

## 7. RAKENNUSPROJEKTI

2.11.2017

Kurssiin kuuluvan isomman rakennusprojektin aiheeksi valittiin yksinkertainen ajoneuvodatan loggaussovellus. Pääasiallisena perusteluna valinnalle on kirjoittajan osalta työelämän puolelta tuleva paine perehtyä kyseiseen sovellusalueeseen, joten aihealueena tämä tarjoaa mahdollisuuden lyödä useampi kärpänen yhdellä iskulla. Mutta toki myös siviilielämän helpottaminen monin tavoin on yhtäläillä taustalla, kuten toivon mukaan projektin aikana tullaan huomaamaan.

Nykyaikaisissa henkilöautoissa (sekä yhtäläillä raskaassa kalustossa) liikkuu monenlaista sensori- yms. dataa. Paitsi että auton eri alijärjestelmät kuten moottorinohjain, abs-jarrut, luistonesto, vakionopeudensäädin jne. ovat tästä datasta täysin riippuvaisia, voi niistä olla oikein käytettynä myös hyötyä auton käyttäjälle. Esimerkiksi polttoaineen kulutuksen, ajonopeuden sekä kaasun/vaihteiden käyttötavan keskinäistä yhteyttä voitaisiin tilastoida datasta ja tehdä näin analytiikkaa kuljettajan ajotavan vaikutuksesta polttoaineen kulutukseen. Uudemmissa autoissa, joissa renkaiden ilmanpaine on erikseen anturoitu, voidaan ohjelmoida logiikka antamaan hälytys kun ilmanpaine laskee alle määritellyn kynnyksrajan - näin renkaiden vierintävastus saadaan pidettyä aisoissa, ja polttoainetta säästyy. Jos ja kun järjestelmä on kytketty verkkoon, käyttömahdollisuudet laajenevat jälleen. Ohjelmallisesti voidaan esimerkiksi tarkkailla polttoaineen tasoa, sekä saavutettaessa kynnyksarvo etsiä automaattisesti kuljettajalle halvin tankkauspaikka nykyisen reitin varrelta (ainakin teoriassa, olettaen että saatavilla on sopiva rajapinta josta tiedot saadaan haalittua).

Ajoneuvojen data on tätä nykyä CAN-väylädataa (Controller Area Network) ja siihen päästään käsiksi esimerkiksi OBD-diagnostiikkaliitännän kautta. Yleisin OBD:n käyttötarkoitus liittyy epäilemättä erilaisten vikakoodien lukemiseen ja nollaamiseen, mutta todellisuudessa liitännä mahdollistaa kaiken CAN-viestiliikenteen lukemisen reaaliajassa. Tämä johtuu siitä, että CAN on osoitteeton protokolla eli kaikki sanomat välitetään verkon kaikille solmuille. Jokaisen solmun tehtävänä on sanomatunnusten perusteella päätellä, tekeekö se yksittäisellä tiedolla jotain vaiko ei. [16]

Rakennusprojekti päätettiin toteuttaa Raspberry PI 3 –alustalle helpon ohjelmoitavuuden ja hyvien liitäntöjen ansiosta. OBD-väylästä data saadaan ELM 327-lukijalla, joka kytketään bluetoothin yli Raspberryyn. Laitteeseen ohjelmoidaan logiikka CAN-viestien dekodaukseen sekä tallentamiseen ihmisluettavassa muodossa. Muihin vaatimuksiin kuuluu mm. virranhallinta, sillä järjestelmän on osattava ajaa itsensä hallitusti alas kun virransyöttö menetetään esim. auton sytytysvirran sammuttamisen seurauksena.

Toistaiseksi on vielä ratkaisematta, tarvitaanko esim. akkua turvaamaan välivaiheen virransaanti.

Projektin toteutus tulee tapahtumaan vaiheittain siten, että aloitetaan perustoiminnallisuudesta ja edetään asteittain kohti vaativampia ominaisuuksia, niin kauan kuin aika ja muut resurssit antavat myöden. Alustava vaihejako on seuraavanlainen:

1. Peruskonfiguraatio: Virransyöttö 12VDC → 5VDC usb muuntajalta, bluetooth yhteys OBD lukijaan (konfigurointi ja testaus), CAN-sanomien lukeminen ja dekodaus sekä tallennus muistikortille.
2. ”Älykäs” virranhallinta – laitteen tulisi sammuttaa itsensä hallitusti kun virransyöttö menetetään – toteutustapa vielä avoin
3. Yhteydet ulkomaailmaan. Toteutetaan esim. älypuhelin-hotspotin kautta internet-yhteys ja pystytetään http-palvelin (vast) joka mahdollistaa (raaka)datan lukemisen etänä.
4. Siirrytään käyttämään nettiyhteyden muodostamisessa dedikoitua GPRS-moduulia (ks. esim. [17]) , jolloin laitteelle saadaan oma SIM-kortti eikä olla riippuvaisia matkapuhelimen jakamasta wifi-yhteydestä.
5. Tietojen automaattinen synkronointi/varmuuskopiointi pilvipalveluun tms.
6. Jatkojalostetaan nettiin päin tarjottavaa näkymää: analytiikka (polttoaineenkulutus, ajotapa), ulkoasu, HTML, CSS...

