PROTOTYPING S.M.A.R.T. VEHICLE NETWORKING PROTOCOL

FIRST REVIEW REPORT

B.Tech Mechanical Engineering

by
NIKHIL PANDITA

[14BME0133]

School of Mechanical Engineering

JANUARY, 2018

Project ID	Winter2018/SMEC/B.Tech/Mechanical/ **17DES0055**
Date of Review	**24.01.2018**
VIT Guide	Prof. **Kalaiarassan G.**

<table>
<tr><td>External Guide</td><td>Name:
Designation:
Mobile:
Email:
Business Unit:</td></tr>
<tr><td>Project Team Members</td><td><u>Student 1</u>
Name: Nikhil Pandita
Reg. No: 14BME0133
Email:
Mobile:
<u>Student</u> 2
Name:
Reg. No:
Email:
Mobile:
<u>Student</u> 3
Name:
Reg. No:
Email:
Mobile:</td></tr>
<tr><td>Guide’s Remarks</td><td></td></tr>
<tr><td>Name and Signature of the guide</td><td></td></tr>
<tr><td>Comments of Reviewer(s)</td><td></td></tr>
<tr><td>Name and Signature of the Reviewer</td><td></td></tr>
</table>

Table of Contents

Abstract.

A novel approach to tackle various inefficiencies of the modern day Vehicle-to-Vehicle communication technology, specifically the modern-day implementation using the Automotive-Grade Linux. The paper begins with sampling the actual hardware and software deployed by the leading manufacturers and industry, highlighting use-cases like the Toyota Prius, Tesla Model S, Reva, etc,

Keywords:

Automotive Communication Networks;

Decentralized Communications;

In-Vehicle Networking;

Hybrid Platooning;

S.M.A.R.T. Automobile Clustering

CHAPTER –I

INTRODUCTION & LITERATURE REVIEW

Introduction

Nowadays , the culture of hybrid, all-electronic S.M.A.R.T. and connected autonomous vehicles is on an ever-peaking demand-curve. This also means an extension of the vehicle-security exploitaions increment we hear about through daily media, about theft, hijacking or simply vandallism. Such an overwhelming need for an automobiles' security and longevity can only be met by the far-reaching, impactful and tailored technology, suited for the respective scenario. Realizing such endeavours could be only possible owing to the O.S.S. collective, and thence garnered resources and source codes. In account of solving this research conquest, such an approach has been applied, in order to be able to cater the needs of almost everyone with a direct contact with a vehicle, or any automobile, public, personal or even private. We, through the medium of this project, would aspire to address such vehicle optimization adversing security related tradeoffs, and conclusively suggest remedies.

Literature Review

List of Sampled Technology Protocols

- 1: Contoller Area Network (C.A.N.)
- 2: MAC layer Addressing
- 3: IEEE 802.11 (b.g.n/a/c)

List of Sampled Technology Hardware

- 1: Adafruit's Arduino UNO
- 2: Raspberry Pi's 'Model B+'
- 3: Reannaisance's PorterBoard 2
- 4: Orange Pi's IoT+ Embedded
- 5: Stock Daragonboard410c (QEMU Emulated VM)

List of Sampled Software Releases

- 1: Automotive Grade Linux (A.G.L., Linux Kernel 3.9)
- 2: Tyzen Operating System (UNIX Kernel 4.11)
- 3: Qt (For the Graphical Release, Applications)
- 4: Ubuntu IoT Core (+2.3.26)

List of Sampled Libraries and Modules:

- 1: AGL
  - 1.1: agl-demo
  - 1.2: agl-appfw-smack
  - 1.3: agl-devel
  - 1.4: agl-netboot

- 2: UNIX
  - 2.1: gawk

- 2.2: wget
- 2.3: git-core
- 2.4: diffstat
- 2.5: texinfo
- 2.6: chrpath
- 2.7: cpip
- 2.8: socat

- 3: libdll
  - 3.1: libsdl.2-dev
  - 3.2: gcc-multilib
  - 3.3: libhvac

Knowledge gained from the literature:

List of Sampled Authoring Softwares:

- 1: Reading/Writing a PCB (.gerber format)
- 2: C, PyPi (Writing Functional Code Snippets)
- 3: make, build (C-lang)
- 4: bash (UNIX Scripting)

Gaps in the Literature

Prologue:

Most of these projects use the standard IEEE 802.11 for communication. But also GSM, UMTS, GPRS protocols are used in some of these projects. Generally WSNs (Wireless Sensor Networks) are deployed uneffectively and thus "platoonong" is inefficient, since the convuluted network is not 'big' enough in terms of the 'no. of nodes' present in the V2V (Vehicle-to-Vehicle) or V2I(Vehicle-to-Infrastructure) network.

