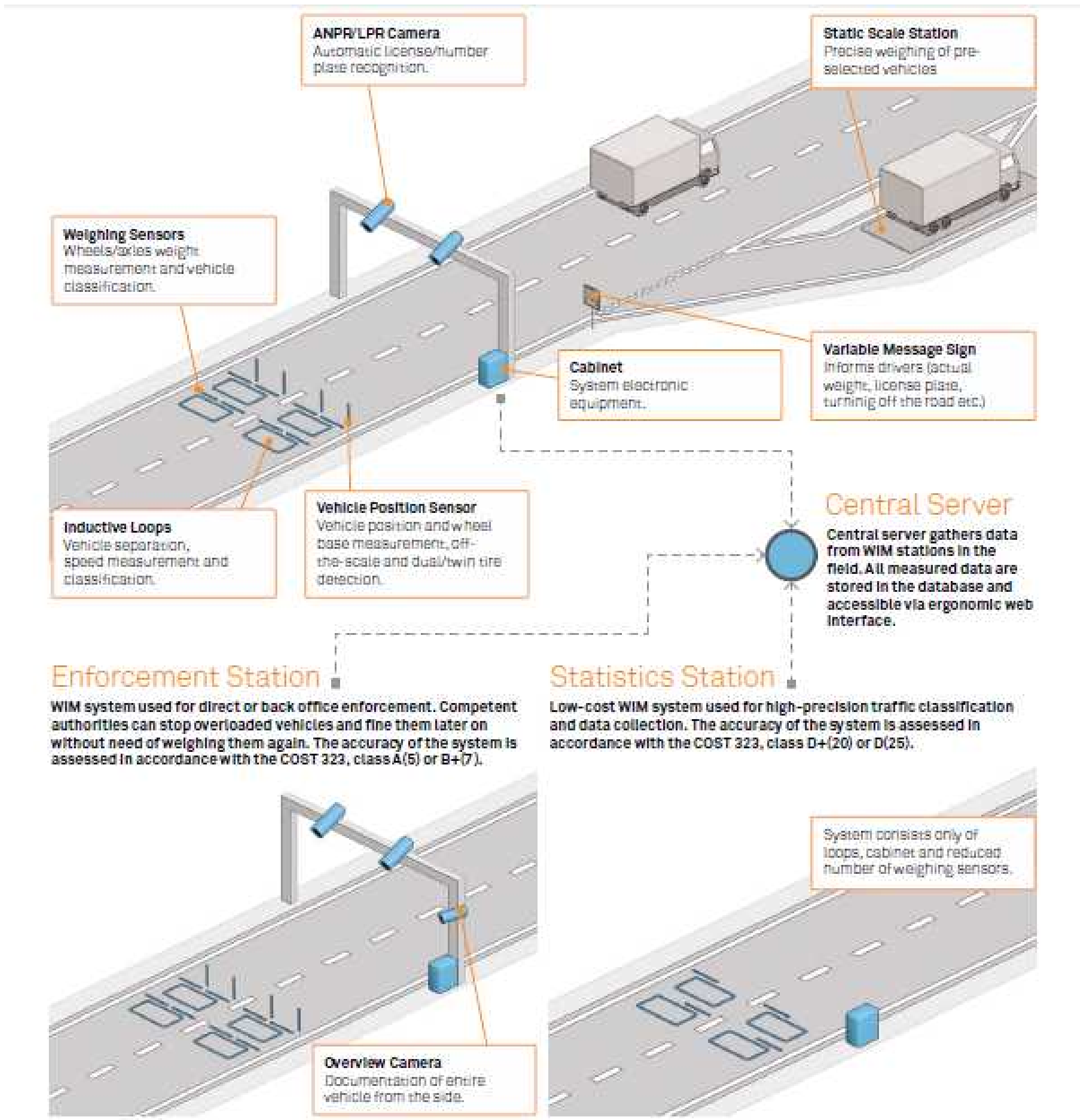

Figure 2.6: High Speed Weighing in Motion System (Weigh in motion, 2014)

2.2.5 **Inputs, Processes and Outputs of Weighing Machine**

All the four technologies for weighing machines get their input from the weight the vehicles exert. Since these weights are analogue signals, they are then converted to digital electronic pulses/voltages that are directly proportional to the weights. The pulse is then sent to the central unit that determine the actual weight of the vehicle by mapping the pulse to a weighing scale. The output that a user sees is then the mapped value on the weigh scale.

2.2.6 Relevance of studying Weighing Machine to our study

This project gets its weight data from the weigh bridges electronic pulses. Just like speed, it would have been redundant to include a weighing mechanism in the new device but a copy of the collected weight from the pulse was saved in the new device.

2.3Alcohol review

Breathalyzers (Alco blows) are used to measure the blood alcohol content of a person.
They were introduced in Kenyan roads to curb the problem of drunk driving. The law requires that no one should drive a vehicle if they have consumed alcohol in excess of 30 microgram's in 100 millilitres of breath, 80 milligrams of alcohol in 100 millilitres of blood and 107 milligrams of alcohol in 100 millilitres of urine (SoftPedia, 2014).
These are the two main technologies used in breathalyzers:

2.3.1 Intoxilyzer

It analyses the optical properties of alcohol in the infrared spectrum. As air is being blown, an infrared beam is projected. A photocell measures the beam for variations. These variations are then sent as electronic pulses. These electronic pulses are directly proportional to the amount of blood alcohol content. World Report on Road Traffic Injury Prevention 2007 (WHO, 2007), (Breathalizer.net, 2014), (HowStuffWorks, 2014)

2.3.2 Alco Sensors

It has a fuel cell that is made of 2 platinum electrodes separated by acid electrolyte porous material. When air with alcohol flows from one side to another, Acetic acid is produced together with positive ions and negative electrons. Due to these three components, an electric current is produced. The current generated then gives an accurate test of the Blood Alcohol content.

World Report on Road Traffic Injury Prevention 2007 (WHO, 2007), (Breathalizer.net, 2014), (HowStuffWorks, 2014)

Due to health reasons; disposable mouthpieces are used on Kenyan roads. (News24, 2014)

2.3.3 **Inputs Processes and Output of an Alco Blow**

The input of the intoxilyzer and Alco sensors is human. However, since this is an analogue signal, the two technologies needs conversion to digital pulses. Even if the conversion method differs in the two technologies as it had been seen earlier, they all end up with a digital electronic pulse that is fed to their control unit. To get the actual alcohol level, which is the output; these pulses are mapped to a scale by the control unit.

2.3.4 Relevance of studying Alco Blow to our study

This review only tries to describe what is been used on the roads. For the project device, an inbuilt Alco blow was developed but it worked just like its two counterparts above. It got an input from the breath; this input was then converted to an electronic pulse which was lastly mapped to the correct BAL scale.

2.4 Previous work on integration frameworks

There are many projects that have been dealing with integration frameworks in different fields. This research discusses some of these works that the project will inherit while coming up with its proposed model.

2.4.1 A Framework and methods for on-board network level fault diagnosis in Automobiles

This research project proposed a framework that achieves automobiles diagnostics, focusing on network level faults. Electronic Control Units (ECU’s) helps achieve advanced vehicle control and eliminates bulky wiring. However, they lead to increased complexity in vehicle fault diagnosis since Off-Board fault diagnosis by using diagnostics codes can be unwieldy when dealing with this complex ECU’s (Suwatthikul, 2008).

2.4.1.1 On-board network level fault diagnosis in Automobiles Framework

The proposed framework presented a way for real-time vehicle diagnosis at network level.

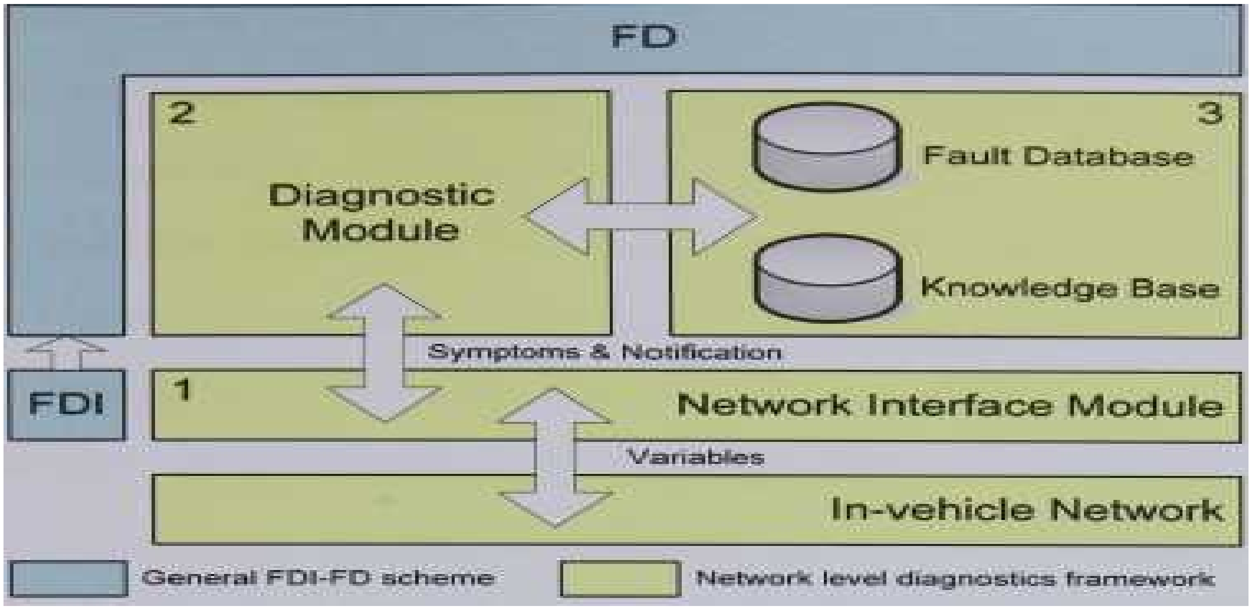

Figure 2.7: Framework for on-board network level Diagnostic (Suwatthikul, 2008)

2.4.1.1.1 Network Interface Module (NIM)

This was responsible for detecting network level faults. After initializing software variables, and retrieving message specifications, it monitored message communication on a network and sent an error notification to the diagnostics module after faults were detected.

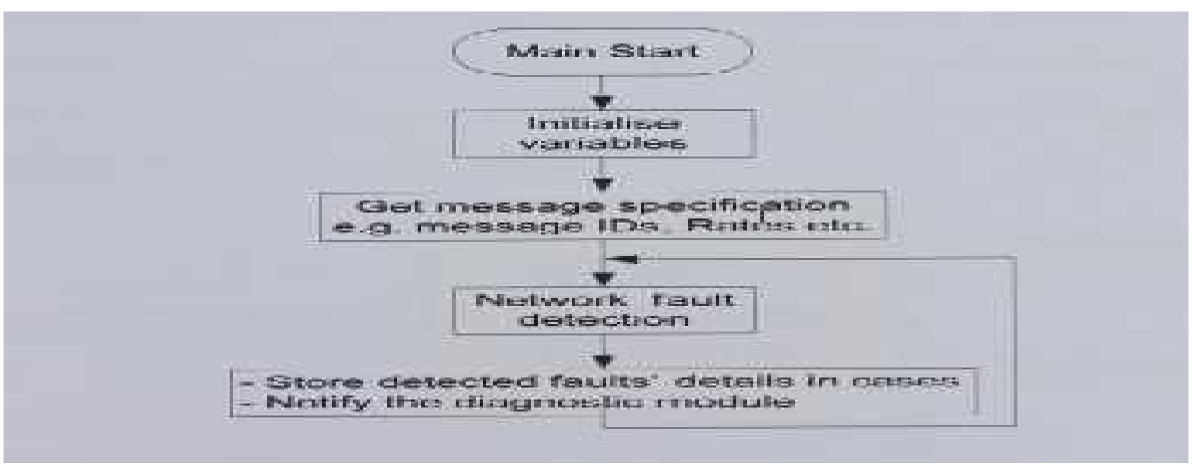

Figure 2.8: Flowchart for NIM (Suwatthikul, 2008)

2.4.1.1.2 Diagnostics Module(DM)

This module did diagnosis. After NIM detected some faults, it passed symptoms of the fault for diagnosis to the DM. After completing the diagnosis, the DM stored diagnostics results in the fault base.

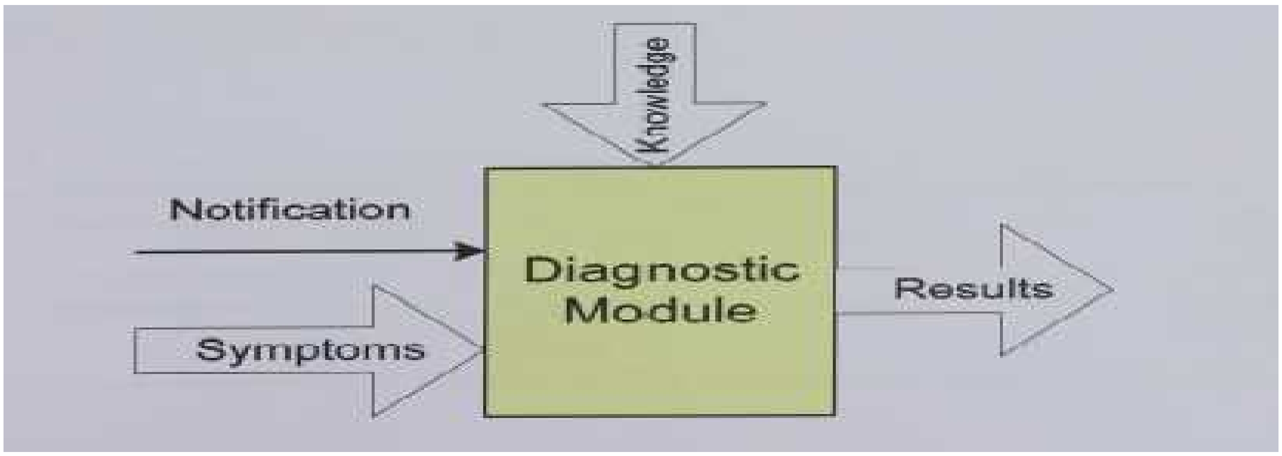

Figure 2.9: Process Flow for DM (Suwatthikul, 2008)

2.4.1.1.3 Knowledge Base and Fault Base

2.4.1.1.3.1 Knowledge Base

This provided a library of correct fault information. Information from manufacturers and device suppliers, message specification and test data was used to build the knowledge base.

