|  |  |
| --- | --- |
| RĪGAS TEHNISKĀ UNIVERSITĀTE | |
| DATORZINĀTNES UN INFORMĀCIJAS TEHNOLOĢIJAS FAKULTĀTE | |
| Informācijas tehnoloģijas institūts | |
|  |  |
| Juris Kondrats | |
| TRANSPORTA IZSEKOŠANAS SISTĒMU FUNKCIONĀLO IESPĒJU ANALĪZE  Bakalaura darbs | |
|  | |
| DITF ITI  Meža ielā, 1/3 – 400  Tālrunis 67089515 | Zinātniskais vadītājs:  Maksims Alekseičevs  zinātniskais asistents, maģistra grāds |
| Rīga - 2015 | |

|  |  |
| --- | --- |
| RĪGAS TEHNISKĀ UNIVERSITĀTE | |
| DATORZINĀTNES UN INFORMĀCIJAS TEHNOLOĢIJAS FAKULTĀTE | |
| Informācijas tehnoloģijas institūts | |
|  | APSTIPRINU |
|  | Modelēšanas un imitācijas katedras vadītājs  Dr.habil.sc.ing. prof. J. Merkurjevs  \_\_\_\_\_\_\_\_\_ |
|  | 2013.gada 29. novembrī |
| Bakalaura darba uzdevums | |
| studentam Jurim Kondratam | |

1. Bakalaura darba tēma:

latviešu valodā: „Transportu izsekošanas sistēmu funkcionālo iespēju analīze”

angļu valodā: „Analysis of transport tracking system functionality”

Apstiprināta ar ITI direktora 201\_. gada “\_\_.” \_\_\_\_\_\_\_\_\_ rīkojumu Nr. \_\_\_\_\_\_

2. Bakalaura darba nodošanas termiņš: 2014. gada 26. Maijs.

3. Problēmas nostādne:

Mūsdienās, aizvien biežāk sastopamies ar problēmu, ka kādam uzņēmumam ir nepieciešams personīgais autoparks. Diemžēl, autoparka uzturēšana prasa nopietnus materiālus un laika ieguldījumus. Galvenās problēmas autoparka uzturēšanā ir automašīnu tehniskā stāvokļa novērtēšana, degvielas patēriņa samazinājums, kā arī dažādas drošības problēmas. Lai efektīvi risināt šīs problēmas, ir nepieciešama centralizēta sistēma, kas spētu uzkrāt informāciju par katras automašīnas datiem un pārvietojuma specifikācijām.

4. Bakalaura darba mērķis:

Izpētīt un izanalizēt transporta izsekošanas sistēmu funkcionālās iespējas un balstoties uz šo informāciju izstrādāt transporta izsekošanas sistēmas konceptu. Sniegt rekomendācijas šādu sistēmu pielietošanā, uzņēmuma autoparka kontrolēšanai.

|  |
| --- |
| RĪGAS TEHNISKĀ UNIVERSITĀTE |
| DATORZINĀTNES UN INFORMĀCIJAS TEHNOLOĢIJAS FAKULTĀTE |
| Informācijas tehnoloģijas institūts |
| TRANSPORTA IZSEKOŠANAS SISTĒMU FUNKCIONĀLO IESPĒJU ANALĪZE |
| Juris Kondrats |
| Anotācija |

Bakalaura darba ietvaros tika izpētītas Globālās Pozicionēšanas Sistēmas datu iegūšanas iespējas, kā arī, tika izpētīti un aprakstīti Globālās Pozicionēšanas Sistēmas darbības principi un signālu aprite starp virszemes pavadoņiem un Globālās Pozicionēšanas Sistēmas signālu uztveršanas moduļiem. Papildus, tam tika apskatīti dažādi transportlīdzekļu izsekošanas sistēmu raksturojumi un iespējas.

Tāpat darbā tika apkopota un analizēta informācija par transportlīdzekļa datordiagnostikas iespējām.

Iegūstot iepriekš aprakstīto informāciju, darba autors izstrādāja transportlīdzekļu izsekošanas sistēmas konceptu.

Viena no šī koncepta sastāvdaļām ir aplikācija, kas paredzēta darbam „Android” operētājsistēmas vidē un kura spēj nolasīt globālas pozicionēšanas moduļa (programmatūras testiem tika izmantots mobilais tālrunis ar iebūvētu GPS moduli) kā arī datus no OBD-II skenera (kā OBD-II skeneris tika izmantots ELM 327 skeneris ar ”Bluetooth” datu apmaiņas interfeisu, ērtāka savienojuma nodrošināšanai ar ierīci). Iegūstot šos datus, aplikācija nosūta tos uz serveri tālākajai apstrādei.

Kā otrā transportlīdzekļu izsekošanas sistēmas koncepta sastāvdaļa tika izstrādāta aplikācija, kas darbojas uz servera, saņem, apstrādā un saglabā datubāzē reāla laikā no transportlīdzekļa saņemtus datus.

Darba apjoms - 54. lpp., 1 tabula, 27 attēli un 3 pielikumi.

|  |
| --- |
| RIGA TECHNICAL UNIVERSITY |
| FACULTY OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY |
| Institute of information technology |
| FUNCTIONAL ANALYSIS OF TRANSPTORT TRACKING SYSTEM |
| Juris Kondrats |
| Annotation |

In this paper was investigated the Global Positioning System data acquisition opportunities, as well it describes the Global Positioning System operating principles and signal flow between the surface and the Global Positioning System reception modules. In addition, it was dealt with different vehicle tracking system characteristics and capabilities.

Also, in this paper was colected and analysed information on vehicle transport computer diagnostics opportunities.

Obtaining information described above, the author developed the concept of transport vehicle tracking system.

One of the components of the this concept is application which can work on „Android” operating system environment and can read global positioning module (in a software tests were used a cell phone with a GPS built-in module) as well as data from the OBD-II scanner (as the OBD-II scanner was used the ELM 327 scanner with ’ Bluetooth ’ data exchange interface for easier connection to device). Obtaining these data, the application sent them back to the server for processing.

As part of the second vehicle tracking system concept was developed application that runs on a server, to receive, process and store the data received from the vehicle in real time.

The work contains 54 p., 1 table, 27 figures and 3 appendixs.

|  |
| --- |
| РИЖСКИЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ |
| ФАКУЛЬТЕТ КОМПЬЮТЕРНЫХ НАУК И ИНФОРМАЦИОННОЙ ТЕХНОЛОГИИ |
| Институт информационной технологии |
| ФУНКЦИОНАЛЬНЫЙ АНАЛИЗ СИСТЕМ СЛЕЖЕНИЯ ЗА ТРАНСПОРТОМ |
| Юрис Кондратс |
| Аннотация |

В рамках данной бакалаврской работы были исследованы возможности получения данных глобальной системы позиционирования а также были рассмотрены и описаны принципы работы глобальной системы позицирования и обмен сигналоми между спутниками и дачиками приема сигнала. Дополнительно были осмотрены главные принципы работы систем транспортного слежения.

Также в работе была рассмотрена и проанализирована информация касающиеся возможностям транспортный компьюторный диагностики.

Получив и обработав вышеупомянутую информацию, автор работы разработал концепт системы транспортного слежения.

Одной из составляющих данного концепта является аппликация которая предназначена для работе на операционной системы „Android” и которая может считывать данные из ГПС модуля (для тестов использовался мобильный телефон с встроенным модулем GPS) a также данные из OBD-II сканера (как OBD-II сканер был использован ELM 327 сканер с „Bluetooth” портом для обеспечения удобного и простова сединения с устройством). Получив данные, аппликация отправляет их на сервер для дальнейшей обработки.

В качестве второй основной составной частью системы отслеживания транспортных средств была разработана программа, которая базируясь на сервере получает, обрабатывает и сохраняет в базу данных данные полученные в реальном времени из транспортных средств.