Traditionally, IEE standards like the Basic layers have been employed tor the information transmission. Routing inside a low power area network (LoWPAN) might be considered a challenge, as the RPL has to work over lossy radio links, with battery-powered nodes, multihop mesh topologies and frecuent topology changes.

To give a solution several working groups are giving support to the RFC's for this protocol. One of them is the routing over lowpan and lossy networks (ROLL) who is in charge of routing tasks.

Meanwhile the "6LoWPAN" is trying to bring the new IPv6 addressing system to these resource-constrained devices. We try to predict the points of failure in such technology from a penetration and security testing point of view and conlclude with instantly applicable remedies via pull requests to the FOSS code repositories.

MAC/PHY layer based communication:

Radio waves and infrared have been studied to give medium support to IVCs. The radio waves include micro, millimeter and VHF waves. The communication with millimeter waves and infrared are usually directional, while VHF is used for broadcast. The typical radio bandwidth used in IVC is 5.9 GHz in US, 5.8 GHz in Japan and 5.8 GHz in Europe. The FleetNet project chose ULTRA TDD due to the availability of the unlicensed frequency band 2010-2020 MHz in Europe. Most projects, however, have adopted the use of infrared (CarTALK, COOPER, JSK, PATH…).

There are two approaches in developing MAC for IVCs. One is using IEEE 802.11 as a radio interface, while the other consists on extended 3G technology, such as CDMA for distributed access.

Both of them have to be modified and adapted to provide an efficient solution for IVCs.

The advantage of using IEEE 802.11 is the inherited support for distributed coordination in ad hoc mode. On the other hand, 3G extensions present high granularity for data transmission.

Network layer based communication:

Almost all routing protocols used by the different IVC projects are position-based. In addition, existing MAC ad hoc protocols could be directly applied. But if an optimal performance is desired taking into account the linear nature of the networks seen in section III, modification of the existing routing protocols must be performed. In addition, the features most of vehicles offer nowadays makes possible to get position information via GPS or GIS, very useful for routing. The protocol uses a forwarding scheme which avoids beacons for impactful transmission and effective sensing.

Objectives:

- : To come up with a neat-networking protocol schema to address inefficient hop-on /ad-hoc communication propogation delays, possibly trying to implement in a decentralized contract.
- : To be able to successfully reproduce the hardware based real-time implementation of the AGL release on an ARM based development board.
- : To provide an future roadmap for Non-hybrid cum Hybrid on-road network integration in a cheap (feasible), environment-friendly (sustainanble) and energy-efficient (if not, utilitarian) by means of a snap-on dashboard powered by a simple smartphones' sensors, transmitters and transduecers.
- : To demonstrate a successful implementation of AGL (improvised fork) during the final review.

Design Elements included :

Engineering Standards*	*Prototype and Fabrication*
Design Analysis*	*Experimentation*
Modelling and Simulation	*Software Development*

Realistic Constraints to be addressed :

Economic	*Ethical*
Environmental	*Health and Safety*
Social	*Manufacturability*
Political	*Sustainability*

CHAPTER –II

METHODOLOGY AND EXPERIMENTAL PROCEDURE

2.1 Methodology

The AGL is first ported to a VM with a usual DebianOS base kernel.

Next, the images are downloaded and flshed onto a SDHCeMMC Memory Card.

Third, the auxiliary input and output peripherals are serially connected to the used Raspberry Pi, or UNO module.

2.2 Experimental Procedure

Hardware Requirements

- Dragonboard410c
- 96Boards Compliant Power Supply
- Linksprite 96Boards Touch Screen
- Sensors Mezzanine
- Audio Mezzanine(Required if using External Arduino)
- Arduino Uno(Optional)
- DC motor with Propellers
- L298 Motor Driver

- 5mm LED's
- 330 ohm resistors
- Connecting wires

Arduino

Controlling Fan Speed and LED intensity are handled by the Arduino. Sensors Mezzanine has an ATMega328 microcontroller comaptible with Arduino Uno. We use that or any external Arduino Uno for PWM control.

In case of using Sensors Mezzanine, the sketch can be uploaded by using Dragonboard410c itself..

If using Sensors Mezzanine, please follow the below steps on Dragonboard410c running Debian otherwise use Arduino IDE on the host system for programming.