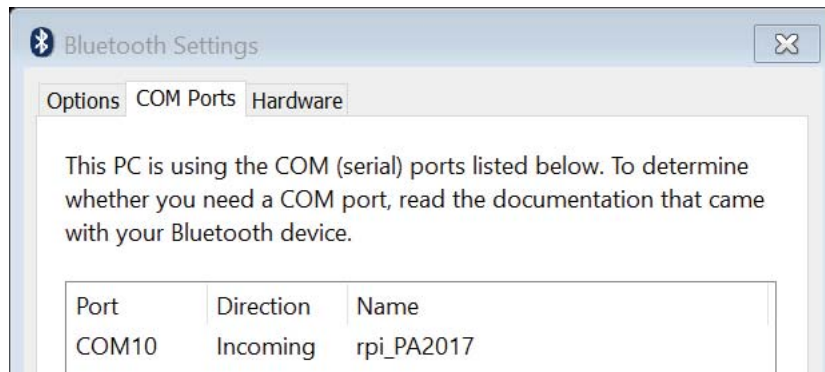
Projektille budjettikatoksi on määritelty 100 euroa. Tarvittava bluetooth-OBd lukija on olemassa jo valmiina, kuten myös tarvittavat sähkö/USB-johdot sekä auton tupakansytytinmuuntaja. Raspberry PI 3 – yksikkö tilattiin CDON.comista. Käyttökohdetta ajatellen kotelo kokoonpanolle on välttämätön hankinta, kuten tietysti myös 16GB microSD-muistikortti käyttöjärjestelmän ja ohjelmalogiikan tallennusta varten. Toistaiseksi hankinnoista on kustannuksia syntynyt 59,70 €. Mikäli myöhemmin päätetään hankkia GPRS-moduuli, syntyy tästä lisäkustannuksia, jotka voivat pistää budjetin ahtaalle.

#### 4. – 5.11.2017

Projektin alkuvaiheessa syntyi ajatus, että olisi kätevää kyetä simuloimaan ELM327-bluetooth rajapinnan tarjoamaa viestiliikennettä. Tällöin ei olla sidoksissa fyysisesti autoon, vaan sovelluskehitystä voidaan tehdä mukavasti sisätiloissa, mikä helpottaa etenkin alkuvaiheen kokeiluja. Tarkoitusta varten löytyikin Windows-alustalle tehty obdSIM [18]. Kyseinen näppärä pikkuohjelma kykenee emuloimaan useammankin eri CAN-protokollan mukaista viestiliikennettä ja välittämään sitä bluetooth-sarjayhteyden yli vastaanottavalle laitteelle, tässä tapauksessa siis Raspberryille. Vastaanottavassa päässä tietojen käsittelyyn valittiin Python OBD-kirjasto [19], joka on vaivatonta ottaa käyttöön Raspbianin natiivin Python-tuen ansiosta.

Emuloinnin toteutus pääpiirteittäin Windows 10-ympäristössä:

1. Asennetaan ObdSIM ja asetetaan Windows 'Test Modeen'
2. Bluetooth-asetuksista luodaan virtuaalinen COM-portti, joka välittää bluetoothin yli tiedot raspberryyn (kuva)



3. Käynnistetään simulaatio komentoriviltä ohjaten liikenne oikeaan porttiin

```
obdsim.exe -w COM10
```

4. Ellei ole jo, luodaan laitepari PC:n ja rasperryn välille bluetoothctl työkalulla tai graafisen UI:n kautta:

```
sudo apt-get install blueman
```

5. Asetetaan Raspberry kuuntelemaan sarjaporttiliikennettä bluetoothin kautta

```
sudo rfcomm bind /dev/rfcomm0 XX:XX:XX:XX:XX:XX
```

Missä XX... on lähettävän laitteen Bluetooth-osoite.

6. Asennetaan Python OBD ja tarvittaessa tätä ennen PIP

```
#Asenna pip: https://pip.pypa.io/en/latest/installing/
curl https://bootstrap.pypa.io/get-pip.py -o get-pip.py
sudo python get-pip.py
```

```
#Asenna python-obd
#http://python-obd.readthedocs.io/en/latest/
pip install obd
```