Объем работы - 54 стр., 1 таблиа, 27 рисунков и 3 приложение.

saturs

[Ievads 8](#_Toc420531607)

[1. Problēmas Nostādne 10](#_Toc420531608)

[2. TEORĒTISKĀ DAĻA 12](#_Toc420531609)

[2.1. Transporta izsekošanas sistēmas 12](#_Toc420531610)

[2.1.1. Transporta izsekošanas sistēmu raksturojums 12](#_Toc420531611)

[2.1.2. Transporta izsekošanas sistēmu iespējas 13](#_Toc420531612)

[2.1.3. GPS tehnoloģiju efektivitātes novērtējums 13](#_Toc420531613)

[2.1.4. Globālās pozicionēšanas sistēma 15](#_Toc420531614)

[2.1.5. A-GPS tehnoloģija 16](#_Toc420531615)

[2.1.6. Globālās pozicionēšanas sistēmu uztvērēji „Android” ierīcēs 18](#_Toc420531616)

[2.1.7. Darbs ar globālās pozicionēšanas sistēmu uztvērējiem „Android” OS vidē. 18](#_Toc420531617)

[2.2. Transporta datordiagnostika 20](#_Toc420531618)

[2.2.1. Transporta diagnostikas OBD II standarta raksturojums 20](#_Toc420531619)

[2.2.2. OBD-II datu pārraides iespējas. 21](#_Toc420531620)

[2.2.3. Protokols ISO 9141-2 22](#_Toc420531621)

[2.2.4. ELM372 OBD-II datordiagnostikas skeneris 23](#_Toc420531622)

[2.2.5. OBD-II komandas 24](#_Toc420531623)

[2.2.6. Darbs ar ELM327 diagnostikas skeneri „Android” operētājsistēmas vidē. 26](#_Toc420531624)

[2.2.7. OBD savienojuma simulācija 27](#_Toc420531625)

[2.3. Transportlīdzekļa koordināšu attēlojums 29](#_Toc420531626)

[2.3.1. „Leaflet” Javascript bibliotēka 29](#_Toc420531627)

[2.3.2. „Leaflet Routing Machine” JavaScript bibliotēka 30](#_Toc420531628)

[3. Programmatūras Izstrāde Un testēšana 31](#_Toc420531629)

[3.1. Sistēmas sastāvdaļu kopējais apraksts 31](#_Toc420531630)

[3.2. Datu bāze 32](#_Toc420531631)

[3.3. „Android” izsekošanas sistēmas aplikācija 34](#_Toc420531632)

[3.4. Web aplikācija transportlīdzekļa izsekošanai 42](#_Toc420531633)

[4. Novērtējums un Rekomendācijas 51](#_Toc420531634)

[SECINĀJUMI 53](#_Toc420531635)

[Literatūras saraksts 55](#_Toc420531636)

[Pielikumi 60](#_Toc420531637)

[1.Pielikums. Aplikācijas testa laikā iegūto datu tabula. 60](#_Toc420531638)

[2.pielikums. Web aplikācijas koda piemēri 61](#_Toc420531639)

[3.Pielikums. „Android” aplikācijas koda fragmenti 65](#_Toc420531640)

Ievads

Mūsdienās, aizvien biežāk, mēs sastopamies ar problēmu, ka kādam uzņēmumam ir nepieciešams personīgais autoparks. Diemžēl autoparka uzturēšana prasa nopietnus materiālus un laika ieguldījumus. Galvenās problēmas autoparka uzturēšanā ir automašīnu tehniskā stāvokļa novērtēšana, degvielas patēriņa samazinājums, kā arī dažādas drošības problēmas.

Lai efektīvi risināt šīs problēmas, ir nepieciešama centralizēta sistēma, kas spētu uzkrāt informāciju par katras uzņēmumā esošās automašīnas datiem un pārvietojuma specifikācijām. Šāda sistēma nozīmīgi atvieglotu uzņēmuma vadības darbu un palīdzētu racionalizēt uzņēmuma autoparka darbību. Šādas sistēmas galvenā funkcionālā prasība ir iespēja noteikt kāda konkrēta automobiļa atrašanās vietu, jebkurā laika momentā.

Kā piemēru šīs sistēmas izmantošanai var minēt jebkuru lielveikalu tīklu kompāniju. Šāda veida izsekošanas sistēma palīdzētu izsekot automobiļu kustību, kā arī optimizēt tās darbību, izstrādājot izdevīgākus maršrutus un kontrolējot piegādes laiku.

Šī darba galvenais uzdevums ir izpētīt un izanalizēt transporta izsekošanas sistēmu funkcionālās iespējas un izveidot transportlīdzekļa izsekošanas sistēmas aplikācijas konceptu, kurš sastāvēs no mobilās aplikācijas, kura apstrādās un nosūtīs uz serveri transportlīdzekļa tehniskos un atrašanās vietas datus.

Lai realizētu šo uzdevumu, tiks izstrādāta Web aplikācija, kas spēs saņemt un apstrādāt informāciju no mobilās ierīces ar iebūvētu globālās pozicionēšanas sistēmas moduli, kā arī sniegt rekomendācijas šādu sistēmu pielietošanā, uzņēmuma autoparka kontrolēšanai. Web aplikācijai ir nepieciešams izvadīt galvenos statistikas datus. Par zināšanu avotu šajā darbā, pārsvarā, tiks izmantota dažādu autoru literatūra, programmatūras ražotāju oriģinālā dokumentācija un interneta resursi.

Šis dokuments, galvenokārt, sastāvēs no 5 daļām. Pirmajā daļa tiks aprakstīta pētāmā problēma un šī darba aktualitāte.

Otrajā un trešajā daļā tiks aprakstītas dažādu pozicionēšanas un auto diagnostikas datu ieguves tehnoloģijas. Šajā daļā tiks aprakstītas šo tehnoloģiju datu ieguves noteikšanas, kā arī datu pārraides iespējas, kā arī dažādi veidi, kā doto funkcionalitāti implementēt transportlīdzekļa izsekošanas aplikācijā, kas darbosies „Android” operētājsistēmas vidē.

Ceturtajā daļā tiks aprakstīta izveidotā transportlīdzekļa izsekošanas aplikācija, kura darbosies „Android” operētājsistēmas vidē. Šī aplikācija nodrošinās savienojumu ar transportlīdzekļa OBD-II portu pateicoties ELM 327 datordiagnostikas modulim, kā arī noteiks atrašanās vietas koordinātes.

Viens no papildus mērķiem, šajā darbā ir izstrādāt Web aplikāciju, izmantojot bezmaksas risinājumus un tehnoloģijas. Šīs web aplikācijas galvenā funkcionalitāte ir datu iegūšana (no iepriekš minētās transportlīdzekļa izsekošanas aplikācijas) un apstrādāšana, kā arī apstrādāto datu izvade lietotājam saprotamā formātā. Šī aplikācijas darbība tiks aprakstīta trešajā daļā.

Piektajā daļā tiks sniegtas rekomendācijas izveidotā izsekošanas sistēmas koncepta pielietošanā un tiks veikts darba novērtējums.

Darba beigās tiks veikti secinājumi, kuros tiks īsumā izklāstīti un analizēti noteiktie mērķi un to īstenošana.

# Problēmas Nostādne

Starp plašajiem transportlīdzekļu izsekošanas sistēmu piedāvājumiem ir maz kompāniju, kuras piedāvā modernu risinājumu. Pārsvarā visas esošās plaši izplatītās sistēmas piedāvā salīdzinoši līdzīgus produktus, skatoties tieši uz funkcionālajiem aspektiem. Visi dati un aprēķini, kas tiek attēloti vai izskaitļoti dotajās sistēmās, ir iegūti vienīgi no GPS moduļu rādījumiem. Pēc autora domām, būtu diezgan parocīgi, lai šādas sistēmas paralēli datu ieguvei no GPS moduļiem spētu iegūt datus arī par transportlīdzekļa tehnisko stāvokli. Šāda veida funkcionalitāte palīdzētu transporta kompānijām sekot savā īpašumā esošo transportlīdzekļu tehniskajam stāvoklim.

Vēl viena problēma ir šādas sistēmas ieviešana. Aplikācijas, kuras pašlaik dominē tirgū piedāvā lietotājiem iegādāties atsevišķu GPS modeli, kurš darbojas autonomi, un viņa uzstādīšanai nereti ir nepieciešams profesionāls elektrotehniķis, kā arī nereti ir jāveic salīdzinoši lielas un sarežģītas izmaiņas transportlīdzekļa elektroniskajā sistēmā. Pēc autora domām tas nav ērts pasākums gala lietotājam, tāpēc darba autors piedāvā vienkāršāku risinājumu – izmantojot mobilu ierīci, kurā ir iebūvēts GPS modulis, un ir iespējams savienojums ar internetu. Šajā darbā tiks izveidota izsekošanas aplikācija, kas darbosies „Android” vidē, bet darba autors uzsver, ka šādu sistēmu ir iespējams izveidot arī „iOS” un „Windows” operētājsistēmu vidē.

Kā jau darba autors minēja iepriekš, pēc viņa domām ļoti perspektīva funkcija šādā sistēmā būtu transportlīdzekļa tehnisko datu ieguve. Starp šiem datiem kā pamat dati varētu tikt iegūti transportlīdzekļa odometra rādījumi, spidometra rādījumi, degvielas daudzuma un patēriņa rādījumi, dzinēja apgriezienu skaita minūtē rādījumi, kā arī transportlīdzekļa sastāvdaļās un kontrolmezglos radušos kļūdu uzskaite.