$ cd ~/Documents

$ git clone https://github.com/96boards-projects/agl-demo.git

$ cd agl-demo/arduino/hvac

Now open the hvac.ino using Arduino IDE and flash it onto the Sensors Mezzanine or Arduino Uno.

Dragonboard410c

Execution environment: Host PC

Software Dependencies:

*$ sudo apt-get install gawk wget git-core diffstat unzip texinfo gcc-multilib *

build-essential chrpath socat libsdl1.2-dev xterm cpio curl

Downloading AGL Source Code

AGL uses repo tool for maintaining repositories. We need to download the source on the host machine and cross compile it for Dragonboard410c.

$ export AGL_TOP=$HOME/workspace_agl

$ mkdir -p $AGL_TOP

$ mkdir -p ~/bin

$ export PATH=~/bin:$PATH

$ curl https://storage.googleapis.com/git-repo-downloads/repo > ~/bin/repo

$ chmod a+x ~/bin/repo

Next, download the stable branch of AGL.

$ cd $AGL_TOP

$ repo init -b dab -m dab_4.0.2.xml -u https://gerrit.automotivelinux.org/gerrit/AGL/AGL-repo

$ repo sync

Building AGL

Now, to build the agl-demo-platform for Dragonboard410c.

$ source meta-agl/scripts/aglsetup.sh -m dragonboard-410c agl-demo agl-appfw-smack agl-devel agl-netboot

Now to move to the directory where we have cloned the agl-demo project from 96Boards Projects Org.

$ cd agl-demo

Copying the custom HVAC recipie to AGL source

$ cp hvac_git.bb $(AGL_TOP)/meta-agl-demo/recipes-demo-hmi/hvac/hvac_git.bb

Executing bitbake command by moving to the build directory of AGL source.

$ cd $(AGL_TOP)/build

$ bitbake agl-demo-platform

Flashing AGL onto Dragonboard410c

Once the build has been completed, we have to flash the boot and rootfs images onto Dragonboard410c. Now, boot Dragonboard into fastboot mode by following the instructions here. Then follow the below instructions to flash AGL onto Dragonboard410c.

$ cd $AGL_TOP/build/tmp/deploy/images/dragonboard-410c

$ sudo fastboot flash boot boot-dragonboard-410c.img

$ sudo fastboot flash rootfs agl-demo-platform-dragonboard-410c.ext4

Hardware Setup to execute HVAC demo.

Make sure the Dragonboard410c is powered off

Connect DC motor and LEDs to Arduino as per above schematic

Connect LCD to Dragonboard410c via HDMI cable for display and Micro USB cable for touch input

Power on your 96Boards CE with compatible power supply

Dragonboard410c should now boot into AGL and homescreen should be visible.

HVAC Utilities

Execution environment: Dragonboard410c

Navigate to the HVAC application from the Homescreen.

1. To control the Fan speed, change the position of the slider at the top.

2. To control the LED intensities, change the values of L/R temperatures by dragging up the LO box.

3. Turn off power by using the following:

$ sudo cd ..