2.4.1.1.3.2 Fault Data Base

This stored diagnostics results, and relevant fault information. This database could be accessed by either an onboard whole vehicle management system, if implemented, or an external off board diagnosis tool.

2.4.2 Monitoring Physical Threats in a data centre

This research project proposed a framework for monitoring data center environments. The framework helped to monitor different classes of threats and provided best practices in leveraging the collected data to reduce downtime (Christine Cowan, 2013).

The diagram below shows different threats to a data center that were identified:

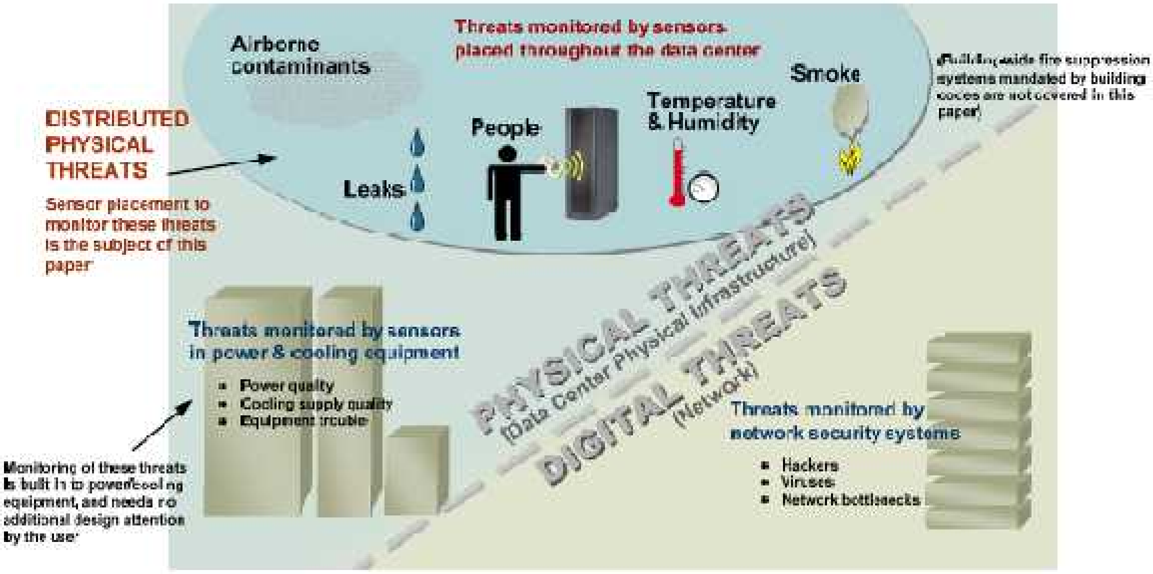

Figure 2.10: Threats to a data center (Christine Cowan, 2013)

The research project proposed different sensors that could be used to monitor those threats:

Table 2.1: Threats and Sensors

THREAT	SENSOR
Air Temperature	Temperature Sensors
Humidity	Humidity Sensors
Liquid Leaks	Rope Leak Sensors Spot Leak Sensors
Human Error And Personal Access	Digital Video Camera, Motion Sensors, Door Contacts, Glass-Break Sensors, Vibration Sensors
Smoke/Fire	Supplemental Smoke Sensors
Hazardous Airborne Contaminants	Chemical / Hydrogen sensor Dust Sensors

2.4.2.1 Data Centre Physical Threats Monitoring Framework

Aggregating Sensor Data

After selecting and placing sensors, the collected data from sensors was analyzed. Rather than send all sensor data directly to a central collection point, the research project proposed to have aggregation points distributed throughout the data center with alerts and notifications capability at each aggregation point. The reasons for aggregation points were to eliminate the single point of failure risk of a single central aggregation point and also support point of use monitoring of remote server rooms and telecom closets. Aggregators communicated through the IP network with central monitoring system

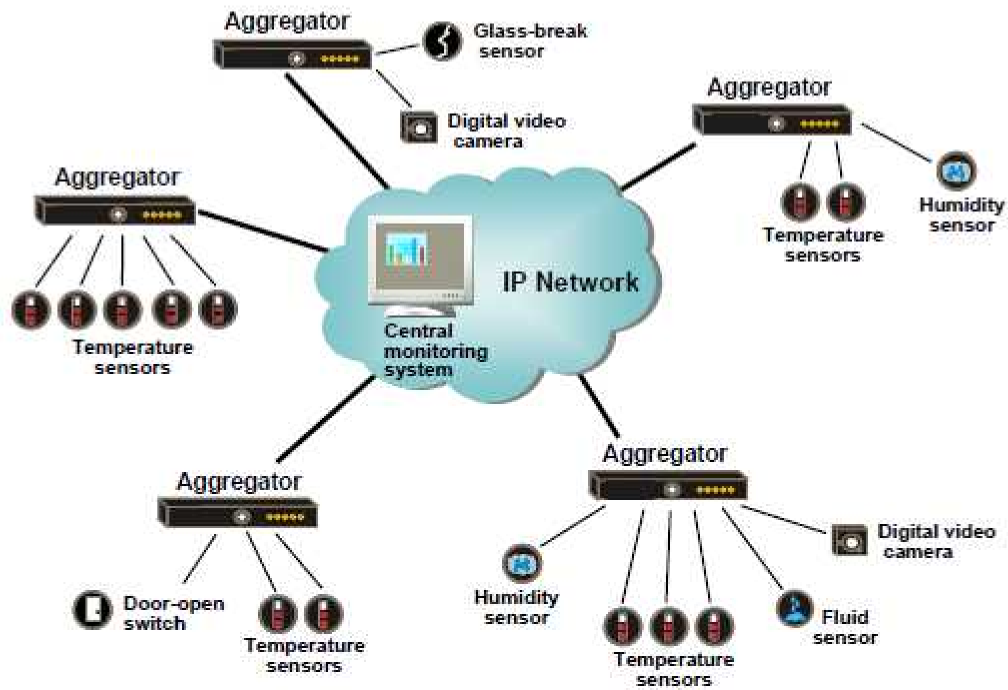

Figure 2.11: Aggregating the sensor data (Christine Cowan, 2013)

The aggregators interpreted the sensor data and sent alerts to the central system and/or directly to the notification list.

2.4.3 A Two-Stage Diagnosis framework for Wind Turbine Gearbox Condition Monitoring

This research project presented a vibration-based two stage fault detection framework for failure diagnosis of rotating components in a wind turbine (Tamilselvan, 2013).

2.4.3.1 Two-Stage Health Diagnosis framework

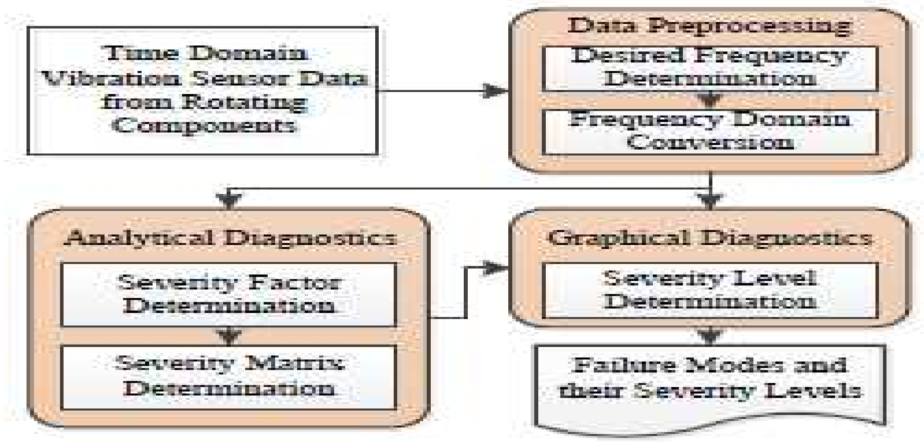

Figure 2.12: Two stage health Diagnosis framework (Tamilselvan, 2013)

The framework was made up of three modules: Data Preprocessing, Analytical diagnostics and Graphical Diagnostics.

2.4.3.1 .1 Data Preprocessing

The rotating components of a wind turbine provided time domain vibration data. This module converted this time domain signal into a frequency domain signal. It first determined the right range of frequency to use, determined relationship between sensors and components so that different frequency ranges were noted and lastly did the conversion.

2.4.3.1 .2 Analytical Diagnostics

This module aimed at detecting defects using sideband and kurtosis evaluation. It included Severity factor determination and Severity Matrix Determination. It followed 4 steps: (i) determined maximum amplitude values for sidebands and desired frequency; (ii) Determined Kurtosis values for the sideband; (iii) calculated severity factors 1, 2 and 3; and lastly, Formulated defect severity matrix.

Sidebands were indicators of the failure modes in the frequency spectrum on each rotating component based on their spread on both sides of the desired frequency.

2.4.3.1 .3 Graphical Diagnostics

This module determined the level of severity of the defect. It got its input from analytical diagnosis results. The failure modes and their severity levels were determined from the frequency domain signals by graphical diagnosis hence provided a graphical verification.

2.5 Common Issues in Integration Frameworks

There are some issues that have been affecting previous research on integration frameworks as seen in the below discussion.

The first issue is data formatting. Data formatting involves combining data that can come with different formats from its sources. It is still a pervasive challenge faced in frameworks that need to query across multiple autonomous and heterogeneous sources. (Bruce Steve, 2012) , (Hopkins George G., 2012).

The second issue is data integration. Data integration involves combining data residing in different sources and providing users with a unified view of these data. It is still a pervasive challenge faced in frameworks that need to query across multiple autonomous and heterogeneous sources. (Wikipedia, 2015) , (Halevey, 2012), (Ives Zachary G, 2012).

Thirdly, System redundancy is a major issue when it comes to frameworks with multiple sources of inputs. (Younis Mohamed F. Younis , 2007), (Waegli Adrian , 2008), (White Jacob M , 2012). The redundancy introduces a backup system to the framework incase the primary components fail. Issues always arise here when it

Fourthly, Dependability is a big issue. It describes the ability of a system or component to function under stated conditions for a specified period of time. It is a major issue when it comes to frameworks with multiple sources of inputs. (Edward Jeremy, 2007) , (Bernard Koelf, 2009). An effective framework should always produce the right output for the inputs provided. However, this is not always the case as seen from those two references.

Lastly, Data integrity is still a major issue when it comes to frameworks with multiple sources of inputs. (Younis Mohamed F. Younis , 2007), (Waegli Adrian , 2008), (White Jacob M , 2012). It refers to maintaining and assuring the accuracy and consistency of data over its entire life-cycle, and is a critical aspect to the design, implementation and usage of any system which stores, processes, or retrieves data.

2.6 Conceptual Framework

This research project borrowed its framework from on-board network level fault diagnosis in Automobiles that had been discussed in 2.4.1.1.

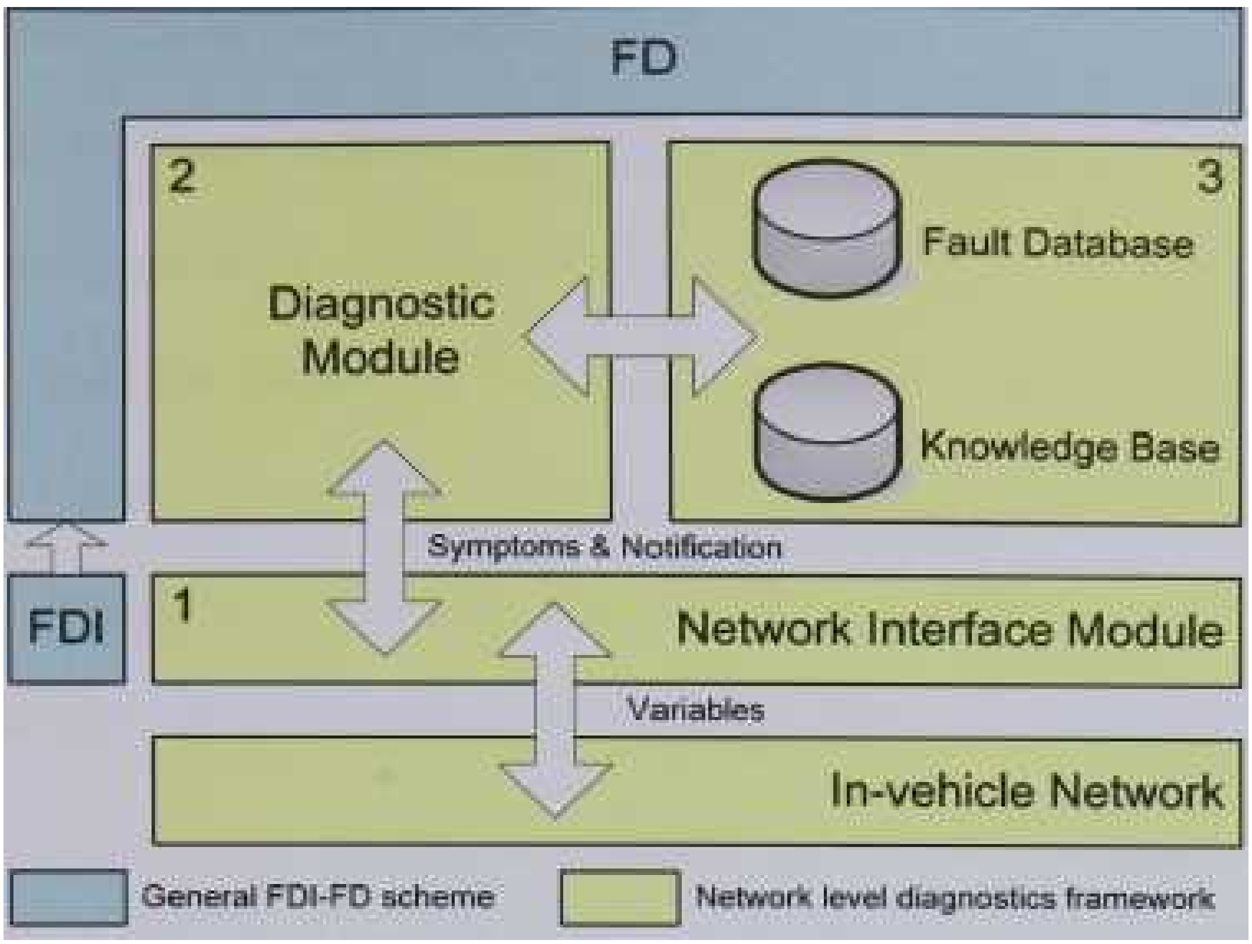

Figure 2.13: On-board network level Diagnostics (Suwatthikul, 2008)

2.6.1 Network Interface Module (NIM)

This was responsible for detecting network level faults, in on-board network level fault diagnosis in Automobiles. After initializing software variables, and retrieving message specifications, it monitored message communication on a network and sent an error notification to the diagnostics module after faults were detected.