Pēc autora domām šāda funkcionalitāte ļautu rūpīgāk sekot transportlīdzekļa tehniskajam stāvoklim kā arī kontrolēt un optimizēt transportlīdzekļa darbību. Zinot datus par degvielas patēriņu un degvielas bākas tilpumu, lietotājas varētu pamanīt, ja transportlīdzeklim tiktu ļaunprātīgi nolieta degviela. Diemžēl šāda parādība ir bieži sastopama Latvijas uzņēmumos. Zinot odometra rādījumus, lietotājs varētu sekot vai transportlīdzeklis netiek lietots ārpus darba un netiek grozīti odometra rādījumi. Savlaicīga transportlīdzekļa darbības laikā pamanītas kļūdas dažkārt spēj savlaicīgi novērst radušos problēmu neļaujot tai kļūt sarežģītākai, un, neradot darbības traucējumus citos, transportlīdzekļa darbības mezglos, tādejādi samazinot izdevumus, kas tiek tērēti transportlīdzekļa tehniskā stāvokļa uzturēšanai, kā arī padarīt ilgāku transportlīdzekļa ekspluatācijas periodu.

Šīs funkcionalitātes nodrošināšanai ir nepieciešams ērts, viegli ieviešams kā arī ekonomiski izdevīgs risinājums. Kā vienu no variantiem ko redz autors ir datu ieguve caur OBD-II portu. Darba gaitā tiks apskatīts tieši šis variants. Šis ports ir izvietots lielākajā daļ transportlīdzekļu, kuri ir ražoti pēc 2000. gada, kā arī šobrīd tirgū ir pieejams liels klāsts dažādu skeneru, kuri spēj nolasīt informāciju no šī porta. Skeneru cenu svārstība svārstās no 5 līdz 1000 dolāriem par vienu vienību, bet absolūti visi skeneri var nolasīt transportlīdzekļa pamatdatus. Šajā darbā tiks izmantots ELM 327 skeneris, kura cena ir 5 dolāri un tam ir „Bluetooth” ports.

Darba galvenais uzdevums ir parādīt, ka šāds risinājums ir iespējams, tādēļ autors necentīsies īstenot un pielietot visu šī skenera potenciālu, bet gan parādīt pamat funkcionalitāti.

# TEORĒTISKĀ DAĻA

## Transporta izsekošanas sistēmas

Šajā sadaļā tiks apskatīta teorētiska informācija par transporta izsekošanas sistēmu kopējām iezīmēm un to funkcionalitāti.

### Transporta izsekošanas sistēmu raksturojums

Transportlīdzekļu izsekošanas ierīces, ieskaitot GPS sistēmas, tiek uzstādītas transportlīdzekļos, lai uzglabātu datus par braucošu transportlīdzekli vai transportlīdzekļu parku. Šāda veida ierīces ir ļoti noderīgas transportlīdzekļa ātruma izsekošanai un atrašanās vietas noteikšanai. Tika pierādīts, ka šādas monitoringa sistēmas pilnīgi atmaksājas automobilim, veicot 15000 km garu noskrējienu.[1]

Šādas sistēmas, galvenokārt, ir domātas autoparku kontrolei, kuri nodarbojas ar pasažieru un kravu pārvadājumiem. GPS monitoringa sistēma, ļauj izsekot uz kartes visus, kāda konkrēta autoparka transportlīdzekļus, un noteikt to parametrus (pārvietošanās ātrums, dzinēja temperatūras rādītāji, degvielas patēriņš u.t.t) reālajā laikā. Šāda sistēma ir spējīga automātiski sekot transportlīdzekļa darba zonai, respektīvi, iepriekš noteiktajam maršrutam. Ja, gadījumā, transportlīdzeklis pamet šo noteikto darba zonu, kontroles sistēmas lietotājam nekavējoties tiks paziņots par notikumu.[2]

Transporta izsekošanas sistēmas sastāv no divām daļām – programmatūras un iekārtas. Kā iekārta, galvenokārt, tiek izmantoti automobiļu globālas pozicionēšanas sistēmu uztvērēji – stacionāras ierīces iekārtas, kas pieslēdzas transporta elektriskajam tīklam. Nereti, kā globālās pozicionēšanas sistēmas uztvērējs tiek izmantots dators vai mobilais telefons, kuram ir pieejama GPS uztveres funkcija. Kā papildinājums šiem moduļiem var tikt izmantoti temperatūras un degvielas sensori, komunikācijas ierīces (mikrofons un skandas) Šī darba izstrādes laikā tiks izmantots mobilais telefons ar „Android” operētājsistēmu un iebūvētu globālās pozicionēšanas sistēmas moduli.[3]

Programmatūras daļa sastāv no datu servera un klienta programmnodrošinājuma.

Datu serveris saņem, apstrādā un uzglabā datus, kas tiek saņemti no GPS uztvērējiem. Klienta (dispečera) programmnodrošinājums ir aplikācija, kas atvieglo iegūto un apstrādāto datu analīzi.[4]

### Transporta izsekošanas sistēmu iespējas

GPS uztvērējs aprēķina objekta koordinātas, nosakot attālumu starp uztvērēju un četriem (var pieņemt, ka viens no tiem atrodas zemes centrā), vai vairāk GPS pavadoņiem, izmērot laiku, kāds pagājis starp signālu noraidīšanām no katra pavadoņa un to saņemšanām uztvērējā. Savukārt kopā ar signālu no pavadoņa tiek pārraidītas attiecīgā pavadoņa koordinātas. Ņemot vērā radiosignāla izplatīšanās laiku, ir nosakāma distance no šā pavadoņa.[5]

Katrs pavadonis šādi dod lodveida pozīcijas virsmu ap sevi. Divi pavadoņi jau dod riņķa līniju, kur viņu dažāda diametra "burbuļi" šķērso viens otru. Trešā pavadoņa "burbulis" šķērso šo riņķa līniju divos punktos. Ceturtais pavadonis dod pēdējo izvēli - tā "burbulis" šķērso vienu no šiem diviem atlikušajiem punktiem.[6]

Var pieņemt, ka šis ceturtais pavadonis atrodas zemes centrā, tad šī iedomātā pavadoņa pozīcijas virsma tiek aizstāta ar jūras virsmu, kas formas ziņā tuvu lodei ap Zemes centru. Līdz ar to, pieņemot, ka uztvērējs atrodas tuvu zemes virsmai, nepieciešami tikai trīs pavadoņi pozīcijas noteikšanai. GPS uztvērējos tas parasti tiek apzīmēts ar funkciju "3D/2D mode", kas var arī nebūt, ja uztvērējs nav domāts jūras lietošanai.[7]

### GPS tehnoloģiju efektivitātes novērtējums

GPS tehnoloģiju efektivitātes novērtējums tiek dots pēc vairākiem rādītājiem. Galvenie no tiem ir:

* materiālo ieguldījumu un izmaksu samazināšana;
* ekonomiskais izdevīgums; ražošanas rentabilitātes palielināšana;
* darba ražīguma palielināšana;
* kvalitatīvākas produkcijas ieguve;
* ekonomiski pamatota investīciju līmeņa kalkulēšana;

Pavadoņu lietošanu transporta kustības kontrolē ir saistīta ar šādām izteiksmēm:

* Pavadoņu monitorings;
* GPS monitorings;
* Transportu monitorings.[8]

Visi šie jēdzieni nes vienu un to pašu jēgu. Monitorings - tā ir regulāra izsekošana un uzdoto objektu novērošana. Mūsdienas tādu veidu pakalpojumus sniedz dažādi uzņēmumi, kuri ļoti ātri attīstās, jo to piedāvātās funkcijas ir ļoti pieprasītas mūsdienu tirgū. Lielā mērā, visi uzņēmumi piedāvā aptuveni vienādus pakalpojumus, bet to galvenie sadalījuma tipi, ir aprakstīti tālāk.

Pilsētas transporta izsekošanas tipa sistēmas, izmanto vairāk uzdevumiem, kurus, risina reālā laikā - vilcēju, novākšanas tehnikas, steidzamo dienestu un iekasētāju, moto kurjeru, takšu un sabiedriskā transporta autotransporta monitoringam.

Globālā tipa sistēmas, kuras apkalpo transportu jebkurā teritorijā. Izmantojamas galvenokārt attālinātiem un starptautiskiem pārvadājumiem, tehnikas darba kontrolei attālinātos rajonos ar neattīstītu infrastruktūru.

Globālais kravu tips atbild pat kravas kontroli jebkurā teritorijā. Izmantojams galvenokārt kravas atrašanās vietu autonomai kontrolei, starptautisku un starppilsētu pārvadājumu laikā. Pilsētas kravu tips izmantojams galvenokārt kravas atrašanās vietas autonomai kontrolei pilsētu teritorijās. [9]

Tagad reālajā piemēra aplūkosim sistēmu, kura tika projektēta kravas kustības monitoringam. Sistēmas nosaukums ir BN – Global Cargo.