7. Testataan yhteys Python OBD:llä

```
import obd

#Parametreina asetettu portti ja baud rate
#Portin paikalle None → automaattinen haku
connection = obd.OBD('/dev/rfcomm0', 9600)

cmd = obd.commands.SPEED # select an OBD command (sensor)
```

```

response = connection.query(cmd) # send the command, and parse the re-
sponse

print(response.value) # returns unit-bearing values thanks to Pint
print(response.value.to("mph")) # user-friendly unit conversions

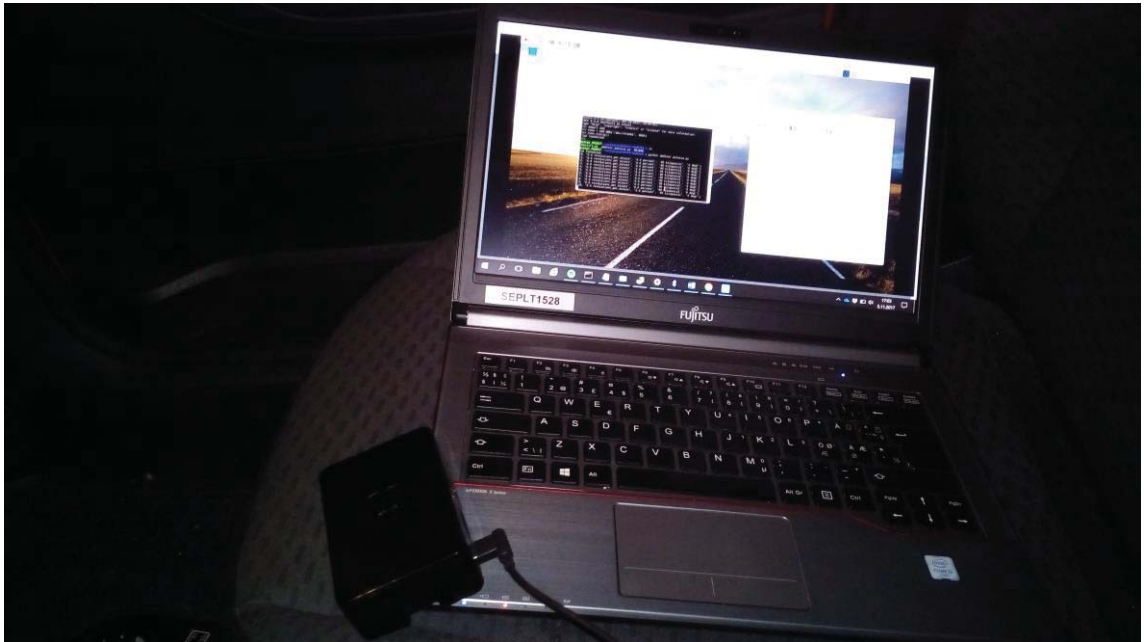
```

Jos kaikki meni hyvin niin yhteys on nyt luotu ja tietoja pystyy lukemaan erittäin mukavalla ja helppokäyttöisellä Python-syntaksilla, josta löytyy lisätietoa Python OBD kirjaston dokumentaatiosta [19]. Koska linux ja bluetooth on yleisesti ottaen hieman kenkkumainen yhdistelmä, kaikki ei kuitenkaan välttämättä mene hyvin. Tällöin vianselvittelyyn kannattaa varautua käyttämään 0 – 8 tuntia aikaa. Lisätietoa ja keskustelua aihepiiristä esim. lähteistä [20][21][22].

6.11.2017

Testattiin bluetooth-yhteyden toimivuus ja datalogging-perusominaisuudet autossa. ELM327-tökkeli OBD-väylään, raspberry ottaa virran tupakansytyttimestä ja ohjaus läppäriltä VNC-pilviyhteyden yli. Matkapuhelin-hotspot vaaditaan yhteyksien jakamista varten, koska dedikoitua gsm-moduulia ei tässä vaiheessa vielä ole Raspissa. Tiedot tallentuvat csv-formaattiin onnistuneesti, mutta valitettavan moni PythonOBD:n funktiokutsu tuottaa pelkkiä Null-arvoja ECU:ltä. Mm. polttoaineen kulutustietoa ei vielä saada onnistuneesti.





11.11.2017

Siirryttiin käyttämään MySQL-alustaa tietojen tallennukseen csv-tiedostojen sijaan. Asennettiin Raspiin LAMP-konfiguraatio (Apache, MySQL, PHP) jonka päälle voidaan toivottavasti tulevaisuudessa toteuttaa myös sovellukseen kuuluva web-reaaliaikanäkymä. Tällä hetkellä (simulaatio)datan tarkastelu onnistuu kotiverkossa phpMyAdminin kautta.

192.168.0.11:3000/phpmyadmin/sql.php?db=obdlogger&token=79b758610d960492...