For this research, it was collecting data from the NGSG, inbuilt Alco Blow and the weighbridge. After this data was collected, it was fed to diagnostic module

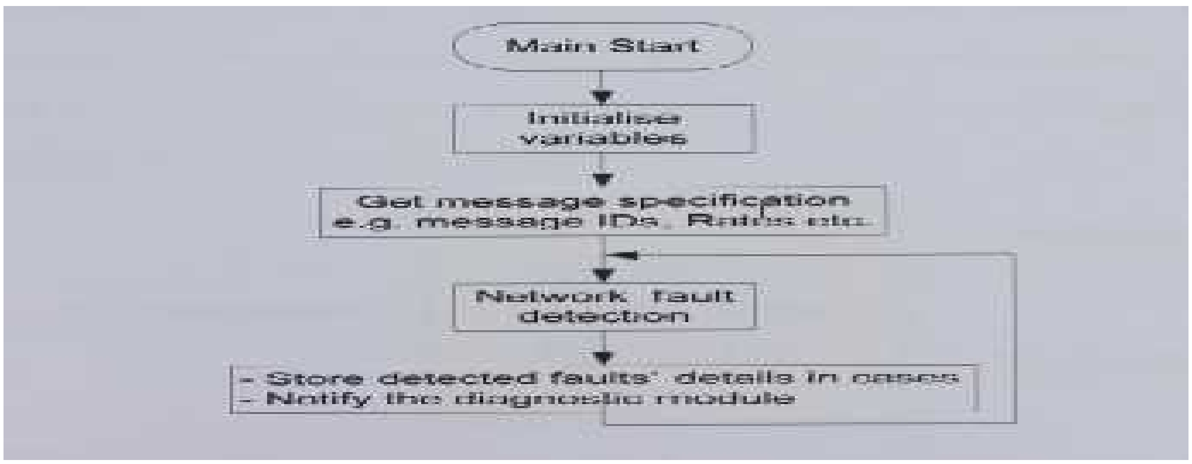

Figure 2.14: Flowchart for NIM (Suwatthikul, 2008)

2.6.2 Diagnostics Module (DM)

This module did diagnosis, in on-board network level fault diagnosis in Automobiles. After NIM detected some faults, it passed symptoms of the fault for diagnosis to the DM. After completing the diagnosis, the DM stored diagnostics results in the fault base.

For this research, it received processed data from all aggregated modules of the NIM. It has logic to retrieve the knowledge from the knowledge base when it started up and also saved the data from the alcohol processing module, weights processor, and also instances of committed offences to the fault database. It used rules to identify if the set limits that had been retrieved from the knowledge base had been violated.

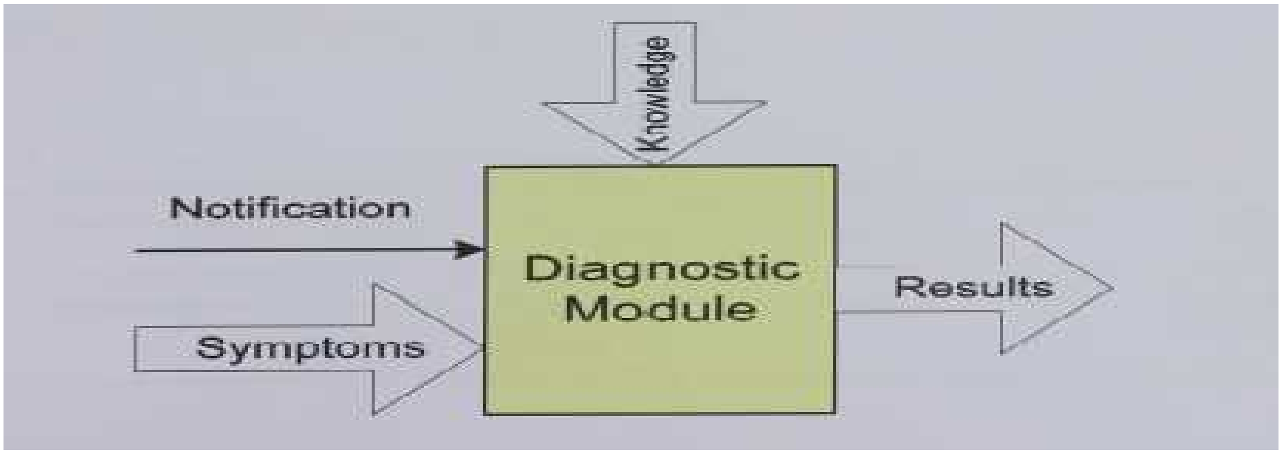

Figure 2.15: Process Flow for DM (Suwatthikul, 2008)

2.6.3 Knowledge Base and Fault Base

Knowledge database provided a library of correct fault information and fault data base stored diagnostics results, and relevant fault information, in on-board network level fault diagnosis in Automobiles.

For this research, Fault base was the main repository for storing data collected from the alcohol processing module and also the weights processor. It also logged committed offences. It had an interface that connected to the integration system. In this case, the integration system was the ticketing system. Knowledge base stored the rules that determined the limits set for the alcohol, speed and weights.

2.7 Relationship between the Conceptual Framework and the research Problem

The main research problem that this research project tried to solve was lack of an integration framework for speed, alcohol and weight monitoring. This conceptual framework provided a framework in which I could monitor the three factors; hence it was of great value.

Even if a lot of research had been done on integrating frameworks, the research project discovered that no work had been done on integrating the 3 factors before, probably due to the fact that NGSG were tailored for the Kenyan market and their usage is relatively new in the roads.

3 METHODOLOGY

This chapter starts by outlining the research strategy that was used, the sources of data and their relevance, tools that were used to collect that data and also their justification. After this, it briefly touches on how the testing was done, and how this testing would answer the research questions. It then discusses the architectural model in details, showing how it fits in the conceptual framework. A detailed description of each component, in the architectural model is also discussed, showing the relationship between the components in the architectural model and the ones in the conceptual framework. After this, a discussion of the computing importance of the proposed framework is done. Lastly, clear steps and guidelines are given on practical implementation of the framework.

3.1 Research Strategy

This research project was an applied computing research. It started by investigating all the set requirements that the new generation speed governors, weighting bridges and breathalyzers must fulfil before they are authorized for usage. This review of requirements was important as these requirements were part of the data that was required for the device to be developed. The set requirements also provided the rules that were to govern the device. To investigate these requirements, the research reviewed requirements documents for NGSG, weigh bridges and Alco Blows manufacturers. It also did a survey of what the Kenyan law requires for the usage of the three devices.

After this, the research identified all the different new generation brands of speed governors, breathalyzers and the digital weighing bridges in use, identifying the data flow process for all those devices. This was important as the available different brands had different architectural structures, and all of these were to be interfaced and integrated in one hardware device. For instance, different speed limiters used different file formats for their data storage. To achieve this, the research reviewed data flows for the different new generation brands of speed governors, breathalyzers and the digital weighing bridges in use.

Thirdly, the research came up with a new framework that interfaced and integrated all those different devices. This was important as it was the research main objective. To achieve this, a suitable framework was selected from previous research work. Slight modifications were done in order to suit the research conditions.

Fourthly, the research developed a hardware prototype, based on the new framework. This was important so that the research could identify if the proposed framework would be successfully implemented in a hardware device. The theoretical framework would have been useful if only it would be implementable in a hardware device. The hardware was built on the Arduino framework and used both C and Arduino C languages for implementation.

Lastly, the hardware prototype was tested with some simulation data. This was important as the research would then tell if the hardware device developed, based on the proposed framework, was working correctly. To achieve this, sample Speed data was simulated by varying a potentiometer and this data was stored in the device memory in form of text files. Sample data for weights was also simulated by varying a potentiometer. Since the hardware system has an embedded Alco Blow, sample data for BAL was simulated by exposing the Alco Blow to different alcohol levels.   Tests were also done on the ticketing system to see if it was issuing tickets as required.

3.2 Sources of data and their relevance

This project derived its data from three main sources: Technical Users (Manufacturers), Non-Technical sources (users), and researchers.

The technical users supplied data on how the device data flows. For instance, speed governor manufacturers gave some information on the formats in which they store the data in the devices, and weight bridge operators also provided data on the ranges of generated electric pulses that were used for different weights. One of the non-technical users was the Traffic Police. Data was collected from them showing different limits used in the roads. This was necessary for integration with the automatic ticketing system. Previous research gave the project information on some of the related work to the study. This was instrumental when coming up with the framework and also the model.

3.3 Tools methods for data collection and their justification

Data collection was done by documents analysis, literature review, questionnaires and also interviews.

Documents analysis and Literature review were used in collecting technical information in the collected manuals. Different devices have different way of operation. To understand all these different ways of operation the research needed to use documents analysis and literature review to collect all that data. Those two tools were also be used to collect data from previous researchers.

Questionnaires and interviews were also done on both the technical and non-technical data sources. For instance, these two tools were used in order to collect data on how the devices work, and also to collect data from the traffic officers.

3.4 Testing

Testing was done through simulation experiments. These simulations provided the weight, BAL and speed samples. Sample Speed data was simulated by varying a potentiometer, and this data was stored in the device memory in form of text files. Sample data for weights was also simulated by varying a potentiometer. Since the hardware system has an embedded Alco Blow, sample data for BAL was simulated by exposing the Alco Blow to different alcohol levels. Tests were also done on the ticketing system to see if it was issuing tickets as required.

That testing was driving at answering the research questions. First, it was aimed at verifying if the framework was convenient enough to monitor speed, weights and the BAL. Secondly, tests were made to verify that the built prototype, based on the chosen framework, can be used to monitor the three factors. Thirdly, since the prototype was designed for use on the roads, tests were done to verify that it could retrieve data from a real NGSG, and weighbridge. Fourthly, proper tests were done to verify that all the instances of the three factors were logged correctly as they were collected. Lastly, proper tests were done to show that the prototype could be integrated to a third party system, meant to improve the user experience.

3.5 Architectural Model

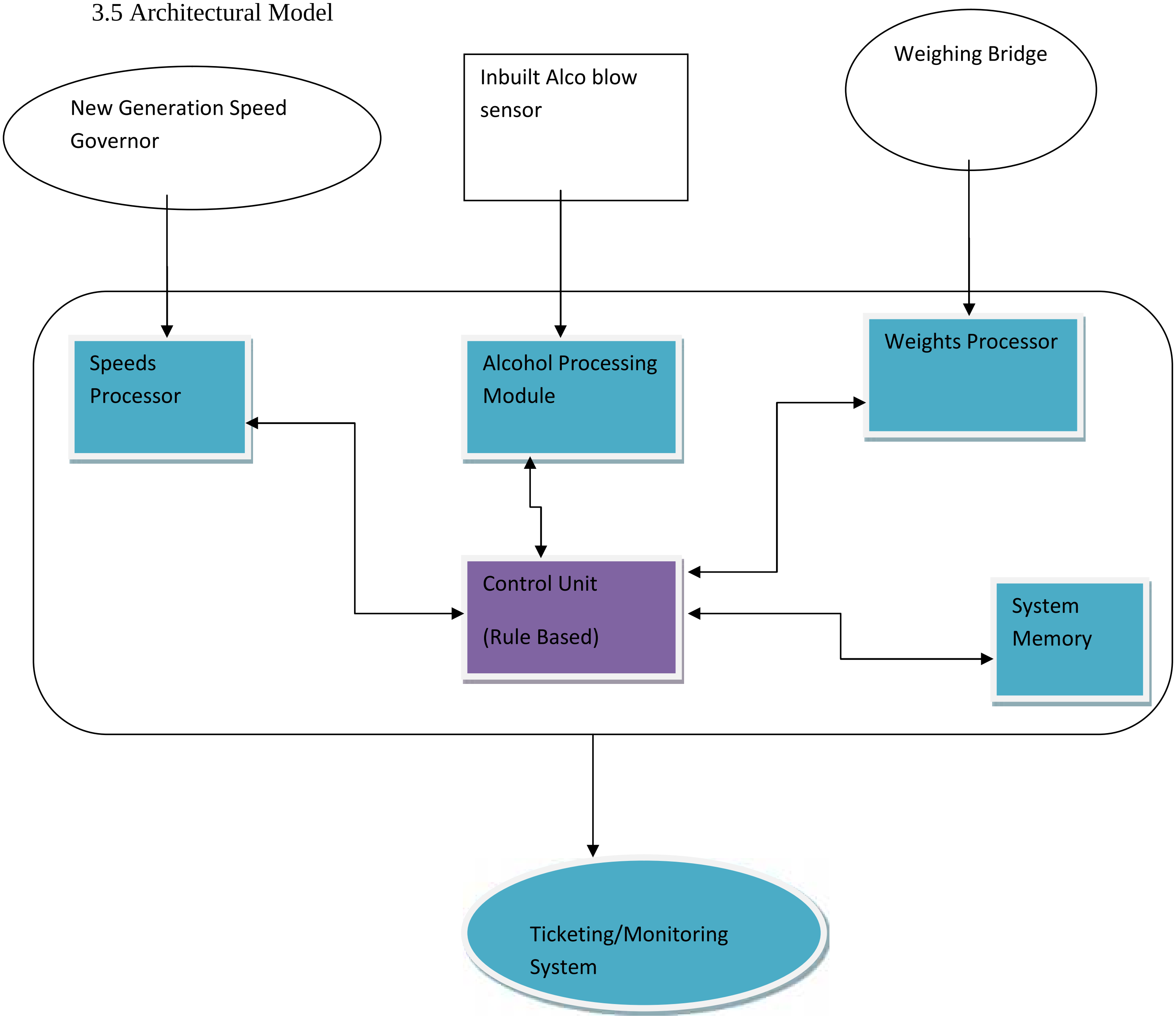

Figure 3.1: Architectural Model

3.6 Architectural Model Components.

3.6.1 Speeds Processor

This module got the data that was logged in second generation speed governor. Since that data can be in different formats, it then had libraries that enable it to read from those different formats. Currently, all NGSG logs the data in a text file. When the device started up, it loaded its variables that were fetched from the knowledge base via control unit. All data from NGSG was processed here. The processed data was then sent into the control unit.