BN – Global Cargo ir monitoringa un pārvaldes sistēma, kura ir paredzēta kravas atrašanās vietas noteikšanai, kā arī, transporta kontrolei kurš nav aprīkots ar monitoringa sistēmām. Tās funkcionālas iespējas:

* Atrašanas vietas definēšana - ātrumi, kravas un transportlīdzekļu kustības virzienu noteikšana automātiskā režīmā ar iepriekš uzstādītu laika intervālu;
* Atrašanas vietas un kravas kustības maršrutu attēlojums uz elektroniskas kartes;
* Visas informācijas, par transportlīdzekļa pārvietojumu ar kravu, glabāšana datu bāzē;
* Dažādu atskaišu izveidošana no datu bāzē glabātajiem datiem;
* Maršruta punktu kontrole uzdotajā laika periodā.[10]

BN – Global Cargo ir sistēmas komplekss, kas sastāv no autonomiem abonentu termināļiem un dispečera darbvietas. Transporta vienībā vai kravas konteinerī uzstāda BN - Global Box navigācijas terminālu, kurš automātiski nosaka atrašanas vietu, ātrumu, kravas vai transportlīdzekļa kustības virzienu. Tālāk pa GPRS datu pārraides kanāliem uzstādītais termināls automātiskajā režīmā pārsūta savākto informāciju uz dispečera darbvietu.[11]

Dispečera darbvieta ir aprīkojama ar datoru, kuram ir pieeja Internetam un darba iespējām ar FTP. Datorā tiek uzstādīts specializēts programmas nodrošinājums, no termināla saņemtās informācijas apstrādei. Informācija nonāk datu bāzē, atspoguļojas informatīvās tabulās un pēc tām uz var tikt izvadīta uz kartes.

### Globālās pozicionēšanas sistēma

Globālā navigācijas satelītu sistēma (GNSS), kas plašāk zināma kā Globālā pozicionēšanas sistēma (GPS) ir globālās pozicionēšanas jeb, vienkāršāk izsakoties, - precīzas atrašanās vietas noteikšanas sistēma. Tā sniedz precīzu informāciju par objekta atrašanās vietu dienā un naktī, visos laika apstākļos jebkurā vietā uz Zemeslodes.[12]

Patlaban pasaulē vienlaikus pastāv trīs GPS sistēmas - NAVSTAR (Navigation System with Timing And Ranging), GLONASS (Globālā Navigācijas Sistēma) un GALILEO. Latvijā tiek izmantota NAVSTAR GPS sistēma, kas turpmāk tiks plašāk apskatīta. To veido trīs savstarpēji saistīti posmi - satelītu segments, kontroles segments un lietotāja segments.

Satelītu segments - sastāv no satelītu kopas, kas riņķo ap zemi. 1973. gadā ASV valdība militāriem nolūkiem uzsāka projektu ar mērķi radīt mākslīgos pavadoņus, lai līdz ar to palaišanu orbītā ap Zemi, izveidotu globāla mēroga novērošanas sistēmu – nodēvētu par StarFIX. 1978. gadā tika palaists orbītā (aptuveni 22350 km augstumā) pirmais šī tipa pavadonis. Sistēmas izveide un pilnveidošana risinājās līdz 1994. gadam un tās pilnvērtīga izmantošana, tika uzsākta 1995. gadā. Šodien šo pavadoņu skaits ir jau 28. No tiem pozicionējošo koordināšu noteikšanā tiek izmantota 24 „OmniSTAR” pavadoņu kopa, kas raida L joslas frekvencēs (L1 1575,42 MHZ; L2 1227,60 MHZ) un ir paredzēta Globālās Pozicionēšanas Sistēmas (GPS) nodrošināšanai. [13]

Civiliem nolūkiem šo signālu sāka lietot ASV kopš 1997. gada. Sistēmas darbību nodrošina 21 aktīvais satelīts un trīs aktīvās rezerves satelīti, kas riņķo aptuveni 20 000 kilometru augstumā. Satelīti izvietoti sešās orbītās, un katrs satelīts apriņķo Zemi divas reizes dienā (ik pa 12 stundām). Tādējādi no jebkuras vietas jebkurā laikā ir redzami 4 līdz 8 satelīti. Kontroles segments – sastāv no sakaru, komunikāciju, datu uzkrāšanas, integrācijas, analīzes un kontroles iekārtām. Tās tiek izmantotas, lai novērotu, uzturētu un administrētu GPS satelītus un sistēmu. Piecas sakaru stacijas izvietotas uz Zemes ar galveno kontroles staciju Kolorādo Springsā. [14]

Galvenajā kontroles stacijā tiek uzkrāti dati par katru satelītu, tā stāvokli un statusu, kas tiek saņemti no pārējām kontroles stacijām u.c. avotiem. Galvenā kontroles stacija analizē informāciju un nosūta navigācijas, laika un citu informāciju uz katru satelītu. Galvenā kontroles stacija veic arī satelītu kursa koriģēšanu, to uzturēšanu un apkalpošanu. Lietotāju segments – sastāv no individuālu lietotāju grupas, kurā katram lietotājam ir viens vai vairāki GPS uztvērēji, kas veic satelītu raidīto signālu uztveršanu un pārvērš tos triju dimensiju koordinātēs. GPS signālu uztvērēji tiek ražoti kā patstāvīgi aparāti (stacionāri vai pārnēsājami) un kā PC (personālajam datoram), borta datoram un citām ierīcēm pieslēdzamas plates.

GPS satelīti augu diennakti ļauj noteikt jebkura zemeslodes punkta koordinātes reālā laika režīmā. Sistēmai kopš tās izveidošanas līdz pat šim laikam ir nosaukums NAVSTAR, bet apzīmējums GPS radās tad, kad to sāka izmantot arī civiliem mērķiem. „OmniSTAR” L-joslas signāls ir brīvi pieejams visā pasaulē, izņemot Antarktiku. Lai izmantotu GPS signālu, tā lietotājam ir nepieciešams attiecīgs signāla uztvērējs, kā arī dators ar atbilstošu programmatūras nodrošinājumu. Signālu uztvērējā ir uztveršanas modulis un mazgabarīta 7 antena ar pastiprinātāju. GPS ierīces parasti ir aprīkotas ar 12 kanāliem, kas ļauj vienlaikus uztvert praktiski visus uztveršanas zonā esošos pavadoņus. Jo no vairāk satelītiem uztverami signāli, jo precīzāk nosakāmas vietas koordinātas. Vidēji ir uztverami septiņu vai astoņu satelītu signāli. Civilā GPS nodrošina ticamību līdz 99% ar precizitāti līdz 3 m. [15]

GPS sistēma patlaban tiek lietota visā pasaulē kā militārajā, tā arī civilajā sfērā. Ar tās

palīdzību tiek kontrolēti militārie procesi, vadīta ieroču darbība, koordinēti pārvadājumi, kuģu, laivu un automašīnu kustība, meklēti nolaupītie transportlīdzekļi, novērota dzīvnieku migrācija, kontrolēti kosmosa aparāti, veikta zemes virsmas precīza kartēšana u.c. To arvien plašāk izmanto arī lauksaimniecībā precīzākai un efektīvākai saimniekošanai.

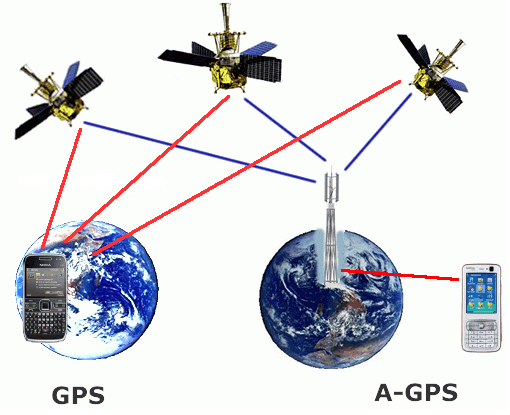
GPS pozicionēšana ir balstīta uz katra satelīta izplatītajiem radiosignāliem 10.23 GHz

pamat frekvencē. Tie izplata signālus arī citās frekvencēs, kas tiek atvasinātas no pamat frekvences. GPS uztvērēja trīsdimensiju (garuma, platuma un augstuma) koordinātu noteikšana notiek, izskaitļojot attālumu no signāla uztveršanas vietas līdz vairākiem satelītiem. Nosakot to savstarpējo izvietojumu, uztvērējierīce pēc ģeometrijas metodēm izskaitļo savas koordinātes. Teorētiski, lai noteiktu savu atrašanās vietu plaknē, pietiek ar trim satelītiem, bet praktiski, lai atrastu arī vietas augstumu, kā arī, rēķinoties ar iespējamām mērījumu precizitātes kļūmēm, vajadzīgi vismaz četri. [16]

### A-GPS tehnoloģija

A-GPS (Assisted GPS) sistēma tiek izmantota, lai paātrinātu Globālās pozicionēšanas sistēmas darba uzsākšanu mobilajā ierīcē vai arī ar A-GPS palīdzību ir iespējam noteikt atrašanās vietas koordinātes bez tiešas datu saņemšanas no Globālās pozicionēšanas sistēmas pavadoņiem. Informācija par pašreizējo satelītu atrašanās stāvokli šajā gadījumā tiek saņemta no mobilo sakaru tīkla operatora. Priekš tam mobilajā ierīcē vai komutatorā ir jābūt nokonfigurētam interneta savienojumam.[16]