Server: localhost:3306 » Database: obdlogger » Table: Entry

Browse Structure SQL Search Insert Export More

Show all Number of rows: 25 Filter rows: Search this table Sort by key:

+ Options

	EntryID	Timestamp	DeviceID	RPM	Calc_load	Sp
<input type="checkbox"/> Edit Copy Delete	1	2017-11-11 19:02:28		2003	0	
<input type="checkbox"/> Edit Copy Delete	2	2017-11-11 19:02:36		560	0	
<input type="checkbox"/> Edit Copy Delete	3	2017-11-11 19:02:46		1448	0	
<input type="checkbox"/> Edit Copy Delete	4	2017-11-11 19:02:57		3500	0	
<input type="checkbox"/> Edit Copy Delete	6	2017-11-11 19:54:31	1	3308	24.7058823529	
<input type="checkbox"/> Edit Copy Delete	7	2017-11-11 19:54:36	1	3891.75	42.7450980392	
<input type="checkbox"/> Edit Copy Delete	8	2017-11-11 19:54:42	1	4377	60.7843137255	

Teoriassa olisi mahdollista toteuttaa porttioshaus, jotta web-palvelimen tarjoamat rajapinnat olisivat käytettävissä myös oman verkon ulkopuolelta. Tätä testattiin kaapelilaajakaistayhteydellä varustetussa kotiverkossa onnistuneesti, mutta kun samaa



yrityttiin puhelimesta jaetulla hotspot-yhteydellä ei sitä saatu toimimaan. Liikkuvasta autosta yhteyksien toteuttaminen järkevästi on siis edelleen hieman kysymysmerkki.

Ohje yms-matskua vastaisuuden varalle:

Apache/mysql käyttöönottoon linuxissa [23].

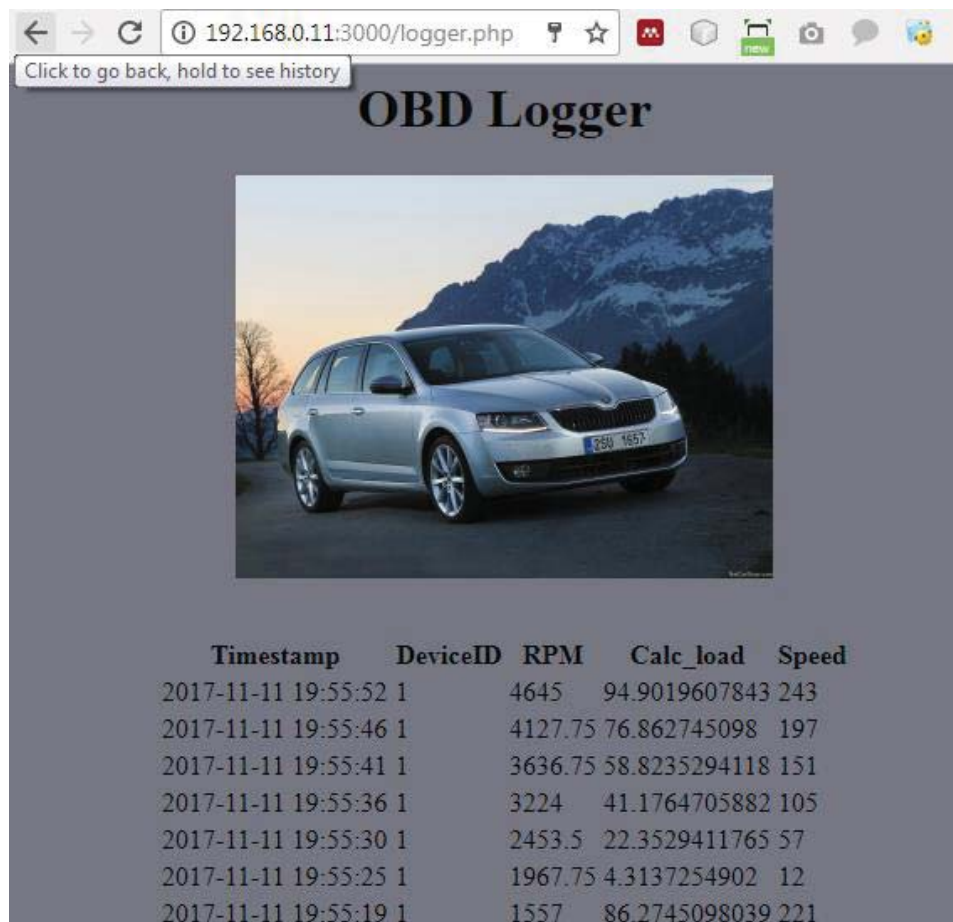
Python rajapinta MySQL käyttöön [24].

12.11.2017

Ensimmäinen viritelmä fronttinäkymästä kyhätty sisältäen kirjautumistoiminnon sekä tietokannan tietojen näyttämisen. Tällä hetkellä näytetään 15 uusinta tietoa kannasta kyselyllä:

”SELECT \* FROM Entry ORDER BY Timestamp DESC LIMIT 15”

Käytettyjen teknologioiden avainsanoja mm. html, php sekä erityisesti tietokantayhteyden luomiseen käytetty mysqli-kirjasto [25][26]. Ulkoasussa ja toiminnallisuudessa riittää vielä viilattavaa.