3.6.2 Inbuilt Alco blow sensor

This project used an inbuilt alcohol sensor to simulate values that are got from a breathalyzer. The sensors provided an electric pulse, depending on the level of alcohol present, to the alcohol processing module.

3.6.3 Alcohol Processing Module

This module processed the data that was got from the alcohol sensor. When it started up, it loaded its variables that were fetched from the knowledge base via control unit. This processed data was then sent to the control unit. This module also had a timer that activated collection of the signal from the inbuilt Alco blow sensor.

3.6.4 Weights Processor

This module processed the signal received from the weighing bridge. When it starts up, it loaded its variables that are fetched from the knowledge base via control unit. The signal received was an electric pulse. Since the voltages received were high, then it first mapped them from 0-5 volts pulses that could be used in an Arduino processor. In this project, these values were simulated from a potentiometer. The final pulse was sent to the control unit for processing.

3.6.5 Control Unit

This was the central processing area. It received processed data from all aggregated modules of the NIM. It has logic to retrieve the knowledge from the knowledge base when it started up and also saved the data from the alcohol processing module, weights processor, and also instances of committed offences to the fault database. It used rules to identify if the set limits that had been retrieved from the knowledge base had been violated.

3.6.6 System Memory

This was made up of two blocks: Knowledge base and Fault Base.

Fault base was the main repository for storing data collected from the alcohol processing module and also the weights processor. It also logged committed offences. It had an interface that connected to the integration system. In this case, the integration system was the ticketing system. Knowledge base stored the rules that determined the limits set for the alcohol, speed and weights.

3.6.7 Ticketing/Monitoring System

This is any third party system that is interfaced to the hardware device via fault base. In this case, it was a basic ticketing/Monitoring system.

3.7 Relationship between the conceptual framework and the architectural model.

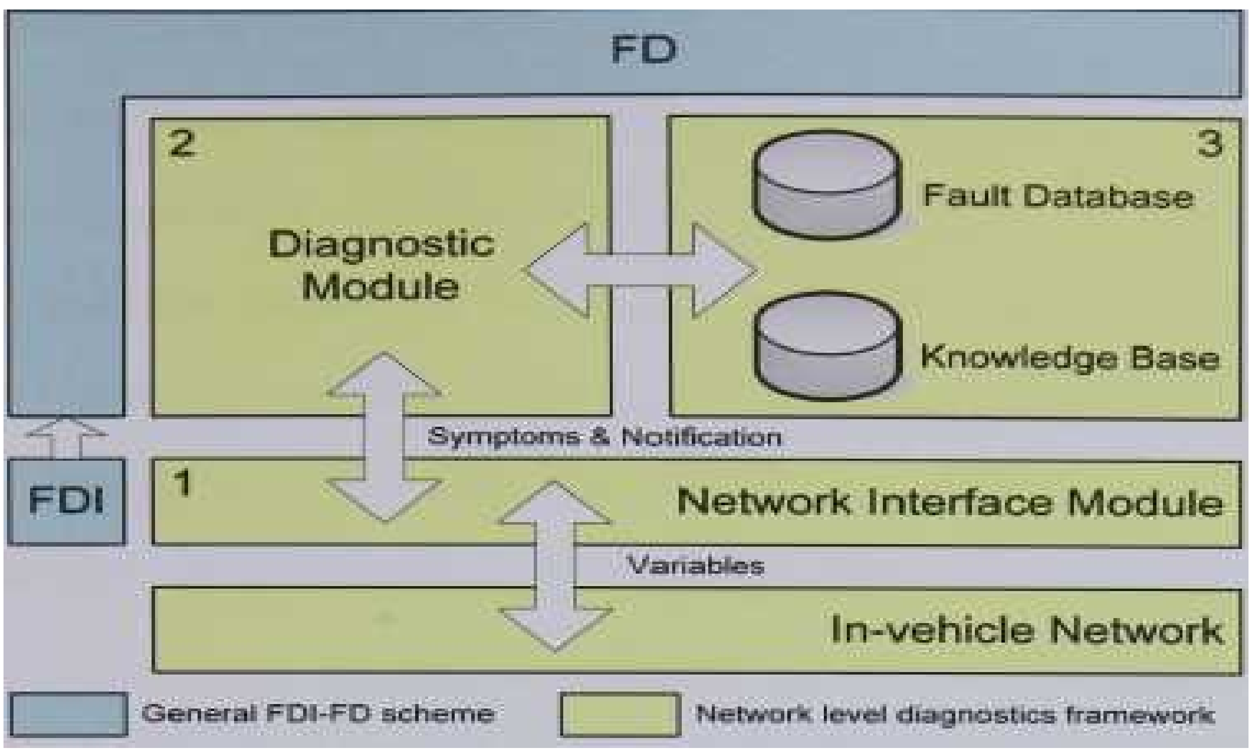

Figure 3.2: On-board network level Diagnostics   (Suwatthikul, 2008)

The Network interface Module in the conceptual framework was used to collect data from the NGSG, inbuilt Alco Blow and the weighbridge. After this data was collected, it was fed to diagnostic module.

The Diagnostics Module received processed data from all aggregated modules of the NIM. It has logic to retrieve the knowledge from the knowledge base when it started up and also saved the data from the alcohol processing module, weights processor, and also instances of committed offences to the fault database. It used rules to identify if the set limits that had been retrieved from the knowledge base had been violated.

The Fault base was the main repository for storing data collected from the alcohol processing module and also the weights processor. It also logged committed offences. It had an interface that connected to the third party integration system. In this case, the integration system was the ticketing system. Knowledge Base stored the rules that determined the limits set for the alcohol, speed and weights.

3.8 Computing Importance of this framework

Before this research, there had been no attempts to come up with a framework that monitors Speed, Blood Alcohol Level and Weights in a PSV vehicle. This research effectively bridged this gap. Initially, over speeding, overloading and drunk driving were monitored separately on the Kenyan roads. Even if there had been many projects dealing on integrating physical factors into one framework, no work had been done on integrating these three factors. This then means that this research added knowledge about a framework that does monitoring of the factors concurrently.

3.9 Practical Implementation of the framework

To successfully implement the framework in a hardware device, the components below were developed in the order that they are explained.

3.9.1 Speeds Processor

This was intended to retrieve data from NGSG. The steps used here were:

- Libraries were built to retrieve the NGSG data that can be in different formats.

- The module loaded all the set variables it needed from the knowledge base via the control unit.
- The data processed was then sent to control unit

3.9.2 Inbuilt Alco blow sensor

This was an inbuilt alcohol sensor. It was intended to simulate values that can be got from a breathalyzer. The sensor provided the equivalent electrical pulses for all the alcohol levels the device was exposed to. The step use here was

- An inbuilt alcohol sensor was developed.

3.9.3 Alcohol Processing Module

This module processed the data that was got from the alcohol sensor. To implement it;

- The module loaded all the set variables it needed from the knowledge base via the control unit.
- The data processed was then sent to control unit
- A timer was developed to activate collection of the signal from the inbuilt Alco blow sensor

3.9.4 Weights Processor

This module processed the signal received from the weighing bridge. The steps used here were:

- The module loaded all the set variables it needed from the knowledge base via the control unit.
- Since the voltages to be received were high, then it first mapped them from 0-5 volts pulses that could be used in an Arduino processor. For this project, these values were simulated from a potentiometer
- The final pulse was sent to the control unit for processing.

3.9.5 Control Unit

This was the central processing area. To implement it;

- Libraries were developed that received processed data from all aggregated modules of the NIM.
- Logic to retrieve the knowledge from the knowledge base when it started up and also saved the data from the alcohol processing module, weights processor, and also instances of committed offences to the fault database was developed.
- Rules to identify if the set limits that had been retrieved from the knowledge base had been violated were developed.

3.9.6 System Memory

This was made up of two blocks: Knowledge base and Fault Base. To implement it;

- Flash memory was extended by adding an SD shields to the Arduino board
- A Fault base library for storing data collected from the alcohol processing module and the weights processor was developed. This library also logged committed offences
- An interface that connected to the integration system was developed
- A Knowledge base library for storing the rules that determined the limits set for the alcohol, speed and weights were developed.

3.9.7 Ticketing/Monitoring System

This was any third party system that is interfaced to the hardware device via fault base. In this case, it was a basic ticketing/Monitoring system. To implement it: PLX-DAX integration was done to the fault database.

4. SIMULATION TESTS

Testing was important in this research as it was the only way the research would answer its research questions and also verify that it had correctly achieved its objective. The aims of these tests were;

- Verifying if the framework was convenient enough to monitor speed, weights and the BAL.
- Verifying that the built prototype, based on the chosen framework, could be used to monitor the three factors.
- Verifying that the hardware device could retrieve data from a real NGSG, and weighbridge.
- Verifying that all the instances of the three factors were logged correctly as they were collected.
- Verifying that the prototype could be integrated to a third party system, meant to improve the user experience.

In an aim to achieve the above five milestones, the research divided its testing into four experiments, that catered for each parameter. The speed experiment, BAL experiment, weight experiment, Combined Factors experiment and third party integration system experiment. For each experiment, purpose, objective and the procedure are explained in this chapter. After the experiments, results are discussed briefly and also an analysis and conclusion. The results and analysis was meant to determine if the testing objectives were met.

4.1 Experiments

To achieve the testing objectives, the research carried out these four experiments. The limits used in the simulation projects were 80 (for speed), 56000 (for Weight) and 1.5 (for Alcohol). 1.5 corresponded to the agreed limit of 30 microgram's in 100 millilitres.

4.1.1 Speed Experiments

The main purpose of this experiment was to see if the chosen framework would be able to integrate speed. This drove at solving the main research problem. These objectives were to be achieved in the test;

1. Generate speed values from a Speed potentiometer
2. Save the values from the potentiometer to a file
3. Verify if the control unit was able to correctly identify when the speeds were exceeded or not.

To achieve these objectives, the below steps were followed:

1. Data was simulated by varying a Speed potentiometer for a range of 0 to 180. The potentiometer values were cleaned by the mapping of all its values to a range of 0 to180. The 0-180 range corresponded to the speed a PSV can generate on the road. (0-180 km/h)
2. The generated data was saved in different file formats. The formats used were .RTFF, .CSV, .TXT, .DOC, .DOCX, .XLSX and .XLX.
3. Data was fetched from these files, mimicking the data that can be got from NGSG data file into the control unit.

4.1.1.1 Results, Analysis and Conclusion

The tables below show some of the results that were got in the experiment. BAL and weights were kept constant without exceeding their limits. The graph also shows results of below and above the set limit.

Table 4.1: Speed Experiments for Speed values less or equal to 80

Weight	Alcohol	Speed	Limit Exceeded	Ticket Issued
3,000	1	80	N	N
3,000	1	77	N	N
3,000	1	78	N	N
3,000	1	70	N	N
3,000	1	34	N	N

Table 4.2: Speed Experiments for Speed values exceeding 80

Weight	Alcohol	Speed	Limit Exceeded	Ticket Issued
3,000	1	81	Y	Y
3,000	1	88	Y	Y
3,000	1	89	Y	Y
3,000	1	99	Y	Y
3,000	1	90	Y	Y

Table 4.3: Speed Experiments for Speed values below and exceeding 80

Weight	Alcohol	Speed	Limit Exceeded	Ticket Issued
3,000	1	80	N	N
3,000	1	55	N	N
3,000	1	89	Y	Y
3,000	1	99	Y	Y
3,000	1	77	N	N

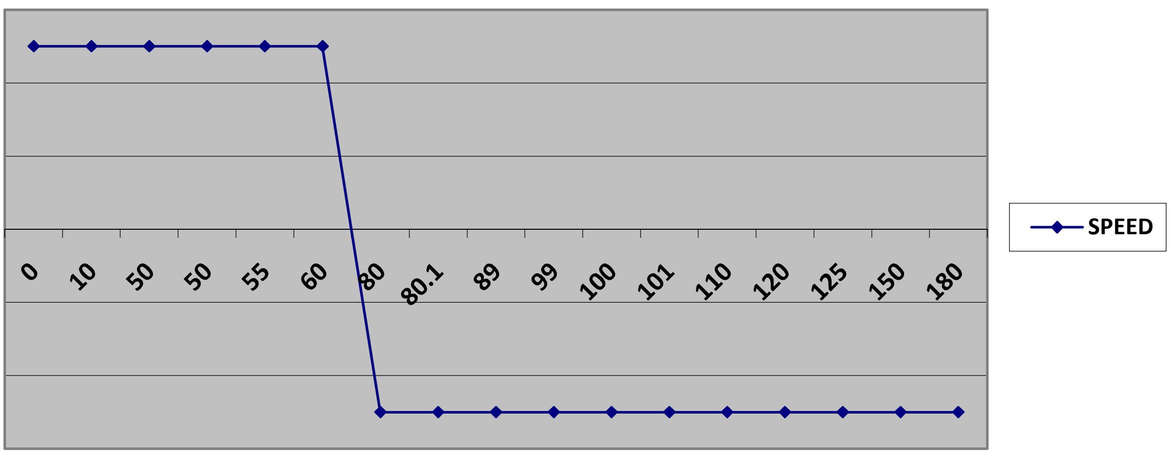

Figure 4.1: Graphical Analysis of the speed tests

The results showed that in all the 15 speed experiments done, the device correctly identified if the limits were exceeded or not. The Reason why all the experiments correctly classified the instance is the Control unit was a rule based system. The speed data generated was also correctly logged. This was important to note since it was one of the research aims.

Due to this, the research concluded that the device could correctly verify if the limits were exceeded or not. It also concluded that the device would correctly save all the speed instances generated in future.

4.1.2 BAL Experiments

The main purpose of this experiment was to see if the chosen framework would be able to integrate BAL. This drove at solving the main research problem. These objectives were to be achieved in the test;

1. Generate BAL values from an inbuilt Alco blow
2. Verify if the control unit was able to correctly identify when the BAL were exceeded or not.

To achieve these objectives, the below steps were followed:

1. Data was simulated by varying a Speed potentiometer for a range of 0 to 5. The Alco blow values were cleaned by the mapping of all its values to a range of 0 to 5. The 0-5 range corresponded to the BAL that is generated by an Alco blow of 0 to 100 milliliters.
2. Data was fetched from this Alco blow into the control unit.