Var izdalīt divus galvenos režīmus darbā ar A-GPS. Pirmais – galvenais, kad visu darbu atrašanās vietas koordinātu noteikšanai veic mobilo sakaru operatora infrastruktūra. Otrs – palīgveids, kas paātrina GPS iekārtas darbības sākumu, atjaunojot informāciju par efirmīdām almanahiem, un tā satelītiem izmantojot mobilo sakaru operatora datu bāzi. Šajā gadījumā iebūvētais GPS uztvērējs pats pieņem un apstrādā informāciju koordinātu noteikšanai. Galvenās A-GPS īpašības ir paaugstināta jutība un ātra veiktspēja. Atvērtās telpās un plašumos A-GPS ļauj iegūt atrašanās vietas koordinātes ar precizitāti līdz 15 metriem.[17]



2.1.att. **A-GPS un GPS darbības attēlojums**

A-GPS tehnoloģijas izstrādes un ieviešanas nepieciešamību veicināja liels aiztures laiks, kas rodas GPS darbības sākumā, līdz brīdim, kad tiek, nodibināts kontakts ar nepieciešamo pavadoņu skaitu. Šī savienojuma aizture var ilgt dažas sekundes, vai arī vairākas minūtes ierīcei atrodoties apvidū ar augstām ēkām. A-GPS izmantošanai mobilajās ierīcēs nepieprasa ievērojamas projektētāju izmaksas, jo salīdzinājumā ar GPS čipu un GPS antenām A-GPS čips aizņem daudz mazāk vietas, kā arī principā var būt integrēti bāzes iekārtas platformā.

Jāatzīmē, ka, lai nodrošinātu A-GPS darbību mobilā operatora tīklā, nepieciešama infrastruktūra, izvērsta uz tā bāzes stacijām, kas ļauj retranslēt GPS signālus. Papildu izmaksas mobilo sakaru operatoriem par aprīkojumu var ietekmēt to ka A-GPS pakalpojums var tikt piedāvāts par papildu samaksu.[18]

### Globālās pozicionēšanas sistēmu uztvērēji „Android” ierīcēs

Mūsdienās, aizvien izplatītākas, kļūst mobilās ierīces, kas darbojas „Android” operētājsistēmas vidē. Gandrīz katrā mobilās ierīces modelī, kura tiek ražota sākot no 2011. gada, ir iebūvēts globālas pozicionēšanas modulis. Iebūvējamie modeļi, galvenokārt, atšķiras savā starpā ar elektroenerģijas patēriņu un atrašanās vietas noteikšanas precizitāti. Jaunākie globālas pozicionēšanas modeļi var noteikt objekta atrašanās vietu ar precizitāti līdz 3 metriem.[19]

Dēļ tā, ka mobilās ierīces ir paredzētas tam, lai tās spētu darboties pēc iespējas ilgāk bez uzlādes no papildus enerģijas avota, ražotāji cenšas samazināt globālās pozicionēšanas moduļa darbībai nepieciešamo enerģijas daudzumu, kas savukārt noved pie signāla pavājināšanos. Vājš signāls, palielina mērījumu kļūdu, tādejādi padarot moduļa rādījumus neprecīzus. It īpaši, vājš signāls rada lielas problēmas pilsētas apstākļos, jo vājam signālam ir salīdzinoši grūti tikt cauri ēku jumtiem, kā arī pilsētas parasti ir piesārņotas ar daudz un dažādiem elektromagnētiskajiem signāliem.[20]

### Darbs ar globālās pozicionēšanas sistēmu uztvērējiem „Android” OS vidē.

Lai varētu sekmīgi darboties un programmēt „Android” aplikācijas, ir jāizprot „Java programmēšanas valodu”. Darbs ar globālās pozicionēšanas sistēmas moduli „Android” operētājsistēmas vidē, atšķiras no darba ar citiem moduļiem. Priekš darba ar GPS moduli, „Android" vidē ir paredzēta speciāla pakete „Android.Location”, kas satur sevī tādas klases kā „Location”, „LocationListener” un „LocationManager”. Galvenā šīs paketes specifika ir tāda, ka tā spēj nodrošināt mērķa pozicionēšanu, ne tikai ar GPS uztvērēja palīdzību, bet arī citādākos veidos, piemēram, noteikt atrašanās vietu izmantojot IP rādījumus. Šajā darbā, darba autors apskatīs tikai paketes piedāvātās funkcionālās iespējas, kas ir tieši saistītas ar globālās pozicionēšanas sistēmas moduli. [21]

GPS uztvērējs var tikt ieslēgts un atslēgts ar lietotāja palīdzību. Lai sekotu šīm statusa pārmaiņām interfeisā ir ieviestas funkcijas „onProviderEnabled()” un „onProviderDisabled()”. Turklāt gadījumā, ja GPS uztvērējs pārtrauks savu darbību kādas neparedzētas situācijas dēļ, piemēram, iebraucot tunelī, šim gadījumam var tikt izmantota „onStatusChange()” metode. Iepriekš apskatītās metodes ļauj aplikācijai operatīvi rīkoties, piemēram pārstāt nosūtīt ģeolokācijas datus uz serveri, līdz brīdim, kad GPS modulis atsāks savu darbību. „onLocationChanged()” metode ļauj apstrādāt pozīcijas maiņas notikumus. Izmantojot šo metodi, ir Iespējams nodrošināt datu pārraidi uz serveri tikai tajos gadījumos, ja tiek mainīta pozīcija.

„Location” klase, kas ir iekļauta „Android.Location” paketē, ļauj nolasīt ar globālās pozicionēšanas sensora palīdzību nolasītos datus. Metodes „getLongitude()” un „getLatitude()”, ļauj nolasīt atrašanās vietas garuma un platuma koordinātes. Metode „getAccuracy()” ļauj nolasīt noteiktās pozīcijas precizitāti, metros. „Android” operētājsistēmas uzturētāji apgalvo, ka precizitāte tiek noteikta kā 68% rādiusa ticamība. Tas nozīmē, ka, ja uzzīmēt apli, kura rādiuss ir vienāds ar precizitātes rādījumiem, un tas ir centrēts nolasītās koordinātes viduspunktā, tad pastāv 68% ticamība, ka reālās koordinātes punkts atrodas šajā aplī. [22]

Metode „getAltitude()” norāda augstumu virs WGS 84 references elipsoīda. Metode „getTime()” ļauj noteikt laiku koordinātes nolasīšanas momentā, UTC formātā. Metode „getSpeed()” ļauj noteikt pārvietošanās ātrumu, mērvienībā metri/sekundē. Šī metode aprēķina ātrumu, nosakot attālumu starp esošo koordinātes punktu un pirmspēdējo nolasīto koordinātes punktu un dalot to ar laika sprīdi starp mērījumiem. Šāda ātruma noteikšana nav pietiekami precīza, jo ātrums ir tieši atkarīgs no koordināšu precizitātes. Pat stāvot uz vietas, nolasīto koordināšu kļūdas dēļ ir iespējams iegūt pozitīvu ātrumu, kas ir lielāks par 0 metri/sekundē. Darba autors uzskata, ka labākais variants, kā rīkoties šādā situācijā, ir nolasīt rādītājus no transportlīdzekļa borta datora. Šī iespēja tiks plašāk apskatīta nākamajā nodaļā.

## Transporta datordiagnostika

Šajā nodaļā īsumā tiks aprakstītas transportlīdzekļu diagnostikas iespējas, kā arī tiks veikta ELM327 funkcionālo iespēju analīze.