Timestamp	DeviceID	RPM	Calc_load	Speed
2017-11-11 19:55:52	1	4645	94.9019607843	243
2017-11-11 19:55:46	1	4127.75	76.862745098	197
2017-11-11 19:55:41	1	3636.75	58.8235294118	151
2017-11-11 19:55:36	1	3224	41.1764705882	105
2017-11-11 19:55:30	1	2453.5	22.3529411765	57
2017-11-11 19:55:25	1	1967.75	4.3137254902	12
2017-11-11 19:55:19	1	1557	86.2745098039	221

GPS-moduuli laitettu tilaukseen hintaan 21,80 sis. toimituskulut [27].

15.11.2017

Paranneltu web-näkymää, staattisen kuvan tilalla nyt google maps-upote joka näyttää autosta viimeisen tunnetun sijainnin. Tietokantaskeema ja tiedonkeruupään koodi muutettu vastaamaan tätä (lisätty Entry-tauluun uudet sarakkeet gps-koordinaateille). Sijaintitiedot ovat toistaiseksi simuloituja, mutta myöhemmin on tarkoitus lisätä tietojen lukeminen oikeasta GPS/GLONASS vastaanottimesta. Lisätty myös alkeellinen kirjautumisenhallinta php-sessioilla.

Loggauspäähän tehty myös hiukan muutoksia. Skripti kykenee nyt sammuttamaan raspberrystä virrat kun halutut ehdot täyttyvät – esim. yhteys bluetooth-lukijaan menetetään tai jokin kanava (esim. moottorin kierrosluku) tuottaa tarpeeksi pitkään pelkkiä 0-arvoja. Kehitysvaiheessa sammutus kannattaa pitää kuitenkin pois päältä.

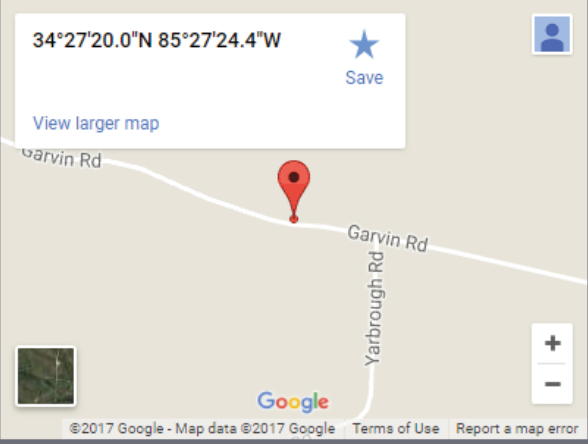
212.149.236.220:3000/logger.php

## OBD Logger

34°27'20.0"N 85°27'24.4"W

[Save](#)

[View larger map](#)



Garvin Rd

Garvin Rd

Yarbrough Rd

Google

©2017 Google - Map data ©2017 Google Terms of Use Report a map error

Timestamp	DeviceID	GPS_Lat	GPS_Long	RPM	Calc_load	Speed
2017-11-15 11:56:50		34.455555	-85.456789			
2017-11-15 11:32:51	1	60.452513	22.222222	0	0	0
2017-11-15 11:32:47	1	60.452513	22.222222	0	0	0
2017-11-11 19:55:52	1	0	0	4645	94.9019607843	243
2017-11-11 19:55:46	1	0	0	4127.75	76.862745098	197
2017-11-11 19:55:41	1	0	0	3636.75	58.8235294118	151
2017-11-11 19:55:36	1	0	0	3224	41.1764705882	105
2017-11-11 19:55:30	1	0	0	2453.5	22.3529411765	57
2017-11-11 19:55:25	1	0	0	1967.75	4.3137254902	12
2017-11-11 19:55:19	1	0	0	1557	86.2745098039	221
2017-11-11 19:55:14	1	0	0	1142.5	68.6274509804	175
2017-11-11 19:55:09	1	0	0	682.75	50.5882352941	130
2017-11-11 19:55:03	1	0	0	6263.5	32.5490196078	84
2017-11-11 19:54:58	1	0	0	5828.25	14.9019607843	39
2017-11-11 19:54:52	1	0	0	5336.5	96.862745098	248

[Log out](#)

Käytännön kehitystyö VNC-etätyöpöydällä on etenkin pilviyhteyden kautta tökkivää ja etenkin skriptien kirjoittelu voi viiveen takia olla melkoista tuskaa. Onneksi graafista käyttöliittymää linuxissa tarvitaan harvoin, joten on olemassa parempi ratkaisu. SSH-yhteys ei ole oletuksena päällä raspberryssä, ja se saadaan käyttöön laitteen asetuksista komennolla `sudo raspi-config`. Lisäksi tarvitaan muutamia muita ennakkovalmisteluja ja säätöjä, joihin tutustuminen kannattaa aloittaa tiedostosta `/etc/ssh/sshd_config`. Autentikointimenetelmäksi lienee ainakin kehitysvaiheessa luontevin vaihtoehto perinteinen käyttäjätunnus/salasanakirjautuminen. Tunnukset ovat samat, kuin fyysiselle laitteelle suoraan kirjautuessa. SSH ei oletuksena myöskään huoli tyhjää salasanaa, joten mikäli kirjautuminen ei onnistu, kannattaa ensimmäisenä varmistaa että käyttäjätunnus on olemassa ja että salasana on asetettu. Myös `/var/log/auth.log` saattaa olla kätevä työkalu kirjautumisongelmien selvittelyyn.