$ poweroff --no-latch

CHAPTER –III

RESULTS AND DISCUSSION

PHASE I

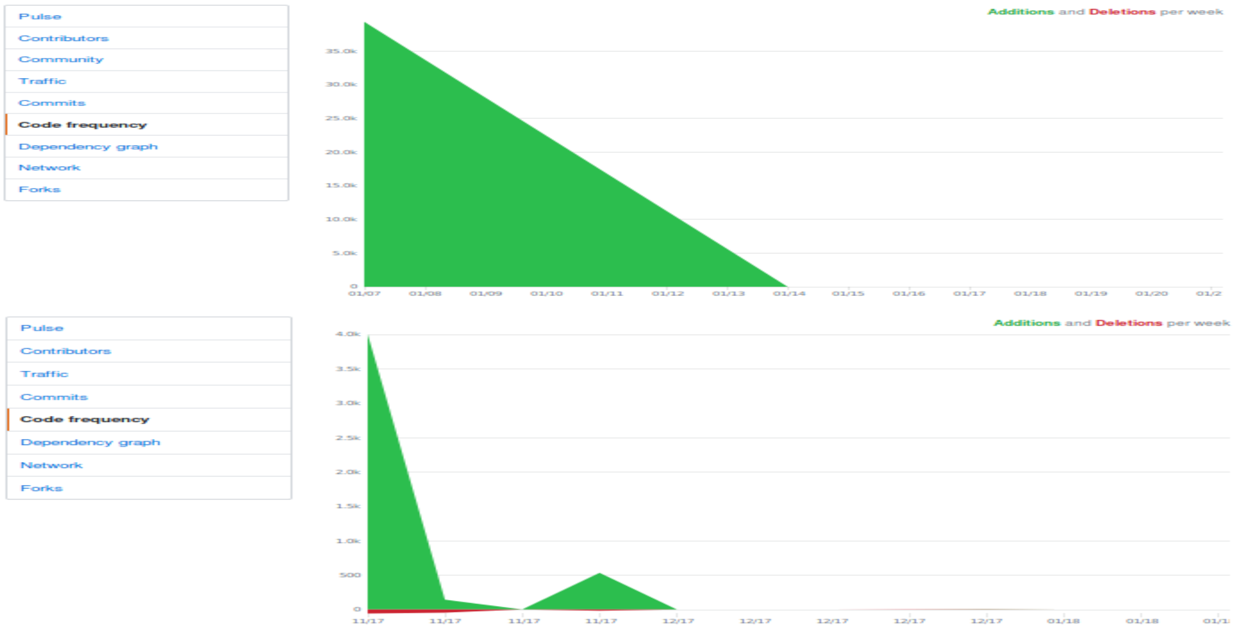
3.1 Work done so far:

- Successfully studied the architecture of a PVB (printed) board.
- Gained a deep understanding of remote-sensing and GIS in application-layer deployment.
- Ported AGL unto raspberryPi and successfully emulated on a HDMI-connected monitor.
- Sent a Pull Request that was successfully merged wih the source at git.automotivelinux.com

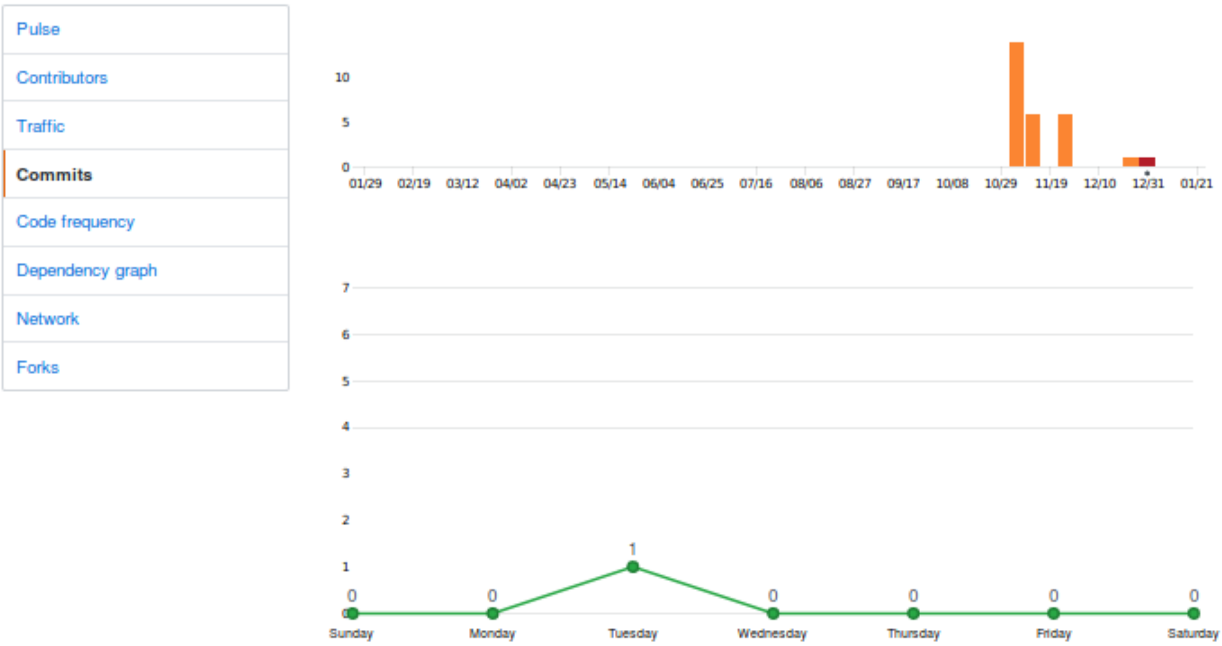
3.2 Work to be done

- : To come up with a neat-networking protocol schema to address inefficient hop-on /ad-hoc communication propogation delays, possibly trying to implement in a decentralized contract.
- : To be able to successfully reproduce the hardware based real-time implementation of the AGL release on an ARM based development board.
- : To provide an future roadmap for Non-hybrid cum Hybrid on-road network integration in a cheap (feasible), environment-friendly (sustainanble) and energy-efficient (if not, utilitarian) by means of a snap-on dashboard powered by a simple smartphones' sensors, transmitters and transduecers.
- : To demonstrate a successful implementation of AGL (improvised fork) during the final review.