### Transporta diagnostikas OBD II standarta raksturojums

Mūsdienās daudzas vieglās automašīnas ir aprīkotas ar pašdiagnosticējošām auto sistēmām. 1970. un 1980. gadu sākumā, ražotāji sāka izmantot elektroniskas motora vadības sistēmas un dzinēja diagnostikas tehnoloģijas. Iemesls tam bija stingrāku EPA (Vides aizsardzības aģentūra - Vides aizsardzības aģentūra, kas atrodas Amerikas Savienotajās Valstīs) prasību ieviešana, saistībā ar dažādām nevēlamām izdegušo gāžu emisijas sastāvdaļām atmosfērā. Laika gaitā, OBD (on board diagnostics) sistēma kļuvusi par sarežģītu sistēmu. OBD-II - jauns standarts izstrādāts 1990. gadu vidū, nodrošina pilnu kontroli pār auto dzinēju, kā arī ļauj novērot dažādas citas auto sastāvdaļas un papildu auto agregātmezglus ierīces un diagnosticēt šo mezglu pareizu darbību. [23]

OBD-II standarta izstrāde prasības un rekomendācijas tiek veikts piedaloties CARB (California Air Resourcer Board – Kalifornijas Gaisa resursu padome) un SAE (Society of Automotive Engineers – Automotīvo inženieru sabiedrība). Standarts OBD-II nodrošina daudz precīzāku kontroli pār dzinēju, transmisiju, katalizatoru, u.c. agregātiem. Piekļūstot borta datoram, sistēmas informāciju ieguvi var veikt ne tikai izmantojot specializētus skenerus, bet arī izmantojot universālus skenerus. Kopš 1996. gada visas pārdotās tērauda automašīnas ASV robežā atbilst OBD-II standartam. [24]

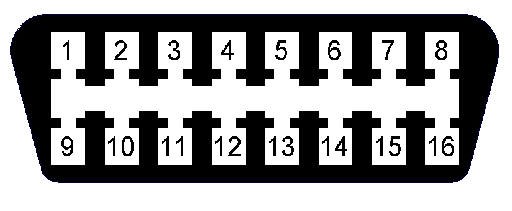
Eiropā, līdzīgi standarti tradicionāli tiek pieņemti ar nelielu kavēšanos attiecībā uz Amerikas Savienotajām Valstīm. Līdzīgi noteikumi EOBD (European On Board Diagnostic) stājās spēkā 2000. gada 1. janvārī. Ar EOBD un OBD-II diagnostikas procesu elektronisko sistēmu vienotiem standartiem, tagad var tikt izmantots viens un tas pats skeneris, visām marku automašīnām bez adapteru izmantošanas. OBD-II sistēma ir izstrādāta, lai uzraudzītu transportlīdzekļa funkcionālo sistēmas un komponentes, kas ietekmē emisijas (izplūdes) kvalitāti iekaitot degvielas sistēmas, aizdedzes sistēmas, EGR, benzīna tvaiku uztveršanas sistēmas, skābekļa sensorus, sildītājus, katalizatorus, katalizatoru sildītājus un otrreizējā gaisa iesūkšanas sistēmas. Gaisa maisījums, kas nepieciešams transporta darbībai, kā arī, transporta izmešu rādījumi tiek atjaunoti patstāvīgi. Citas transportlīdzekļa sistēmas un to sastāvdaļas tiek pārbaudītas 1 reizi brauciena laikā (Drive Cycle).[25]

Kāda agregāta kļūdas atrašanas gadījumā OBD-II saglabā kļūdas kodu elektroniskajā vadības bloka atmiņā un signalizē ar kļūdas indikatora palīdzību (MIL - nepareizas darbības indikators, kas apzīmējas ar „Check Engine” vai vienkārši „Check”). Tādejādi, izmantojot OBD-II skeneri un atbilstošu programmatūru var nolasīt kļūdas un ātri atrast transportlīdzekļa bojājuma cēloni. Papildus, kļūdu kodu nolasīšanai, programma ļauj izdzēst kļūdas, apskatīt fiksētos parametrus (freeze frame data), kontrolēt degvielas sistēmas parametrus, kontrolēt skābekļa sensoru parametrus, pārskatīt un uzraudzīt darbību skābekļa sensoru, novērot sistēmas parametrus reālajā laikā (datu plūsmas), apskatīt pašdiagnostikas pārbaužu rezultātus, kā arī nolasīt transportlīdzekļa identifikācijas datus.[26]

### OBD-II datu pārraides iespējas.

OBD-II standarts izmanto piecus dažādus datu apmaiņas protokolus: ISO 9141, ISO 14230 (otrais nosaukums - KWP2000), PWM, VPW и CAN. Katram protokolam ir vairāki interpretējumi, kuri galvenokārt atšķiras ar datu apmaiņas ātrumu un citām īpašībām. Galvenā no īpašībām pēc kuras var spriest vai transportlīdzeklis atbalsta OBD-II funkcionalitāti ir 16 kontaktu trapecveidīga spraudņa DLC (Diagnostic Link Connector) esamība transportlīdzeklī.[27]

Lielākajā daļā automobiļu šis spraudnis atrodas automašīnas salona iekšpusē zem galvenā rīku paneļa, no vadītāja puses. Tas var būt gan atvērts, gan arī aizvērts ar viegli noņemamu plastmasas vāku. Lai novērtētu dažādu skeneru skenēšanas iespējas priekš katra konkrēta transportlīdzekļa, ir nepieciešams noteikt transportlīdzekļa skenēšanas sistēmas servisa protokolu (ja transportlīdzeklis vispār atbalsta OBD-II funkcionalitāti). Priekš tā ir nepieciešams aplūkot diagnosticēšanas spraudni un noteikt izeju skaitu tajā. Parasti ir aktīvas tikai kāda daļā no visām izejām, jo katrs protokols izmanto sev nepieciešamās izejas.[25]



.. att. **OBD-II diagnosticēšanas spraudnis**

OBD-II diagnostikas spraudņa kontaktu nozīme:

02 - J1850Bus +

04 - Šasijas iezemējums;

05 - Signāla iezemējums;

06 - CAN High (J-2284);

07 - ISO 9141-2 K-Line;

10 - J1850 Bus-;

14 - CAN Low (J-2284);

15 - ISO9141-2 L-Line;

16 - Akumulatora spriegums.

Pēc izvadkanālu esamības var droši spriest par transportlīdzeklī izmantojamo datu apmaiņas protokolu:

* protokols ISO-9141-2 izmanto 4, 5, 7, 15, 16 izejas;
* protokols SAE J1850 VPW (Variable Pulse Width Modulation) izmanto 2, 4, 5, 16 izejas;
* protokols SAE J1850 PWM (Pulse Width Modulation) izmanto 2,4,5,10,16 izejas;[28]

Lielākais transportlīdzekļu klāsts izmanto ISO tipa protokolus. Šobrīd eksistē plašs, nedārgu skeneru klāsts, kas padara transporta diagnostikas iespējas pieejamas gandrīz katram transportlīdzekļa īpašniekam, tādejādi ļaujot tam savlaicīgi pamanīt radušos kļūmi un, iespējams, arī neitralizēt to paša spēkiem. Skeneri ir iespējams savienot ar jebkuru pārnēsājamu mobilo ierīci izmantojot USB datu pārraides vadu vai arī „Bluetooth” tehnoloģiju.[29]

### Protokols ISO 9141-2

ISO 9141 ir viens no izplatītākajiem protokoliem, kuru datu apmaiņai ar OBD-II portu izmanto vairums auto ražotāju. ISO 9141-1, ISO 9141-2, ISO 9141-3, ISO 9141-4 — tās ir dažādas ISO 9141 protokola realizācijas dažādos fizikālajos līmeņos (Physical layers). ISO 9141 ir salīdzinoši lēns protokols, kas nodrošina ātrumu līdz 10 KB / sek.

Protokoli ISO 9141 un ISO 14230 protokoliem sakrīt sakaru līniju izvietojumi, bet galvenā to atšķirtība ir tas ka viņi šis līnijas izmanto dažādos veidos. Tāpēc skeneri, kuri strādā izmantojot standartu ISO 9141, parasti var darboties arī ar ISO 14230, bet ne otrādi.[30]

Informācijas apmaiņa ar elektronisko vadības bloku notiek pa vienu vadu (tikai K-līnija) vai arī pa diviem (K un L līnijām). K-līnija ir divpusēja, un spēj gan nodot, gan saņemt signālu, savukārt L-līnija ir vienvirziena un tās galvenā, un vienīgā funkcija ir savienojuma izveide starp elektronisko vadības bloku un skeneri. Nodrošinot savienojumu L līnija pāriet loģiskā mezgla stāvoklī.

Veidojot savienojumu ar elektronisko vadības bloku, skeneris vienlaikus pa abām līnijām nosūta 8-bitu garu kodu ar ātrumu 5 biti sekundē. Ja kods ir pareizs, elektroniskais vadības mezgls nosūta skenerim 8-bitu ar informāciju par turpmākā savienojuma ātrumu. Apmaiņas ātrumu, nosaka transportlīdzekļa elektroniskais vadības bloks, nevis skeneris.

Pēc tam elektroniskais vadības mezgls nosūta vēl divus kodus ar informāciju par savienojumu un K un L līniju konfigurāciju. Skeneris nosūta šo kodu inversos kodus atpakaļ uz elektronisko vadības mezglu. Ar to arī beidzas skenera inicializācijas process.[31]

### ELM372 OBD-II datordiagnostikas skeneris

OBD-2 skeneri, kuri tiek veidoti uz ELM327 mikrokontroliera pamata, ir ļoti plaši izplatīti un labi zināmi ārvalstu lietotājiem, un ir, ieguvuši savu reputāciju, pateicoties savai drošībai, un universālajām spējām. Pateicoties šīm divām īpašībām lielāka daļa diagnostikas lietojumprogrammatūru ražotāji pasaulē pielāgo savas lietojumprogrammatūras darbam tieši ar šī veida skeneriem. Pašlaik, internetā ir izvietots ļoti plašs datordiagnostikas lietojumprogrammatūru klāsts, kas atšķiras ar piedāvāto funkcionalitāti, darba platformas prasībām, programmēšanas valodas atbalstu. To vidū ir gan komerciālie produkti, gan arī bezmaksas aplikācijas.