Kun SSH on asennettu ja kofiguroitu onnistuneesti, komentojen ajaminen Raspberryssä etänä onnistuu windows-maailmassa esim. Puttylla [29] tai vaikka android-älypuhelimella JuiceSSH-clientilla [30]. Enää ei tarvitse siis välttämättä kantaa edes läppäriä mukana kenttätesteissä, vaikka toki laajoja skriptejä puhelimella työstäessä alkaa aika nopeasti kaivata kunnollista näppäimistöä ja näyttöä.

*16.11.2017*

Päästiin testaamaan juuri postissa saapunutta GPS-vastaanotinta, joten ei muuta kuin USB-tökkeli kiinni raspberryyn.



Aivan ensiksi kannattaa komennolla `lsusb` testata että laite tunnistetaan onnistuneesti. Tämän jälkeen asennettavia juttuja ovat ainakin `gpsd` [31] sekä pythoniin `gps3`-kirjasto [32], joka käyttää ensiksi mainittua. GPS-vastaanottimen toiminnan voi testata komennolla `cgps -s`, jonka pitäisi alkaa pukkaamaan sijaintitietoja ruudulle.





tankkaamaan sekä poikettiinpa vielä testimielessä munkkikahveille paikalliseen sotilaskotiin. Koko tämän ajan loggaus jatkui suunnitellusti siitä huolimatta, että moottori oli välillä sammuksissa. Sen sijaan klo 16.28, kun oltiin jo Yläneellä, tiedonkeruu keskeytyi jälleen täysin yllättäen.

Keskeytymisten tarkkaa syytä ei ole vielä päästy tutkimaan, mutta vahva epäily kohdistuu nimenomaan gps-signaalin luotettavuuteen ja siihen, että koodi ei osaa vielä käsitellä näitä tilanteita asianmukaisesti.

*17.11.2017*

Otettiin merkittävä askel kohti kerrosarkkitehtuuria, Apache/MySQL-palvelin pyörii nyt Raspberryn sijasta kiinteällä laajakaistayhteydellä varustetussa kotikoneessa. Tämä mahdollistaa tietokantakyselyiden tekemisen sekä web-näkymän tarkastelun myös silloin kun raspberryllä ei ole verkkoyhteyttä tai virransyötön kanssa on ongelmia (erittäin yleinen ongelma mobiileissa IoT-laitteissa). Tarkalleen palvelin on nyt toteutettu Windows 7-pöytäkoneella pyörivästä bitnamin LAMP-virtuaalikoneesta, jonka verkkoyhteys konfiguroitiin sillattuun moodiin siten että serveriin pääsee internetistä käsiksi. Tämä vaatii myös porttiohjauksen toteuttamisen reitittimen asetuksista. Tämän jälkeen tietojen katselu onnistuu mistä tahansa, kunhan reitittimen julkinen IP sekä avoin portti on tiedossa. Pitkäaikaiseen käyttöön kyseistä viritystä tuskin kuitenkaan voidaan suositella mm. turvallisuusnäkökohtien vuoksi.

Polttoaineen kulutustiedon saaminen suoraan OBD:ltä on osoittautunut haastavaksi. Python-OBd:ssä on hetkellistä kulutusta varten kyllä oma kysely `obd.commands.FUEL_RATE`, jonka pitäisi teoriassa antaa polttoaineen massavirta yksikössä g/s. Kumpikaan projektin aikien testatuista autoista ei kuitenkaan ole tukenut tätä komentoa suoraan. Tässä vaiheessa projekti on kestänyt jo niin kauan, että autokin on ehtinyt vaihtua välissä vm. -11 Fabiasta vuoden 2014 farmari-Octaviaan. Uudesta autosta saadaan sentään ulos polttoainetankin taso prosentteina sekä ajatut kilometrit, joista kulutuksen periaatteessa voisi laskea, kun tunnetaan polttonestesäiliön kokonaistilavuus. Laskenta tosin vaatisi vielä huomattavaa kehitystyötä sillä tankkaukset pitäisi jotenkin kytä ottamaan huomioon ja tällöinkin tarkkuuden kanssa voisi olla vähän niin ja näin. Kyseiset tiedot on nyt joka tapauksessa lisätty mukaan tiedonkeruuseen.

Tarkemmin ajatellen tietokanta olisi kyllä verkkoyhteysohjelmien varalta syytä kahdentaa siten, että luetut tiedot rekisteröityvät myös etäserverin lisäksi Raspiin paikallisesti. Tietojen synkroonissa pitäminen olisi kuitenkin kokonaan oma harjoituksensa, mihin tuskin ehditään lähitulevaisuudessa paneutua.