4.1.2.1 Results, Analysis and Conclusion

The tables below show some of the results that were got in the experiment. Speed and weights were kept constant without exceeding their limits. The graph also shows results of below and above the set limit

Table 4.4: BAL Experiments for BAL values less or equal to 1.5

Weight	Alcohol	Speed	Limit Exceeded	Ticket Issued

3,000	1	80	N	N
3,000	1.0	80	N	N
3,000	0.5	80	N	N
3,000	0.9	80	N	N
3,000	1.5	80	N	N

Table 4.5: BAL Experiments for BAL values exceeding 1.5

Weight	Alcohol	Speed	Limit Exceeded	Ticket Issued
3,000	2.0	80	Y	Y
3,000	3.5	80	Y	Y
3,000	5.0	80	Y	Y
3,000	4.4	80	Y	Y
3,000	5.8	80	Y	Y

Table 4.6: BAL Experiments for BAL values below and exceeding 1.5

Weight	Alcohol	Speed	Limit Exceeded	Ticket Issued

3,000	0.6	80	N	N
3,000	1.5	80	N	N
3,000	1.9	80	Y	Y
3,000	4.7	80	Y	Y
3,000	0.0	80	N	N

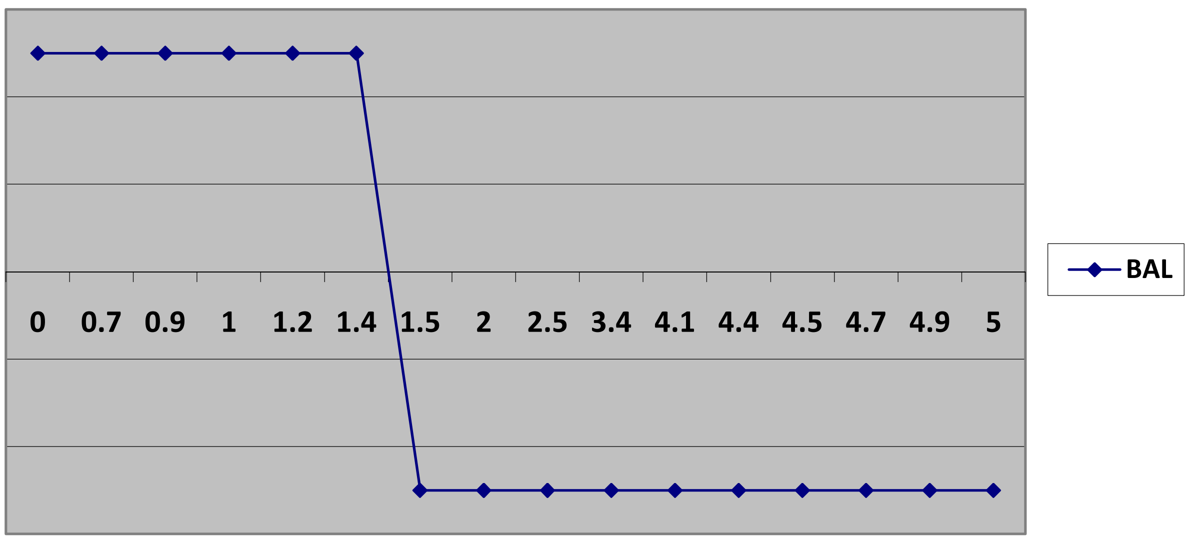

Figure 4.2: Graphical Analysis of the BAL tests

The results showed that in all the 15 speed experiments done, the device correctly identified if the limits were exceeded or not. The Reason why all the experiments correctly classified the instance is the Control unit was a rule based system. The BAL data generated was also correctly logged. This was important to note since it was one of the research aims. Due to this, the research concluded that the device could correctly verify if the limits were exceeded or not. It also concluded that the device would correctly save all the BAL instances generated in future.

4.1.3 Weight Experiments

The main purpose of this experiment was to see if the chosen framework would be able to integrate weight monitoring. This drove at solving the main research problem. These objectives were to be achieved in the test;

1. Generate weight values from a Weight potentiometer
2. Verify if the control unit was able to correctly identify when the weights were exceeded or not.

To achieve these objectives, the below steps were followed:

1. Data was simulated by varying a Weight potentiometer for a range of 0 to 240000. The potentiometer values were cleaned by the mapping of all its values to a range of 0 to 240000. The 0 to 240000 range corresponded to the weight a PSV can generate on the road. (0-240000 Kg)
2. Data was fetched from this Alco blow into the control unit.

4.1.3.1 Results, Analysis and Conclusion

The tables below show some of the results that were got in the experiment. Speed and BAL were kept constant without exceeding their limits. The graph also shows results of below and above the set limit.

Table 4.7: Speed Experiments for Weight values less or equal to 56,000

Weight	Alcohol	Speed	Limit Exceeded	Ticket Issued
56,000	1	80	N	N
46,300	1	80	N	N
33,043	1	80	N	N
44,920	1	80	N	N

21,363	1	80	N	N

Table 4.8: Speed Experiments for Weight values exceeding 56,000

Weight	Alcohol	Speed	Limit Exceeded	Ticket Issued
56,234	1	80	Y	Y
240,000	1	80	Y	Y
221,024	1	80	Y	Y
166,122	1	80	Y	Y
111,777	1	80	Y	Y

Table 4.9: Speed Experiments for Weight values below and exceeding 56,000

Weight	Alcohol	Speed	Limit Exceeded	Ticket Issued
56,000	1	80	N	N
44,554	1	80	N	N
240,000	1	80	Y	Y
133,554	1	80	Y	Y
60,459	1	80	N	N

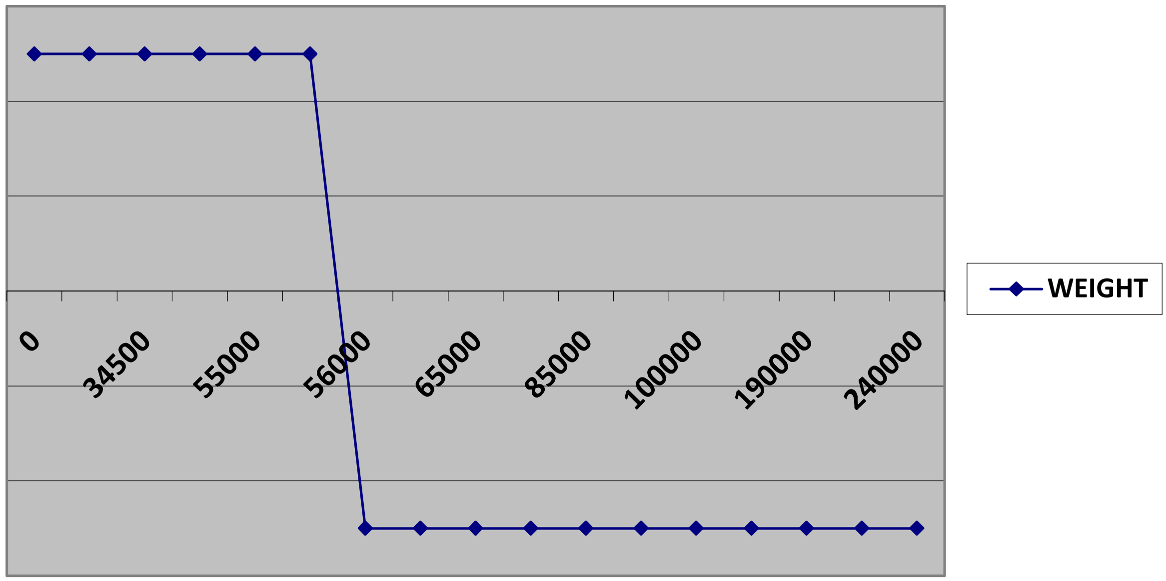

Figure 4.3: Graphical Analysis of the Weight tests

The results showed that in all the 15 weights experiments done, the device correctly identified if the limits were exceeded or not. The Reason why all the experiments correctly classified the instance is the Control unit was a rule based system. The weight data generated was also correctly logged. This was important to note since it was one of the research aims.

Due to this, the research concluded that the device could correctly verify if the limits were exceeded or not. It also concluded that the device would correctly save all the weight instances generated in future.

4.1.4 Combined Factors Experiments

The main purpose of this experiment was to see if the chosen framework would be able to integrate weight, BAL and Speed monitoring. This drove at solving the main research problem. These objectives were to be achieved in the test;

1. Generate weight values from a Weight potentiometer, speed values from a Speed potentiometer and BAL values from the inbuilt Alco blow.
2. Verify if the control unit was able to correctly identify when the limits for the 3 factors were exceeded or not.

To achieve these objectives, the below steps were followed:

1. Data was simulated by varying a Weight potentiometer for a range of 0 to 240000, Speed potentiometer for a range of 0 to 180 and BAL from the inbuilt Alco blow.
2. Data was fetched from this Alco blow, speed logs and the BAL potentiometer into the control unit.

4.1.4.1 Results, Analysis and Conclusion

The tables below show some of the results that were got in the experiment.

Table 4.10: Combined Experiments for Weight, BAL and Speed with values less or equal to the set limits

Weight	Alcohol	Speed	Limit Exceeded	Ticket Issued
56,000	1.3	80	N	N
44,330	1.1	66	N	N
34,743	0.7	15	N	N
14,950	0.2	10	N	N
27,393	0.0	50	N	N

Table 4.11: Combined Experiments for Weight, BAL and Speed with one or more values exceeding the set limits

Weight	Alcohol	Speed	Limit Exceeded	Ticket Issued
56,234	1.2	80	Y	Y
40,000	1.5	85	Y	Y
200,624	1.9	80	Y	Y
16,132	4.3	167	Y	Y
11,567	1.7	180	Y	Y

The results showed that in all the 10 combined experiments done, the device correctly identified if the limits were exceeded or not. The Reason why all the experiments correctly classified the instance is the Control unit was a rule based system. The combined data generated was also correctly logged. This was important to note since it was one of the research aims.
Due to this, the research concluded that the device could correctly verify if the limits were exceeded or not. It also concluded that the device would correctly save all the combined instances generated in future.

4.1.5 Third party integration system Experiments

The main purpose of this experiment was to see if the hardware device could be integrated with a third party system. These objectives were to be achieved in the test;

1. Generate weight values from a Weight potentiometer, speed values from a Speed potentiometer and BAL values from the inbuilt Alco blow.
2. Verify if these values can correctly be retrieved in a third party system.

To achieve these objectives, the below steps were followed:

1. Data was simulated by varying a Weight potentiometer for a range of 0 to 240000, Speed potentiometer for a range of 0 to 180 and BAL from the inbuilt Alco blow.
2. Effects of this data simulation were monitored in PLX-DAQ Excel sheet.

4.1.4.1 Results, Analysis and Conclusion

The diagram below shows some of the results that were got in the experiment.

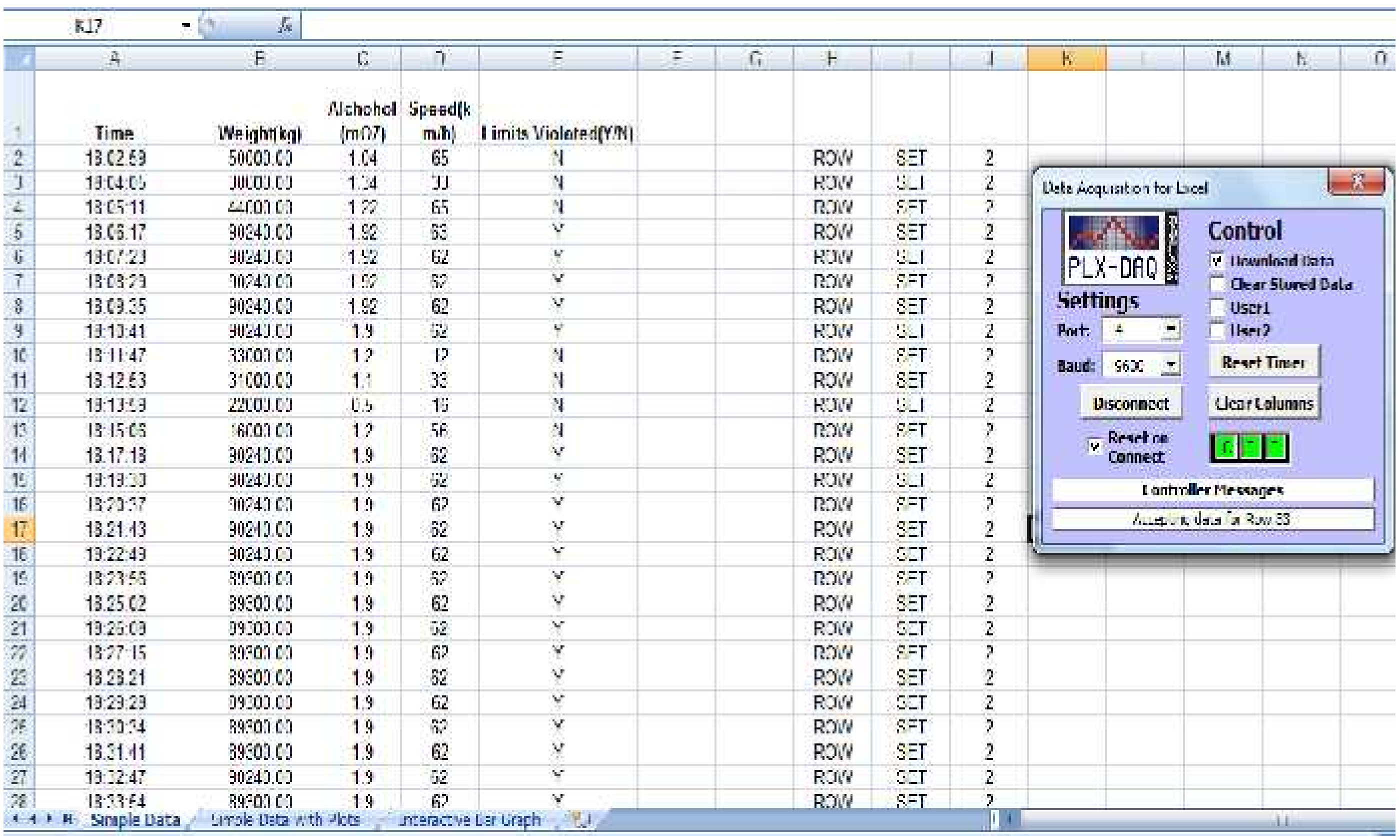

Figure 4.4: Ticketing/Monitoring System

The results showed that in all the 50 third party experiments done, the results were correctly received, logged and monitored in the ticketing/Monitoring system. Due to this, the research concluded that the device could correctly be integrated with a third party system.