Galvenās ELM327 mikrokontroliera piedāvātās diagnosticēšanas iespējas ir :

* Transportlīdzekļa kļūdu (DTC) nolasīšana
* Transportlīdzekļa kļūdu (DTC) dzēšana
* Tekošo degvielas sistēmas darbības parametru izvade
* Tekošo kustības sistēmas darbības parametru izvade

Piedāvātās diagnosticēšanas iespējas ir atkarīgas arī no izmantojamās datu apstrādes programmatūras. Ne visas datu apstrādes programmatūras nodrošina pilnu skenera potenciālu izmantošanu. Ir vērts minēt, ka arī transportlīdzekļi, ne vienmēr spēj nodrošināt saskarni ar visām skenera diagnostikas iespējām.



.. att.**ELM327 diagnostikas skeneris**

ELM327 skeneri atšķiras arī ar datu izvades interfeisu. Ir dažādas skeneru modifikācijas, bet absolūti liekākā daļa no tām visām izvada datus tikai 3 dažādos veidos: izmantojot USB, „Bluetooth”, vai Wi-Fi tehnoloģiju. Šajā darbā tiks izmantots skeneris ar „Bluetooth” tipa datu pārraides interfeisu, kas ir ērtākais veids kā nodot datus portatīvai sistēmai. Šāda skenera izmantošanai ir vairākas pozitīvas iezīmes. Viena no šīm iezīmēm ir tāda, ka šāda veida skeneri ir salīdzinoši lēti. ELM327 čipus ar nepieciešamo programmnodrošinājumu ražo tikai viens uzņēmums pasaulē, bet brīvajā tirgū ir iespējams atrast ļoti daudzus šī skenera pakaļdarinājumus, kuri kvalitātes un funkcionalitātes ziņā, neatpaliek no oriģinālā skenera. Kā otro, no visām pozitīvajām iezīmēm, šī darba autors vēlētos minēt skenera ierīces praktiskumu. Nav nepieciešams veikt tehnoloģiski sarežģītas transportlīdzekļa pārbūves un uzlabojumus. Lai sistēma sāktu darboties, tā vienkārši ir jāiesprauž OBD-II diagnostikas spraudnī un jāsavieno to ar datu apstrādes ierīci.[32]

### OBD-II komandas

OBD komandas tiek veidotas no heksadecimāliem kodiem, kas tiek rakstīti, izmantojot ASCII rakstzīmes. Parasti šīs komandas sastāv no 2 vai vairākiem heksadecimālo skaitļu pāriem, taču ir dažas komandas, kas prasa tikai vienu heksadecimālo pāri.

Pirmais skaitļu pāris OBD komandā norāda OBD režīmu, kas jālieto. Visi nākošie heksadecimālo skaitļu pāri norāda parametra ID (PID), kas tiks nolasīts no noteiktā režīmā. Pastāv 10 dažādi OBD režīmi bet, lietojot skenerus, ir jāsaprot, ka ne visi transportlīdzekļi atbalsta visus 10 režīmus. Pirms testu veikšanas ir rūpīgi jāpārbauda konkrētā transportlīdzekļa protokols, lai redzētu, kuri OBD režīmi un parametri tiek atbalstīti.[33]

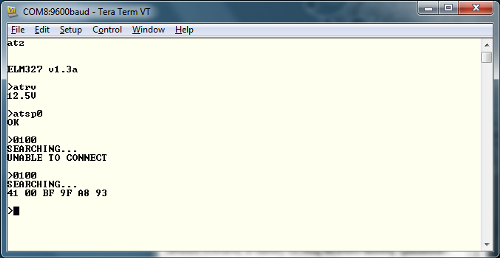
.. tabula

**OBD režīmi**

|  |  |
| --- | --- |
| Režīma numurs | Režīma apraksts |
| 01 | Pašreizējā informācija |
| 02 | Fiksētie transportlīdzekļa parametri |
| 03 | Kļūdu kodu diagnostika |
| 04 | Kļūdu kodu dzēšana |
| 05 | Skābekļa sensoru tests |
| 06 | Aizturēto kļūdu kodu diagnostika |
| 07 | Aizturēto kļūdu kodu diagnostika |
| 08 | Speciālais kontroles režīms |
| 09 | Transportlīdzekļa informācijas pieprasījums |
| 0A | Pastāvīgo kļūdu pieprasījums |

Daudzas no automašīnu ražošanas kompānijām pievieno savus PID kodus, un radītājus, kurus var nolasīt. Tāpēc, situācijā, kad skeneris tiek pielietots kādai konkrētai transportlīdzekļa markai, ir vērts apskatīt un izanalizēt konkrētā transportlīdzekļa piedāvātas OBD funkcijas. Kā arī ir vērts pieminēt to, ka ne katrs OBD-II skeneris spēj nolasīt un apstrādāt visus iespējamos OBD parametrus.

Iespējams viens no svarīgākajiem PID kodiem ir 00. Šis kods darbojas absolūti visos transportlīdzekļos, kuri atbalsta OBD-II. Ar šī koda palīdzību ir iespējams iegūt sarakstu ar visām šajā transportlīdzeklī atbalstāmajām PID komandām. Lai saņemtu šo sarakstu, skenerim ir jānosūta PID kods ar vērtību „0100”. Burtiski, tulkojot šo komandu, mēs iegūstam vaicājumu „režīmā 01, kuras komandas tiek atbalstītas”. Šādā veidā ir iespējams pārbaudīt pieejamās komandas katrā no režīmiem.[34]



2.4.att.**OBD komandas pieprasījuma un atbildes piemērs**

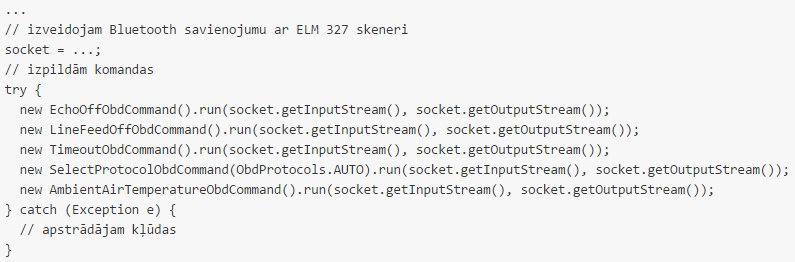
Visām OBD komandu izpildes atbildēm pastāv vienota struktūra. Pirmais atbildes baits (kā redzams piemērā 2.5 attēlā - 0x41) norāda režīmu, kurā komanda tika izpildīta. Tādā veidā transportlīdzekļa borta dators sūta atbildi 0x40 + 0x41 formā. Otrais bits norāda to, kāda komanda tika pieprasīta, tātad kā parādīts 2.3. attēla piemērā – 0x00. Visi nākamie baiti - 0xBF, 0x9F, 0xA8 un 0x93, respektīvi, ir pieprasītās komandas atbilde. Aprakstītajā piemērā tas ir saraksts ar atbalstāmajām transportlīdzekļa komandām, konkrētajā transportlīdzeklī.

Otra visbiežāk pielietojamā komanda ir dzinēju apgriezienu skaita sekundē nolasīšana – „010C”. Visas atbildes tiek sniegtas heksadecimālā formātā.[35]

### Darbs ar ELM327 diagnostikas skeneri „Android” operētājsistēmas vidē.

Viens no izplatītākajiem veidiem, kā savienot ELM 327 diagnostikas skeneri ar „Android” mobilo ierīci ir savienojot tās savā starpā caur „Bluetooth” sensoriem. Šāda veida savienojums, padara šo ierīču lietošanu ērtāku, jo darba telpa netiek norobežota ar citiem savienojuma elementiem, kā pieņemsim vads, kas var radīt diskomfortu, braucot pie stūres.

„Android” aplikācijas izstrādes laikā tiks izmantota „OBD-Java-Api” bibliotāka, kas ir izstrādāta Java valodā. Šī bibliotēka atbild par PID kodu pieprasījumu sūtīšanu un saņemtās atbildes apstrādi. Pateicoties šai bibliotēkai, lai saņemtu informāciju no transportlīdzekļa borta datora, ir nepieciešams palaist atbilstošas informācijas komandu, kas nolasīs un apstrādās atbildi, un atgriezīs atpakaļ atbildi. Lai bibliotēka darbotos, programmētājam ir jāparūpējas par to, lai ierīce tiktu savienota ar skeneri, pateicoties „Bluetooth” portiem.[36]



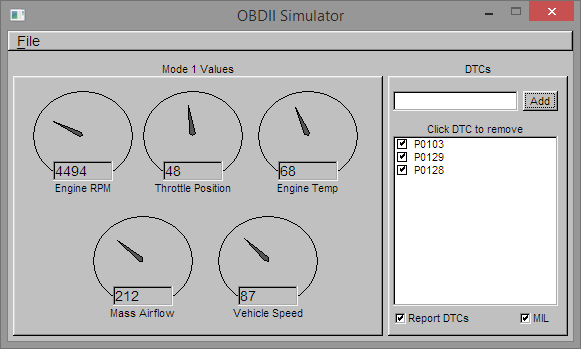
. att. **„OBD-Java-API” komandu piemērs**

2.3 attēlā tika parādīts neliels bibliotēkas darbības fragments. EchoOffObdCommand un LineFeedOffObdCommand klases atslēdz „atbalss” funkciju. TimeoutObdCommand uzstāda atbildes gaidīšanas laiku. Ja atbilde netiek sniega uzstādītajā laikā, tad bibliotēka atsūta „NO\_DATA” ziņojumu. SelectProtocolObdCommand uzstāda savienojuma protokolu. OBDProtocols.AUTO parametrs ļauj bibliotēkai noteikt savienojuma parametru automātiski.