18.11.2017

Projektin kehitystyö keskeytetään toistaiseksi. Tähän mennessä on saatu aikaan toimiva prototyyppi IoT-sovelluksesta, jollaisia tiettävästi on jo myös kaupallisessa käytössä. Kaikki perustoiminnallisuudet kuten tietojen lukeminen OBD:stä, tallennus tietokantaan sekä tarkastelu web-käyttöliittymästä ovat toteutettuna sekä testattu todellisissa käyttöolosuhteissa. Poikkeustilanteita lukuunottamatta, järjestelmä on toiminut jopa hämmästyttävän hyvin. Selkeitä kehityskohteita onkin juuri vahvemman poikkeuskäsittelyn sekä virhetilanteesta toipumisen toteuttaminen. Tällä hetkellä loggausskripti joko kaatuu tai ajautuu ikuisen silmukkaan, mikäli esim. yhteys tietokantaan tai GPS-signaali menetetään. Nämä eivät luonnollisestikaan ole IoT-laitteelle toivottuja ominaisuuksia.

Laitteen tulisi myös optimitalanteessa toimia mahdollisimman autonomisesti siten, että tietoja tallennetaan silloin kun sille on edellytykset olemassa ja kun se on ylipäättään järkevää. Tällä hetkellä loggauksen aloitus tapahtuu manuaalisesti, kuten myös lopetus, ellei suoritus sitten keskeydy virheeseen. Optimitalanteessa laite menisi jonkinlaiseen ”virransäästötilaan”, kun moottori on ollut sammuksissa ajan x, ja käynnistyisi jälleen uudestaan kun se havaitsee että väylässä alkaa liikkua mielenkiintoista dataa. Tällöin laite voitaisiin luultavasti huoletta jättää autoon pidemmäksikin aikaa tietoja tallentamaan.

Kolmas toistaiseksi ratkaisematon haaste on rasperryn mobiili verkkoyhteys, joka toistaiseksi on toteutettu matkapuhelimesta hotspottaamalla, mikä on mahdollistanut myös etätyöpöytä- sekä SSH-käytön langattoman lähiverkon yli. Optimitalanteessa raspissa tulisi olla kuitenkin joko oma integroitu gsm-moduuli tai USB-väylään liitettävä 3/4g-”mokkula”, jolla yhteys muodostetaan. SSH:n käyttömahdollisuus julkisen IP-osoitteen yli olisi ehdoton plussa, joten ratkaisun tulisi optimaalisesti olla sellainen joka tukee tätä. Markkinoilla on olemassa erityisesti ajoneuvokäyttöön tarkoitettuja SIM-korttipaikalla varustettuja reitittimiä, jotka voisivat ajaa asian, ks. esim [33]. Tällaisen hankkiminen harjoitustyötä varten olisi kuitenkin hiukan yliampuvaa.

Projektin                      lähdekoodit                      löytyvät                      repositoriosta  
<https://github.com/donkkis/sulautetut2017s/tree/master/obdPIPA> .

## LIITE 1

Viitteellinen tuntikirjanpito tähän astisista tehtävistä:

Pvm	Tunnit	Osio(t)	Tehtävät
9.9.2017	4	Käyttöjärjestelmät	Virtuaalikoneet, Linux, Python, C++
10.9.2017	8	Käyttöjärjestelmät, Ohjelmointi	GitHub perusteet + linux integraatio, Virtualisointi rautatasolla, Linux Terminal, BitNami LAMP, SSH, MySQL/phpMyAdmin
11.9.2017	4	Ohjelmointi	C++ perus
12.9.2017	6	Ohjelmointi	C++ oliot, Python 3
20.9.2017	4	Arduino	Harjoitukset ja niiden dokumentointi
21.9.2017	3	Arduino	Harjoitukset ja niiden dokumentointi
28.9.2017	2	Arduino	Harjoitukset ja niiden dokumentointi
4.10.2017	4	Raspberry PI	Yleistä luottokorttikoneista, perusasetukset, sense hat dataloggaus
11.10.2017	4	Raspberry PI	Kameramoduulin käyttö, vnc-etätyöpöytä, www-palvelin, bash-skriptaus/crontab linuxissa
12.10.2017	1	Raspberry PI	Dokumentointi
26.10.2017	6	Rakennusprojekti	Suunnittelu
2.11.2017	1	IoT	Kirjallisuustutkimuksen aineistoon alustava tutustuminen ja aiheen valinta.
4.11.2017	8	Rakennusprojekti	Kehitystyö
6.11.2017	6	Rakennusprojekti	Kehitystyö
11.11.2017	6	Rakennusprojekti	Kehitystyö
12.11.2017	4	Rakennusprojekti	Kehitystyö

15.11.2017	8	Rakennusprojekti	Kehitystyö
16.11.2017	8	Rakennusprojekti	Kehitystyö
17.11.2017	8	Rakennusprojekti	Kehitystyö
18.11.2017	2	Rakennusprojekti	Dokumentointi
23.11.2017	2	IoT	Kirjallisuustutkimus
29.11.2017	2	IoT	Kirjallisuustutkimus
30.11.2017	2	IoT	Kirjallisuustutkimus