4.1.5 Testing Conclusion.

The testing carried out was exhaustive enough to achieve the project testing objectives. The testing showed that the prototype built on the proposed framework could effectively monitor all the three factors together. The project also found that data could be retrieved from a NGSG and weigh bridge as simulation tests showed that the weight electronic data and the speed data from the log files could be retrieved in all the different formats of the files. The project also found that the device could log all the instances of the three factors as required. Lastly, the project found that the developed hardware could be integrated with a third party system in an attempt to improve user experience.

5. CONCLUSION

5.1 Discussion

The main research problem that this research project tried to solve was lack of an integration framework for speed, alcohol and weight monitoring. To solve this problem, the project worked with three objectives. These objectives were; designing an integration framework for speed, blood alcohol level and weight monitoring based on previous research work, developing a hardware prototype based on the model that was designed and simulating some data to test and evaluate the prototype. To achieve these objectives, the research worked with these five research questions; could the research come up with a suitable framework for monitoring speed, alcohol and weight, could the research come up with an embedded hardware device that was able to do the monitoring, could the embedded hardware device be able to retrieve data from a speed limiter, weighbridge and an Alco blow, could the proposed device be able to log and record all the instances of speed, alcohol and weight collected and could the proposed device be integrated with third party systems which are meant to improve the user's experience.

In the first three experiments (Speed Experiment, BAL Experiment, Weight Experiment), the results showed that the hardware prototype could effectively monitor its factor, Retrieve data from a NGSG and weigh bridge and log the instance data as the vehicle moved.

In the combined factors experiment, it was discovered that the hardware device would be able to monitor all the three factors together, and also log the data for the three factors for each instance

In the third party integration system experiment, it was discovered that the hardware prototype was able to be integrated to a third-party system in order to improve the user experience when using the device.

These results answered my entire research questions as was intended. In order to get these results, all the three objectives had been followed. A prototype based on the proposed framework had been developed, that was later tested by some simulated data. All this helped solve the research problem of lack of an integration framework to monitor all the three factors together.
In the literature review, it was seen that the three factors were monitored separately. This research effectively came up with an integration framework that was able to monitor the three

factors concurrently, providing detailed reports on the same. The results provided above helps justify that claim.

Due to these results, it would be in order to use the device on the roads to monitor the three factors. In a scale of 1 to 10, the results were benchmarked to 10 since they were able to effectively test all the possible scenarios as needed in the project.

The research process was similar to previous research discussed in the literature review. For instance in On-board network level fault diagnosis in Automobiles, the research designed a suitable framework also based on previous research work. After designing the framework, the research developed a hardware device that would correctly identify and rectify faults in automobiles. Simulation tests were then done, and the results were either indicating if the vehicle had a fault or not. In Monitoring Physical Threats in a data centre, the research designed a suitable framework also based on previous research work. After designing the framework, the research developed an aggregation of hardware computing components device that would correctly identify a physical threat in a data center. Simulation tests were then done, by introducing sample threats into the data center and the results were either indicating if the threat had been identified or not. In the Two-Stage Diagnosis framework for Wind Turbine Gearbox Condition Monitoring, the research designed a suitable framework also based on previous research work. After designing the framework, the research developed a framework that received time domain data from the rotating blades. The time domain data was processed to frequency domain and later diagnosed in the graphical and analytical hardware components. Simulation tests were then done, by introducing failure in the rotating blades and the results were either indicating if the failure had been identified of not.

5.2 Computing Importance of this research

Before this research, there had been no attempts to come up with a framework that monitors Speed, Blood Alcohol Level and Weights in a PSV vehicle. This research effectively bridged this gap. Initially, over speeding, overloading and drunk driving were monitored separately on the Kenyan roads. Even if there had been many projects dealing on integrating physical factors into one framework, no work had been done on integrating these three factors. This then means that this research added knowledge about monitoring the factors concurrently.

5.3 Common Issues in Integration Frameworks

There are some issues that have been affecting previous research on integration frameworks and they were also occurring in this research. These were:

5.3.1 Data formatting

Data formatting involves combining data that can come with different formats from its sources. It is still a pervasive challenge faced in frameworks that need to query across multiple autonomous and heterogeneous sources. (Bruce Steve, 2012) , (Hopkins George G., 2012). In this research, the speed data came from different types of files residing in the different brands of speed limiters. The files formats that were currently in use as this research was done were .RTFF, .CSV, .TXT, .DOC, .DOCX, .XLSX and .XLX.

In this research, data formatting was achieved by introducing libraries that were able to process speed logs irrespective of the formats in which they were stored. The resultant output was a standard output that could be used by the device. In the results for Speed Experiments for weight values, the research discovered that these libraries effectively processed the data from different logs files with different file formats.

5.3.2 Data Integration

Data integration involves combining data residing in different sources and providing users with a unified view of these data. It is still a pervasive challenge faced in frameworks that need to query across multiple autonomous and heterogeneous sources. (Wikipedia, 2015) , (Halevey, 2012), (Ives Zachary G, 2012). In this research, the speed data came from the different brands of speed limiters, the weight values came from different types of weighbridges used on the roads.

In this research, data integration was achieved by introducing libraries that were able to process the speed and the weight data that came from the speed limiters and the weigh bridges. The resultant output was a standard output that could be used by the device. This output was a list value in the form of, <weight, Alcohol, Speed>.

In the results for combined factors experiment, the research discovered that these libraries effectively processed the data from the three inputs and then presented it in a form that the hardware device could use

5.3.3 System Redundancy

System redundancy is a major issue when it comes to frameworks with multiple sources of inputs. (Younis Mohamed F. Younis , 2007), (Waegli Adrian , 2008), (White Jacob M , 2012). The redundancy introduces a backup system to the framework incase the primary components fail.

This was not solved in this project. The speed and weight inputs were directly retrieved from the speed limiters and the weighbridges. The framework may fail to work correctly if these inputs are incorrect. A redundant system would provide two major functionalities: First, they would introduce a backup plan for retrieving the inputs incase the speed limiters and the weighbridges fail and secondly, a redundant system would verify if the inputs retrieved from the weighbridge and the Alco blows are correct.

5.3.4 Data Integrity

Data integrity refers to maintaining and assuring the accuracy and consistency of data over its entire life-cycle, and is a critical aspect to the design, implementation and usage of any system which stores, processes, or retrieves data. It is a major issue when it comes to frameworks with multiple sources of inputs. (Younis Mohamed F. Younis , 2007), (Waegli Adrian , 2008), (White Jacob M , 2012). A redundant system would have helped the research project achieve data integrity but it was not implemented in the system.

The speed and weight inputs were directly retrieved from the speed limiters and the weighbridges. A redundant system would have verified if the inputs retrieved from the weighbridge and the Alco blows were correct.

5.3.4 Dependability / Reliability

Dependability, or reliability, describes the ability of a system or component to function under stated conditions for a specified period of time. .It is a major issue when it comes to frameworks with multiple sources of inputs. (Edward Jeremy, 2007) , (Bernard Koelf, 2009). An effective framework should always produce the right output for the inputs provided. However, this is not always the case as seen from those two references.

In this research, all the instances were correctly classified as having exceeded the limits or not, as seen from the results .This meant that the framework was dependable as an integration framework for speed, alcohol and weight monitoring

5.4 Summary

The main research problem that this research project was solving was lack of an integration framework for speed, alcohol and weight monitoring. To solve this problem, the project tried to achieve three objectives. These objectives were; designing an integration framework for speed, blood alcohol level and weight monitoring based on previous research work, developing a hardware prototype based on the model that was designed and simulating some data to test and evaluate the prototype. To achieve these objectives, the research worked with these five research questions; could the research come up with a suitable framework for monitoring speed, alcohol and weight, could the research come up with an embedded hardware device that was able to do the monitoring, could the embedded hardware device be able to retrieve data from a speed limiter, weighbridge and an Alco blow, could the proposed device be able to log and record all the instances of speed, alcohol and weight collected and could the proposed device be integrated with third party systems which are meant to improve the user's experience.

In the testing phase, the results showed that the hardware prototype could effectively monitor the three factors together, Retrieve data from a NGSG and weigh bridge and log the instance data as the vehicle moved. It was discovered that the hardware device would be able to monitor all the three factors together, and also log the data for the three factors for each instance. Lastly, it was discovered that the hardware prototype was able to be integrated to a third-party system in order to improve the user experience when using the device.

The research project was therefore able to solve its problem of lack of an integration framework for speed, alcohol and weight monitoring. This is the major contribution in the academic world that the project was trying to solve.

This project also contributed a lot socially; since the device monitored all the three factors at the same time, then it meant that no offence would go unnoticed. Since this hardware product kept logs of the offences and also involved automatic tickets creation, this meant reduce cases of bribing. This was because the evidence was permanently logged in the hardware device and also because an automatic penalty given would now require the driver to follow the due process in orders to clear the fines.

5.5 Suggestion for future work

This hardware device logged all the data for Speed, BAL and weight. It would be worth, if some machine learning would be used on this big data, and try to generate some valuable information out of it.

Secondly, as seen from literature review, there are also other factors that are attributed to the high number of deaths and injuries in RTA like; not using safety belts, not using helmets, not using child constraints, playing loud music in PSV vehicles, poor infrastructure among others. It would be worth to integrate these in the hardware device.

Lastly, a redundant weighing system and speed identification system would be worth investigating. This would be aimed at verifying if the data retrieved from the speed limiters and the weighbridges is correct and also getting the speed and weight values in case the speed limiters or the weigh bridges fail.

6. REFERENCES

A. Kodali, K. P. (2012). A Coupled Factorial Hidden Markov Model (CFHMM) for Diagnosing Coupled Faults. *A Coupled Factorial Hidden Markov Model (CFHMM) for Diagnosing Coupled Faults*.

A. Kodali, S. S. (2012). Dynamic set-covering for real-time multiple fault diagnosis with delayed test outcomes. *Dynamic set-covering for real-time multiple fault diagnosis with delayed test outcomes*.

A. Kodali, Y. Z. (2012). Fault Diagnosis in Cyber-physical Systems: Application to Automotive Electric Power Generation and Storage System (EPGS). *Fault Diagnosis in Cyber-physical Systems: Application to Automotive Electric Power Generation and Storage System (EPGS)*.

X. J. (2013). *A system integration framework for coupled multi Physics simulations.* Beijing.

B. Pattipati, B. B.-S. (2012). An EM Approach for Dynamic Battery Management Systems. *15 th International Conference on Information Fusion.* Singapore.

B. Pattipati, C. S. (2011). System Identification and Estimation Framework for Pivotal Automotive Battery Management System Characteristics. *System Identification and Estimation Framework for Pivotal Automotive Battery Management System Characteristics*, 869-884.

B. Wang, W. W. (2012). Fault Localization using Passive End-to-End Measurements and Sequential Testing for Wireless Networks. *Fault Localization using Passive End-to-End Measurements and Sequential Testing for Wireless Networks*, 439-452.

Breathalizer.net. (2014, june 10). *Using a breathalyzer*. Retrieved from Breathalizer.net: http://www.breathalyzer.net/faq.html

C. Sankavaram, A. K. (2013). An Integrated Health Management Process for Automotive Cyber-Physical Systems. *IEEE ICNC 2013.*

C. Sankavaram, K. P. (2012). Fault Diagnosis and Prognosis in Cyber-Physical Systems with an Application to Hybrid Electric Vehicle Regenerative Braking System. *Fault Diagnosis and Prognosis in Cyber-Physical Systems with an Application to Hybrid Electric Vehicle Regenerative Braking System*.

chaikinandsherman. (2014, April 25). *Causes of Truck Accidents.* Retrieved from chaikinandsherman: http://www.chaikinandsherman.com/Personal-Injury/Truck-Accidents/Causes-of-Truck-Accidents/Overloaded-Trucks.aspx

Christine Cowan, C. G. (2013). *Monitoring Physical Threats in a data centre* .

Daily, B. (2014, june 25). *Transporters calls for tougher laws on overloading*. Retrieved from Business Daily: http://www.businessdailyafrica.com/Corporate-News/Transporters-call-for-tougher-laws-to-check-overloading/-/539550/1005044/-/item/0/-/3w29wf/-/index.html

Dalcom. (2014, May 20). *Omata Speed Gorvenor*. Retrieved from Omata Speed Gorvenor: http://www.dalcomkenya.info/index.html

Ehow. (2014, May 12). *How does Truck Station Weight work*. Retrieved from How does Truck Station Weight work: http://www.ehow.com/how-does_5008826_truckweighing-scales-work.html

Hansen, S. (2012). *A Framework for Diagnosis of Critical Faults in Unmanned Aerial Vehicles.*

HowStuffWorks. (2014, August 20). *How stuff works, Breathalyzers*. Retrieved from HowStuffWorks: http://electronics.howstuffworks.com/gadgets/automotive/breathalyzer4.htm

iwacu. (2014, june 11). *Road Deterioration in Burundi*. Retrieved from iwacu: http://www.iwacu-burundi.org/blogs/english/overloaded-trucks-the-main-cause-of-roads-deterioration-in-burundi/

KEBS. (2013). *The Benchmark 2013.* Nairobi.

KEBS. (2014). Maximum road speed Limiters for Motor Vehicles . *Maximum road speed Limiters for Motor Vehicles* .

KeNHA. (2014). *Approved Speed Governors and Vendors.* Nairobi.

KeNHA. (2014, june 1). *Weighbridges*. Retrieved from Weighbridges: http://www.kenha.co.ke/index.php?option=com_content&view=article&id=41&Itemid=52&limitstart=1

Kleberge, P. (2012). *A Framework for Assessing the security of the Connected Car Infrastructure.*

Limited, C. T. (2010). Analytical Comparative Transport Costs along the Northern Corridor . *Analytical Comparative Transport Costs along the Northern Corridor* .

Limited, D. (2014). Omata Speed Limiter and Recorder. *Omata Speed Limiter and Recorder*.

M.Campbell. (2014). *Parallel Multi-Physics Integration Frameworks for Virtual Rocket Science.* New York.

Manyara, C. G. (2006). *Combating Road Traffic Accidents In Kenya: A Challenge For An Emerging Economy.* Nairobi.