3.3 Gantt (Progression / Commits) chart



Commit history as takem from https://github.com/



Code frequency history as takem from https://github.com/

DATE	DAY	TIME	LOG		
21-Dec-2017	Thu	1000hrs.	Sampling		
22-Dec-2017	Fri	1000hrs.	Reconcilation		
23-Dec-2017	Sat	1000hrs.	Off		
24-Dec-2017	Sun	1000hrs.	Off		
25-Dec-2017	Mon	1000hrs.	Literature Review		
26-Dec-2017	Tue	1000hrs.	Literature Review		
27-Dec-2017	Wed	1000hrs.	Literature Review		
28-Dec-2017	Thu	1000hrs.	Literature Review		
29-Dec-2017	Fri	1000hrs.	Literature Review		
30-Dec-2017	Sat	1000hrs.	Off		
31-Dec-2017	Sun	1000hrs.	Off		
1-Jan-2018	Mon	1000hrs.	Analysis		
2-Jan-2018	Tue	1000hrs.	Analysis		
3-Jan-2018	Wed	1000hrs.	Analysis		
4-Jan-2018	Thu	1000hrs.	Analysis		
5-Jan-2018	Fri	1000hrs.	Analysis		
6-Jan-2018	Sat	1000hrs.	Off		
7-Jan-2018	Sun	1000hrs.	Off		
8-Jan-2018	Mon	1000hrs.	Hands-On Development		
9-Jan-2018	Tue	1000hrs.	Hands-On Development		
10-Jan-2018	Wed	1000hrs.	Hands-On Development		
11-Jan-2018	Thu	1000hrs.	Hands-On Development		
12-Jan-2018	Fri	1000hrs.	Hands-On Development		
13-Jan-2018	Sat	1000hrs.	Off		
14-Jan-2018	Sun	1000hrs.	Off		
15-Jan-2018	Mon	1000hrs.	Draft I		
16-Jan-2018	Tue	1000hrs.	Feasibility Analysis		
17-Jan-2018	Wed	1000hrs.	Milestone Scheduling		
18-Jan-2018	Thu	1000hrs.	Final Re-editing		
19-Jan-2018	Fri	1000hrs.	Omissions		
20-Jan-2018	Sat	1000hrs.	Off		
21-Jan-2018	Sun	1000hrs.	Off		
22-Jan-2018	Mon	1000hrs.	Review I		
23-Jan-2018	Tue	1000hrs.	Review I		
24-Jan-2018	Wed	1000hrs.	Review I		
25-Jan-2018	Thu	1000hrs.	Review I		
26-Jan-2018	Fri	1000hrs.	Review I		

Day-to-Day Activity Ledger: D.D.A.

References :

[1] Jawhar, I., Mohamed, N., Zhang, L.: Inter-Vehicular Communication Systems, Protocols and Middleware. pp. 1–3 (2010)

[2] Yang, X., Liu, J., Zhao, F., Vaidya N. H.: A Vehicle-to-Vehicle Communication Protocol for Cooperative Collision Warning. pp 1–14. (2003)

[3] Thangavelu, A., Saravanan, K. Rameshbabu, K.: A Middleware Architectural Framework for Vehicular Safety over VANET (In-VANET).pp 277–282 (2009)

[4] Luo, J., Hubaux, J.: A survey of Inter-Vehicle Communication. pp 1–12. (2004)

[5] Böhm, A.: State-of-the-art in networks aspect for Inter-Vehicle communication. pp 1–25. (2007)

[6] Keskin, U.: In-Vehicle Communication Networks: A literature Survey. pp 14 (2009).

[7] Nekovee., M.: Quantifying Performance Requirements of Vehicle-to-Vehicle Communication Protocols for Rear-end Collision Avoidance. pp. (2008)

Sources and Code Snips:

https://MEE499.github.io/

https://14BME0133.github.io/MEE499/

https://git.automotivelinux.com

https://itsn1x.github.io/agl-7782-0002-0009/