## LÄHTEET

- [1] NotepadQQ-ohjelmiston kotisivut. Viitattu 10.9.2017 <http://notepadqq.altervista.org/wp/download/>
- [2] GitHub Guides Hello World. Viitattu 10.9.2017 <https://guides.github.com/activities/hello-world/>
- [3] GitHub-tili. Viitattu 10.9.2017 <https://github.com/donkkis/>
- [4] Bitnami PhpMyAdmin. Bitnami Docs. Viitattu 10.9.2017 [https://docs.bitnami.com/virtual-machine/components/phpsudmyadmin/?utm\\_source=bitnami&utm\\_medium=virtualmachine&utm\\_campaign=Virtual%2BMachine](https://docs.bitnami.com/virtual-machine/components/phpsudmyadmin/?utm_source=bitnami&utm_medium=virtualmachine&utm_campaign=Virtual%2BMachine)
- [5] Frequently asked questions for virtual machines. Bitnami Docs. Viitattu 10.9.2017 <https://docs.bitnami.com/virtual-machine/faq/#connecting-with-an-ssh-client-on-windows>
- [6] Verkkokeskustelu Bitnami communityssä. Viitattu 10.9.2017 <https://community.bitnami.com/t/accessing-phpmyadmin-remotely-i-get-this-security-error-for-security-reasons-this-url-is-only-accesible-using-localhost-127-0-0-1-as-the-hostname/6718>
- [7] Embedded With LAMP. Käyttäjän miksa007 projekti GitHub-palvelussa. Viitattu 10.9.2017 [https://github.com/miksa007/Embedded\\_with\\_LAMP](https://github.com/miksa007/Embedded_with_LAMP)
- [8] Viope. Olio-ohjelmoinnin perusteet C++ -kielellä. Kurssimateriaali.
- [9] <https://www.arduino.cc/en/Guide/Introduction>
- [10] <https://en.wikipedia.org/wiki/Theremin>
- [11] <https://www.arduino.cc/en/Tutorial/tonePitchFollower>
- [12] Luottokortin kokoiset monitaiturit. Mikrobitti Elokuu 2016.
- [13] [https://www.raspberrypi.org/magpi-issues/Essentials\\_Camera\\_v1.pdf](https://www.raspberrypi.org/magpi-issues/Essentials_Camera_v1.pdf)
- [14] <https://www.raspberrypi.org/documentation/remote-access/vnc/>
- [15] <https://www.raspberrypi.org/documentation/remote-access/web-server/apache.md>
- [16] <https://fi.wikipedia.org/wiki/CAN-v%C3%A4yl%C3%A4>
- [17] [http://ihmevekotin.fi/product/577\\_raspberry-pi-gsm-hat](http://ihmevekotin.fi/product/577_raspberry-pi-gsm-hat)



- [18] <https://icculus.org/obdgpslogger/obdsim.html>
- [19] <http://python-obd.readthedocs.io/en/latest/>
- [20] <https://www.raspberrypi.org/forums/viewtopic.php?t=138145>
- [21] <https://www.raspberrypi.org/forums/viewtopic.php?t=78838>
- [22] <https://www.raspberrypi.org/forums/viewtopic.php?p=521067#p521067>
- [23] <https://pchelp.ricmedia.com/setup-lamp-server-raspberry-pi-3-complete-diy-guide/3/>
- [24] <https://dev.mysql.com/doc/connector-python/en/>
- [25] <http://php.net/manual/en/book.mysql.php>
- [26] [https://www.w3schools.com/php/php\\_mysql\\_intro.asp](https://www.w3schools.com/php/php_mysql_intro.asp)
- [27] [http://ihmevekotin.fi/product/831\\_gpsglonass-vastaanotin-usb-porttiin-u-blox7](http://ihmevekotin.fi/product/831_gpsglonass-vastaanotin-usb-porttiin-u-blox7)
- [28] <https://www.raspberrypi.org/documentation/remote-access/ssh/>
- [29] <http://www.putty.org>
- [30] <https://play.google.com/store/apps/details?id=com.sonelli.juicessh&hl=fi>
- [31] <http://www.catb.org/gpsd/installation.html>
- [32] <https://pypi.python.org/pypi/gps3/>
- [33] <https://teltonika.lt/product/rut950/>
- [34] Sforzin, A. & Conti, M. 2016. RPiDS: Raspberry Pi IDS A Fruitful Intrusion Detection System for IoT
- [35] Checkoway, S., McCoy D., Kantor, B., Anderson, D., Shacham, H., Savage, S., Koscher, K., Czeskis, A., Roesner, F., Kohno, T. 2011. Comprehensive Experimental Analyses of Automotive Attack Surfaces.
- [36] <https://www.youtube.com/watch?v=bHfOziIwXic>