Media, S. (2014, may 25). *Curbing Trucks Overloading*. Retrieved from Standard Media: http://www.standardmedia.co.ke/article/1144030400/new-measures-seek-to-curb-overloading-of-trucks

Media, S. (2014, june 1). *Government re-introduce Alco blow to curb road carnage*. Retrieved from Standard Media: http://www.standardmedia.co.ke/?articleID=2000100233

Nantulya V.M, M.-M. (2009). *Uncovering the Social Determinants of Road Traffic Accidents.* Nairobi.

Nation. (2014, September 17). *New Laws On Technology To Transform PSV Sector.* Retrieved from Nation Media: http://mobile.nation.co.ke/business/New-law-on-technology-to-transform-PSV-sector/-/1950106/2134508/-/format/xhtml/-/5hut8qz/-/index.html

News24. (2014, may 25). *Alco Blow Reintroduction.* Retrieved from News24: http://m.news24.com/kenya/National/News/Alcoblow-re-introduced-to-curb-road-carnage-20131217

Nyberg, M. (1999). *Model Based Fault Diagnosis Methods, Theory, and Automotive Engine Applications.*

Odero, W. (2003). *Road Traffic injuries in Kenya: Magnitude, causes and status and intervention.* Nairobi.

Pattipati, K. (2011). *Fault Diagnosis and Prognosis in a Network of Embedded Systems in Automotive vehicle.* Karachi.

Polioce, K. (2007). *A report on Road Traffic Accidents.* Nairobi.

Razzak J.A, L. S. (1998). *Estimating deaths and injuries due to road traffic accident in Karachi, Pakistan, through the capture-recapture method.* Karachi.

Rosnek. (2013). *Road Safety Network of Kenya.* Nairobi.

S. Zhang, K. P. (2011). Dynamic Coupled Fault Diagnosis with Propagation and Observation Delays. *Dynamic Coupled Fault Diagnosis with Propagation and Observation Delays*.

S. Zhang, K. P. (2012). *ptimal Selection of Imperfect Tests for Fault Detection and Isolation.* London.

Shasharina, S. (2012). *FACETS – A Multi physics parallel Component Framework.* Karachi.

SoftPedia. (2014, July 05). *How a truck Breathalyzers work*. Retrieved from Softpedia: http://news.softpedia.com/news/How-Breathalyzers-Work-89552.shtml

Star, T. (2014, May 12). *Introduction of new Weigh Bridges*. Retrieved from The Star: http://www.the-star.co.ke/news/article-134540/kenya-road-transport-enters-fast-lane-new-weighbridges

Suwatthikul, J. (2008). *Framework and methods for on-board network level fault diagnosis in Automobiles.* Karachi.

Systems, W. (2014, june 1). *Vehicle weighing*. Retrieved from Vehicle weighing: http://www.weighing-systems.com/TechnologyCentre/vehicleweigh.html

Tamilselvan, P. (2013). *A Two-Stage Diagnosis Framework for Wind Turbine Gearbox Condition Monitoring.*

ttcanc. (2014, october 11). *East Africa new Roads regulations*. Retrieved from ttcanc: http://www.ttcanc.org/news.php?newsid=39

WHO. (2007). *World Report on Road Traffic Injury Prevention 2007.* Geneva: Who Press.

WHO. (2012). *World Report on Road Traffic Injury Prevention* . Geneva: WHO Press.

WHO. (2013). *Global Status Report on Road Safety 2013.* Geneva: WHO Press.

WHO. (2014, January 07). *Road traffic injuries. Fact sheet N°358 March 2013.* Retrieved from WHO: http://www.who.int/mediacentre/factsheets/fs358/en/

WHO. (2014, april 15). *World Report on Road Traffic Injuries Prevention, Glossary of Terms.* Retrieved from WHO: http://whqlibdoc.who.int/publications/2004/9241562609.pdf

Wikipedia. (2014, June 16). *Speed Governors*. Retrieved from Wikipedia: http://en.wikipedia.org/wiki/Governor_%28device%29#Public_services_vehicles

Wikipedia. (2014, July 30). *Truck Scale*. Retrieved from Wikipedia: http://en.wikipedia.org/wiki/Truck_scale

Wikipedia. (2014, 11 24). *Weigh in motion*. Retrieved from Wikipedia: http://en.wikipedia.org/wiki/Weigh_in_motion

Works, H. S. (2014, April 11). *How stuff works, Speed Governors*. Retrieved from How stuff works, Speed Governors: http://science.howstuffworks.com/engineering/civil/question626.htm

Zhang, B. (2011). *PHM for Safety Critical Systems: A particle Filtering based system.* London.

7. APPENDICES

APPENDIX I: Approved Speed governors and vendors as at 15th April 2014

APPROVED SPEED GOVERNORS AND VENDORS AS AT 15TH APRIL 2014

S/NO	VENDING COMPANY	SPEED LIMITER(GOVERNOR)	TELEPHONE	IMPORTED GOVERNORS	FITTED AS PER REPORT FORWARDED	IN STOCK
1.	DALCOM ENTERPRISES	OMATA	0721858984	9100	861	0
2.	PINNACLE SYSTEMS LTED	AUTOGRADE	0722375799	1950	649	0
3.	POWER GOVERNORS (K)LTD	PGL CONTRO	0720750363	3800	695	126
4.	POWER GOVERNORS (K) LTD	AUTO CONTRO	0720750363	1000	0	1000
5.	CABLE CAR CORPORATION (K) LTD	SPEEDIK	0725672959	1000	0	1000
6.	EKAS TECHNOLOGIES – NYERI	EKAS	0724315581	521	304	0
7.	SAFERIDER MANAGEMENT SYSTEMS	SAFERIDER SRGLI (HEAVY)	00720941701/0733941701	274	176	98
8.	SIMBA CORPORATION LTD	ELSON TRUCK TECHNOLOGY	070304617	1000	427	20
9.	PAPO HAPO SOLUTION LTD	PHS	0204452666	1500	488	0
10	EUREKA TECHNICAL SERVICE LTD	ROMATIC SYSTEM 880 MICRO+(DMS)	020445155/0720960752	0	0	0
11	MARKMANN & COMPANY LIMITED	KIT KIT	0722457028	200	0	10
12	CONTROL TECH LTD	CONTROL TECH	0703204570/1	50	0	50
13	PAYMATIC	GUARDIAN CL002	0734978746	1000	839	0
14	LAB TECHNOLOGIES INNOVATIONS LTD	TECHTO	0725982293	0	0	0
15	FLEX COMMUNICATIONS LTD	CRUZ	0721413516	0	0	0
16	COMAT TRADING CO. LTD	SAFERIDER SRGLI (HEAVY)	0722754066	150	148	0
17	MVITA UTO DIESEL SYSTEMS	GENTEC MODEL 667	0720976421			
18	KINETIC RESOURCES LTD	TRIDENT	072275066/0722744454			
19	KENSRIN COMPANY LTD	ROSCOE	0724700505			
20	CBET MATATU OWNERS SACCO SOCIETY LTD	5 IN 1 DIGITAL SPPED GOVERNOR	0722339377			
21	CENTRAL VEHICLE SERVICES LTD	CENTRO VS	0724316667			
22	MARTS LOGISTICES LTD	BERKUT MF 2000	0722805382/0724316667			

APPENDIX II: Questionnaire

SECTION A: Company Details

1. Name of the Company……………………………………

2. Location……………………………………

3. Products offered (Tick)
   - o Speed Limiter
   - o Weigh Bridge

SECTION B: Products Details

4. List the inputs of the gadget
   -
   -
   -
   -
5. Explain briefly the process the data undergoes in the gadget

6. List the output (s) of the gadgets.

    -
    -
    -
    -
    -

7. List the data formats of the output (s)

    -
    -
    -
    -
    -
    -

APPENDIX III: Hardware Libraries

Alcohol Weight Processor

Alcohol Weight Processor Main Class

```
/*
AlchoholWeightProcessor.cpp - Library for Processing Weight and Alcohol.
Created by Sam Karuga, September 18, 2014.
Released into the public domain.
*/

#include "Arduino.h"
#include "AlchoholWeightProcessor.h"

AlchoholWeightProcessor::AlchoholWeightProcessor(int csPin,File file,int alcoPin, int
weighPin,int speedLimit,long weightLimit,float alchoholLimit,int ledRed,int ledGreen,int
ledBlue)
{
Serial.begin(9600);

_CSPIN=csPin;
_FILE=file;

_ALCOPIN = alcoPin;
_WEIGHPIN=weighPin;

_LEDRED=ledRed;
_LEDGREEN=ledGreen;
```

```
_LEDBLUE=ledBlue;

pinMode(_LEDRED,OUTPUT);
pinMode(_LEDGREEN,OUTPUT);
pinMode(_LEDBLUE,OUTPUT);


//PLX_DAQ Serial.println(F("Limits initialization"));
_SPEEDLIMIT=speedLimit;
_WEIGHTLIMIT=weightLimit;
_ALCHOHOLLIMIT=alchoholLimit;


//PLX_DAQ Serial.println(_SPEEDLIMIT);
//PLX_DAQ Serial.println(_WEIGHTLIMIT);
//PLX_DAQ Serial.println(_ALCHOHOLLIMIT);


/*

WEIGHTS
-----------------

0-240000, but limit is 5600 kg. 0-240000 maps to 0-1023
this means one unit in 0-1023, maps to 240000/1023 kg= 235kg.
since 0-255 is 0-1023 /4 then to convert the value we get in 0-255 /X/, then
X*235*4 give me the value in kg.

Alcohol
-----------------
```

0-5 MILLI OZ, but limit is 2 milli oz. 0-5 maps to 0-1023. The system will work with micro oz to avoid floats ie 0-5000 micro oz
this means one unit in 0-1023, maps to 5000/1023 kg= 5micro oz.
since 0-255 is 0-1023 /4 then to convert the value we get in 0-255 /X/, then
X*5*4 give me the value in milli oz. To convert to micro oz, then X*5*4/1000==X*0.005*4

```
*/
_SPEED=0;

}


/*
This function Writes weight and alcohol data, got from the sensors into an SD Card. It also records a value got from NGSG
*/
void AlchoholWeightProcessor::writeAWData(int ngsgSpeed)
{
//PLX_DAQ Serial.println(F("Speed Got from NGSG is: "));
//PLX_DAQ Serial.println(ngsgSpeed);
lightLed(0,1,0);




//PLX_DAQ Serial.println(F("Getting data from sensors"));
lightLed(0,1,0);
```

```
long weight=0;
int alcohol=0;

weight=analogRead(_WEIGHPIN);
alcohol=analogRead(_ALCOPIN);



weight=constrain(weight,0,1023);// constrain weighbridge values
alcohol=constrain(alcohol,0,1023);// constrain alcohol values

weight=map(weight,0,1023,0,255);// map weighbridge values
alcohol=map(alcohol,0,1023,0,255);// map alcohol values

_SPEED=ngsgSpeed;
_WEIGHT=weight*235*4;
_ALCHOHOL=alcohol*0.005*4;



//PLX_DAQ Serial.print(F("Weight value is: "));
//PLX_DAQ Serial.println(weight*235*4);
//PLX_DAQ Serial.print(F("Alcohol value is: "));
//PLX_DAQ Serial.println(alcohol*0.005*4);




//trigger smart module and check if laws have been broken. If yes, send an SMS inside it
**SMART MODULE ALSO TRIGGERS THE ACTUATOR -GSM
//laws are loaded from a file.
lightLed(0,0,1);
int limitExceeded=0;
```

```
int smsSent=0;



// SmartComponent smart(_SPEED,weight,alcohol,_LEDRED,_LEDGREEN,_LEDBLUE);
int smartInstance=smartFunc();

//PLX_DAQ Serial.print(F("Smart Instance returned is:  "));
//PLX_DAQ Serial.println(smartInstance);





if(smartInstance==0){
limitExceeded=0;
smsSent=0;
lightLed(0,0,1);
}

if(smartInstance==1){
limitExceeded=1;
smsSent=1;
lightLed(1,0,0);
}

if(smartInstance==2){
limitExceeded=1;
smsSent=0;
lightLed(1,0,0);
}
```

```
//PLX_DAQ Serial.println(F("Initializing SD card for Writings speed , weight and alcohol"));
lightLed(0,1,0);
if(!SD.begin(_CSPIN)){

//  Serial.println(F("Initializing SD card failed!"));
lightLed(1,0,0);
// return;
}



//PLX_DAQ Serial.println(F("Initializing SD card successfully Done!"));
lightLed(0,0,1);
_FILE=SD.open("WA_LOGS.txt",FILE_WRITE);  //8.3 format



//PLX_DAQ Serial.println(F("Writing data into SD Card"));
lightLed(0,1,0);
_FILE.print(weight*235*4);
_FILE.print(",");
_FILE.print(alcohol*0.005*4);
_FILE.print(",");
_FILE.print(_SPEED);
_FILE.print(",");
_FILE.print(limitExceeded==0?"N":"Y"); // SMART MODULE RETURNS A VALUE IF
RULES HAD BEEN BROKEN OR NOT
_FILE.print(",");
_FILE.println(smsSent==1?"Y":"N"); // SMART MODULE RETURNS A VALUE IF SMS
HAVE BEEN SENT OR NOT
_FILE.close();
```

```
Serial.println(F("Writing data into SD Card Completed Successfully"));
lightLed(0,0,1);

//Write to integration system PLX_DAQ
Serial.print("DATA,TIME,"); Serial.print(weight*235*4); Serial.print(",");
Serial.print(alcohol*0.005*4); Serial.print(",");Serial.print(_SPEED);
Serial.print(",");Serial.print(limitExceeded==0?"N":"Y"); Serial.print(","); Serial.print("");
Serial.print(","); Serial.print(""); Serial.print(","); Serial.print("");
Serial.println("ROW,SET,2");
}

/*
This function Reads weight and alcohol data from an SD Card
*/
void AlchoholWeightProcessor::readAWData()
{


//PLX_DAQ  Serial.println(F("Initializing SD card For Reading weight and alcohol data.--We
also include the speed logs"));
lightLed(0,1,0);
if(!SD.begin(_CSPIN)){

//Serial.println(F("Initializing SD card failed!"));
lightLed(1,0,0);
// return;


//PLX_DAQ Serial.println(F("Initializing SD card successfully Done!"));
lightLed(0,0,1);
```

```
_FILE=SD.open("WA_LOGS.txt");   //8.3 format


//PLX_DAQ Serial.println(F("Reading data from SD Card"));
lightLed(0,1,0);

if(_FILE){
//PLX_DAQ Serial.println(F("The data read from file is: "));
lightLed(0,1,0);
while(_FILE.available()){

byte value=_FILE.read();

Serial.write(value);

}
//PLX_DAQ Serial.println(F(""));
//PLX_DAQ Serial.println(F("Reading File data Completed Successfully!"));
lightLed(0,0,1);
}

else
{

//PLX_DAQ Serial.println(F("File Not present"));
lightLed(1,0,0);

}

_FILE.close();
```

```
}

}
/*

SmartComponent smart(_SPEED,weight,alcohol,_LEDRED,_LEDGREEN,_LEDBLUE);
int smartInstance= smart.smartFunc();

NOT USED DUE TO MEMORY FAILURE WHEN I LOAD ANOTHER LIBRARY. TO BE
USED IN CASE I ADD MORE RAM
*/
int AlchoholWeightProcessor::smartFunc()
{

//PLX_DAQ Serial.println(F("Smart Component initialization"));
lightLed(0,1,0);
int speedCase=0;
int weightCase=0;
int alchoholCase=0;
int smartInstance=0;


if(_SPEED>_SPEEDLIMIT){
speedCase=1;
}

if(_WEIGHT>_WEIGHTLIMIT){
weightCase=1;
}
```

```
if(_ALCHOHOL>_ALCHOHOLLIMIT){
alchoholCase=1;
}

// SPEED
switch(speedCase){
case 1:
{
//PLX_DAQ Serial.println(F("Speed Violated"));
smartInstance=1;
lightLed(1,0,0);

break;
}

default:
{
//PLX_DAQ Serial.println(F("Speed Not Violated"));
lightLed(0,0,1);
}

}
//WEIGHT

switch(weightCase){

case 1:
{
//PLX_DAQ Serial.println(F("Weight Violated"));
```

```
smartInstance=1;
lightLed(1,0,0);
break;
}

default:

{
//PLX_DAQ Serial.println(F("Weight Not Violated"));
lightLed(0,0,1);
}

}

//ALCHOHOL
switch(alchoholCase)
{

case 1:
{
//PLX_DAQ Serial.println(F("Alcohol  Violated"));
smartInstance=1;
lightLed(1,0,0);
break;
}

default:

{
//PLX_DAQ Serial.println(F("Alcohol Not  Violated"));
```

```
lightLed(0,0,1);
}

}

if (speedCase==1 ||weightCase==1 ||alchoholCase==1 ){

//PLX_DAQ Serial.println(F("Sending SMS Alert"));
lightLed(0,1,0);
//SEND SMS.

/*if smsSent(){



}else{

Serial.println(F(" SMS Alert Not Sent"));
smartInstance=2;
lightLed(0,0,1);

}*/



}
return smartInstance;

}

/*
This function is intended for lighting LED
```

```
*/
void AlchoholWeightProcessor::lightLed(int redStatus,int greenStatus, int blueStatus)
{

digitalWrite(_LEDRED,redStatus==1?HIGH:LOW);
digitalWrite(_LEDGREEN,greenStatus==1?HIGH:LOW);
digitalWrite(_LEDBLUE,blueStatus==1?HIGH:LOW);
delay(2000);
return;
}
```

Alcohol Weight Processor Header Class

```
/*
  AlchoholWeightProcessor.cpp - Library for Processing Weight and Alcohol.
  Created by Sam Karuga, September 18, 2014.
  Released into the public domain.
*/


#ifndef AlchoholWeightProcessor_h
#define AlchoholWeightProcessor_h

#include "Arduino.h"
#include "SD.h"
#include "SPI.h"



class AlchoholWeightProcessor
{
```

```
  public:
    AlchoholWeightProcessor(int csPin,File file,int alcoPin, int weighPin,int speedLimit,long weightLimit,float alchoholLimit,int ledRed,int ledGreen,int ledBlue);
    void writeAWData(int ngsgSpeed);
            void readAWData();
            int smartFunc();
void lightLed(int redStatus,int greenStatus, int blueStatus);
private:
    int _CSPIN;
private:
    File _FILE;
private:
volatile int _ALCOPIN;
private:
volatile int _WEIGHPIN;
private:
volatile int _LEDRED;
private:
volatile int _LEDGREEN;
private:
volatile int _LEDBLUE;
private:
     int _SPEED;
private:
     long _WEIGHT;
private:
     float _ALCHOHOL;
private:
     int _SPEEDLIMIT;
private:
```

```
    long _WEIGHTLIMIT;
private:
    float _ALCHOHOLLIMIT;

/*# define _SPEEDLIMIT
# define _WEIGHTLIMIT
# define _ALCHOHOLLIMIT          */

};

#endif
```

Config

Config Header Class

```
/*
  Config.h - Configuration file that holds all the data used in the application.
  Created by Sam Karuga, September 18, 2014.
  Released into the public domain.
*/

#ifndef Config_h
#define Config_h

#include "Arduino.h"

class Config
{

# define _MAXSPEEDCONF 180
```

```
# define  _SPEEDLIMITCONF 80

# define  _WEIGHTLIMITCONF 56000

# define  _ALCHOHOLLIMITCONF 1.5

# define  _DATACOLLECTIONTIMECONF 1
};

#endif
```

Smart Component

Smart Component Main Class

```
/*
  SmartComponent.h - Smart Library for Processing Speed , Alcohol and weight by use of AI models..For now, we use simple case based.
  Created by Sam Karuga, September 18, 2014.
  Released into the public domain.
*/

#ifndef SmartComponent_h
#define SmartComponent_h

#include "Arduino.h"
#include "SD.h"
#include "SPI.h"
```

```
class SmartComponent
{
  public:
    SmartComponent(int speed,int weight,int alchohol,int ledRed,int ledGreen,int ledBlue);
            int smartFunc();
            void getLimits();
            void lightLed(int redStatus,int greenStatus, int blueStatus);
private:
    volatile int _CSPIN;
private:
     File _FILE;
private:
    volatile int _LEDRED;
private:
    volatile int _LEDGREEN;
private:
    volatile int _LEDBLUE;
private:
     int _SPEED;
private:
     int _WEIGHT;
private:
     int _ALCHOHOL;
private:
     int _SPEEDLIMIT;
private:
     int _WEIGHTLIMIT;
private:
     int _ALCHOHOLLIMIT;
```

```
};

#endif
```

Smart Component Header Class

```
/*
  SmartComponent.h - Smart Library for Processing Speed , Alcohol and weight by use of AI models..For now, we use simple case based.
  Created by Sam Karuga, September 18, 2014.
  Released into the public domain.
*/

#ifndef SmartComponent_h
#define SmartComponent_h

#include "Arduino.h"
#include "SD.h"
#include "SPI.h"



class SmartComponent
{
  public:
    SmartComponent(int speed,int weight,int alchohol,int ledRed,int ledGreen,int ledBlue);
        int smartFunc();
        void getLimits();
        void lightLed(int redStatus,int greenStatus, int blueStatus);
private:
    volatile int _CSPIN;
```

```
private:
     File _FILE;
private:
    volatile int _LEDRED;
private:
    volatile int _LEDGREEN;
private:
    volatile int _LEDBLUE;
private:
     int _SPEED;
private:
     int _WEIGHT;
private:
     int _ALCHOHOL;
private:
     int _SPEEDLIMIT;
private:
     int _WEIGHTLIMIT;
private:
     int _ALCHOHOLLIMIT;
};

#endif
```

Speed Processor

Speed Processor Main Class

```
/*
```

```
SpeedProcessor.cpp - Library for Processing Speed logs from a New Generation Speed governor, NGSG.
Created by Sam Karuga, September 18, 2014.
Released into the public domain.
*/

#include "Arduino.h"
#include "SpeedProcessor.h"



SpeedProcessor::SpeedProcessor(int csPin, File file,int maxSpeed,int ledRed,int ledGreen,int ledBlue)
{
 Serial.begin(9600);
_CSPIN = csPin;
_FILE=file;
_LEDRED=ledRed;
_LEDGREEN=ledGreen;
_LEDBLUE=ledBlue;
_MAXSPEED=maxSpeed; //maximum speed a car can attain. for matatus, its 180

pinMode(_LEDRED,OUTPUT);
pinMode(_LEDGREEN,OUTPUT);
pinMode(_LEDBLUE,OUTPUT);

}



/*
This function Reads speed data logged in a NGSG.
```

```
*/
int SpeedProcessor::readSpeedData()
{
int ngsgSpeed=0;

//PLX_DAQ Serial.println(F("Initializing SD card For Reading Speed "));
 lightLed(0,1,0);
if(!SD.begin(_CSPIN)){

// Serial.println(F("Initializing SD card failed!"));
 lightLed(1,0,0);
 // return;
}

//PLX_DAQ Serial.println(F("Initializing SD card successfully Done!"));
 lightLed(0,0,1);
_FILE=SD.open("SPEED_LG.txt",FILE_READ);  //8.3 format

//PLX_DAQ Serial.println(F("Reading Speed data from SD Card"));
 lightLed(0,1,0);

if(_FILE){
//PLX_DAQ Serial.println(F("The Speed data read from file is: "));
 lightLed(0,1,0);



 while(_FILE.available()){

byte value=_FILE.read();
```

```
Serial.write(value);

}

//PLX_DAQ Serial.println(F(""));
//PLX_DAQ Serial.println(F("Reading Speed data Completed Successfully!"));
lightLed(0,0,1);
}

else
{

//PLX_DAQ Serial.println(F("File Not present"));
 lightLed(1,0,0);

}

_FILE.close();
return 0;
}

/*
This function write simulated speed data into a text file
*/
int SpeedProcessor::writeSpeedData(int speedPin)
{

int ngsgSpeed=analogRead(speedPin);
ngsgSpeed=constrain(ngsgSpeed,0,1023);
```

```
ngsgSpeed=map(ngsgSpeed,0,1023,0,_MAXSPEED);

//PLX_DAQ Serial.println(F("Initializing SD card For Writing  Speed"));
 lightLed(0,1,0);
if(!SD.begin(_CSPIN)){

// Serial.println(F("Initializing SD card failed!"));
 lightLed(1,0,0);
 // return;
}



//PLX_DAQ Serial.println(F("Initializing SD card successfully Done!"));
lightLed(0,0,1);
SD.remove("SPEED_LG.txt");
_FILE=SD.open("SPEED_LG.txt",FILE_WRITE);  //8.3 format

//PLX_DAQ Serial.println(F("Writing speed data into SD Card"));
 lightLed(0,1,0);
_FILE.print(ngsgSpeed);
_FILE.close();
//PLX_DAQ Serial.print(F("Writing speed data into SD Card Completed Successfully. The
speed written is: "));
//PLX_DAQ Serial.println(ngsgSpeed);
 lightLed(0,0,1);
 return ngsgSpeed;
}
/*
This function is intended for lighting LED
*/
```

```
void SpeedProcessor::lightLed(int redStatus,int greenStatus, int blueStatus)
{

digitalWrite(_LEDRED,redStatus==1?HIGH:LOW);
digitalWrite(_LEDGREEN,greenStatus==1?HIGH:LOW);
digitalWrite(_LEDBLUE,blueStatus==1?HIGH:LOW);
delay(2000);

}
```

Speed Processor Header Class

```
/*
  SpeedProcessor.h - Library for Processing Speed logs from a New Generation Speed governor.
  Created by Sam Karuga, September 18, 2014.
  Released into the public domain.
*/

#ifndef SpeedProcessor_h
#define SpeedProcessor_h

#include "Arduino.h"
#include "SD.h"
#include "SPI.h"

class SpeedProcessor
{
  public:
    SpeedProcessor(int pin,File file,int ledRed,int maxSpeed,int ledGreen,int ledBlue);
        int readSpeedData();
        int writeSpeedData(int speedPin);
```

```
            void lightLed(int redStatus,int greenStatus, int blueStatus);
private:
    volatile int _CSPIN;
private:
     File _FILE;
private:
    volatile int _LEDRED;
private:
    volatile int _LEDGREEN;
private:
    volatile int _LEDBLUE;
private:
    volatile int _MAXSPEED;

            //# define  _MAXSPEED;

};

#endif
```

APPENDIX IV: Sample Main Code

```
#include <Config.h>
#include <AlchoholWeightProcessor.h>
#include <SpeedProcessor.h>
#include <SD.h>
#include <SPI.h>

File spFile,awFile;
/*
pins to be used
RX,TX--> 0,1
```

```
Interupts--> 2,3 [0,1]
Common Carthode alert Led-->4(blue),cathode,5(red),6(green)
SD Library-->10
Alchohol-->A0
WeighBridge-->A1
Speed -->A2
*/

const  int _CSPIN=10;
const  int _ALCHOHOLPIN=A0;
const  int _WEIGHTPIN=A1;
const  int _SPEEDPIN=A2;

const  int _LEDRED=5;
const  int _LEDGREEN=6;
const  int _LEDBLUE=4;
volatile int v_speed=0;

//Read Configuration Data
const  int _MAXSPEED = _MAXSPEEDCONF;
const  int _SPEEDLIMIT = _SPEEDLIMITCONF;
const  long _WEIGHTLIMIT = _WEIGHTLIMITCONF;
const  float _ALCHOHOLLIMIT = _ALCHOHOLLIMITCONF;
int _DATACOLLECTIONTIME = _DATACOLLECTIONTIMECONF;
SpeedProcessor
speedProcessorRead(_CSPIN,spFile,_MAXSPEED,_LEDRED,_LEDGREEN,_LEDBLUE);
AlchoholWeightProcessor
awProcessor(_CSPIN,awFile,_ALCHOHOLPIN,_WEIGHTPIN,_SPEEDLIMIT,_WEIGHTLIM
IT,_ALCHOHOLLIMIT,_LEDRED,_LEDGREEN,_LEDBLUE);
```

```
void setup(){
Serial.begin(9600);
//intergration system code
Serial.println("CLEARDATA");
Serial.println("LABEL,Time,Weight(kg),Alchohol(mOZ),Speed(km/h),Limits Violated(Y/N)");
}

void loop(){

v_speed=speedProcessorRead.writeSpeedData(_SPEEDPIN); //Write speed data. Since am always writing one speed, i can return that speed and use it until the next session ends
speedProcessorRead.readSpeedData(); //get speed data v_speed to get data here when i figure how to convert the bytes into an int
awProcessor.writeAWData(v_speed); //write speed, alchohol and weight data
awProcessor.readAWData(); // read the saved data for validation
delay(_DATACOLLECTIONTIME*1000U);

}
```