### OBD savienojuma simulācija

Darba izstrādes laikā tiks izstrādāta aplikācija, kuras viena no galvenajām funkcionalitātē būs spēja nolasīt transportlīdzekļa OBD-II porta sniegto informāciju, Šī informācija ir nepieciešama, lai spētu noteikt transportlīdzekļa tehniskos rādījumus un kļūdu kodus reālajā laikā.

Programmatūras izstrāde ir ļoti ilgstošs process, kuram nepieciešama pastāvīga koda testēšana. Šāda veida testēšana reālajā darba vidē (izmantojot transportlīdzekļa OBD portu) ir diezgan neērta un apgrūtinoša. Testēšanai programmatūras sākuma stadijā ir nepieciešama aplikācija, kas spētu imitēt transportlīdzekļa OBDII porta signālu plūsmu. Viena no tāda veida Aplikācijām ir „OBD-II Simulator”.



2.7.att **„OBDII Simulator” aplikācijas fragments**

Attēlā numur 2.6. ir redzams „OBD-II Simulator” aplikācijas fragments. Šī ir atvērta koda simulācijas iekārta, kas darbojas gan Linux gan arī Windows operētājsistēmu versijās. Šīs aplikācijas darbības princips ir OBD-II kodu simulācija. Aplikācijas darbībai ir nepieciešams dators ar vienu no iepriekšminētajām operētājsistēmām, kā arī Bluetooth ports. Šo „Bluetooth„ portu iespējams savienot ar ierīci, uz kuras darbosies transportlīdzekļa izsekošanas sistēma, un tādā veidā simulēt signālu starp portiem.

Kā redzams attēlā, ar šis aplikācijas palīdzību ir iespējams simulēt tādus transportlīdzekļa darbības parametrus kā dzinēja apgriezienu skaits minūtē, dzinēja temperatūru, transportlīdzekļa ātrumu. Labajā stūrī lietotājs var ievadīt transportlīdzekļa DTC kļūdu kodus, kurus arī iespējams izmantot simulācijas laikā.

## Transportlīdzekļa koordināšu attēlojums

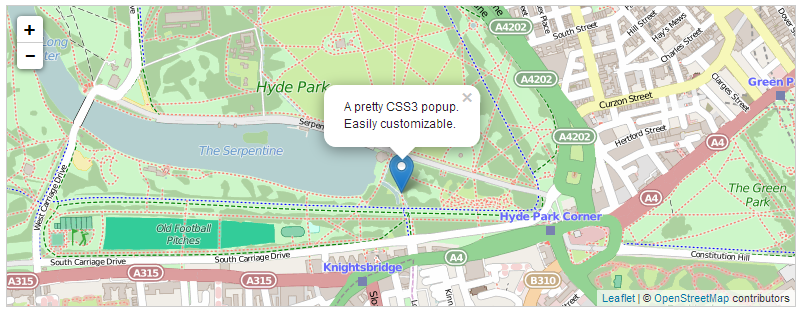
### „Leaflet” Javascript bibliotēka

„Leaflet” ir atvērtā pirmkoda JavaScript bibliotēka paredzēta atainot interaktīvas kartes, kuras atbalsta arī vairākas mobilās ierīces. Šī bibliotēka bija pirmo reizi izlaista 2011. gadā, pateicoties Vladimiram Agafonkinam, kurš arī izstrādāja šo bibliotēku. Viens no lielākajiem šīš bibliotēkas pozitīvajiem aspektiem, ir tas, ka bibliotēkas izmērs ir tikai 30kb, bet, neskatoties uz to, šī bibliotēka spēj atbalstīt praktiski visas galvenās izstrādātāju prasības.

„Leaflet” bibliotēka ir populāra bibliotēka, kura tiek izmantota web aplikāciju izstrādē. Šī bibliotēka atbalsta tādas tehnoloģijas, kā HTML5 un CSS3, un tiek pielietota daudzās populārās web lappusēs, kā piemēram „FourSquare”, „Pinterest” un „Flickr”.

Dotā bibliotēka ļauj izstrādātājiem attēlot nepieciešamo ģeolokācijas informāciju, neizmantojot smagnējas un dārgas GIS aplikācijas. Bibliotēkā piedāvātās kartes un pārējā nepieciešamā informācija tiek glabāta uz publiska servera. Katru reizi kad ir nepieciešamība attēlot kādu kartes fragmentu, tas tiek ielādēts līdz ar mājaslapas ielādi. Šajā gadījumā mājaslapas izstrādātājiem nav jārūpējas, par papildus vietas atbrīvošanu uz servera, jo kartes tiek ielādētas taisni uz klienta datoru.[37]

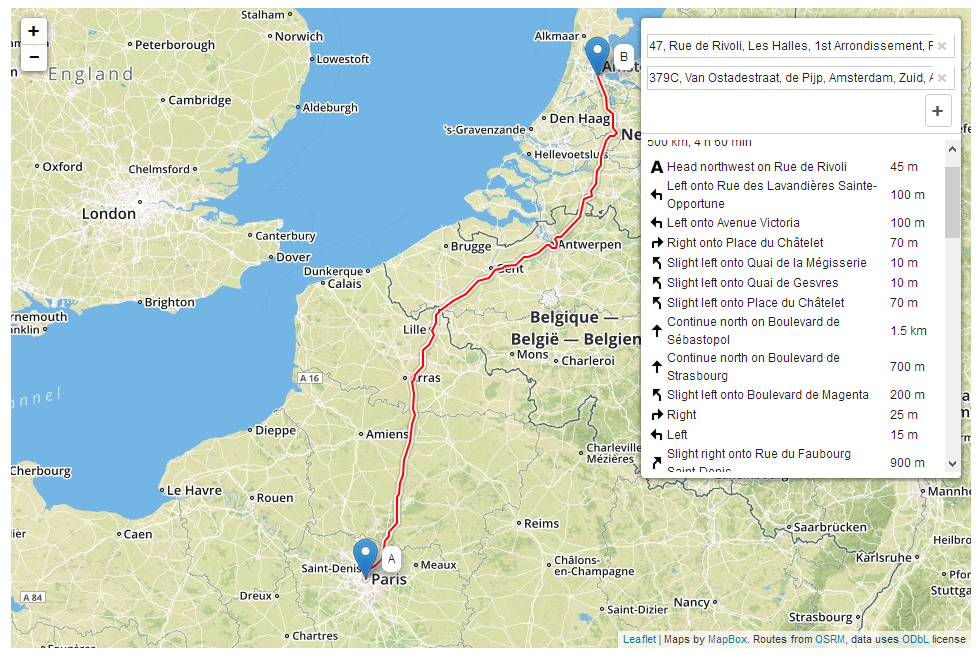
Bibliotēka ļauj ielādēt dažādas kartes un ģeoloģisko datu līmeņus, attēlojot dažādus datus, piemēram, ielu/pilsētu nosaukumus, ceļu stāvokli un noslogojumu. Papildus, izstrādātājiem ir piedāvātas vairāk nekā 100 dažādas papildu bibliotēkas, kartes pilnveidošanai. Lai padarītu kartes ineraktīvākas izstrādātājs, var pievienot marķierus ar dažāda veida informāciju kā tas attēlots 2.8.attēlā.



..att. **Leaflet kartes attēlojuma piemērs**.

### „Leaflet Routing Machine” JavaScript bibliotēka

„Leaflet routing machine” ir paplašinājuma bibliotēka iepriekšējā apakšnodaļā aprakstītajai „Leaflet” bibliotēkai. Tāpat kā „Leaflet” bibliotēka,šī bibliotēka arī ir izstrādāta izmantojot JavaScript bibliotēku, un galvenā tās funkcionalitāte ir nodrošināt iespēju attēlot maršrutu. Lai būtu iespējama maršruta attēlošana, kartē ir nepieciešams norādīt vismaz divus koordinātes punktus. [38]



..att. **„Leaflet routing machine” maršruta attēlojums.**

Šī bibliotēka atbalsta vairāku maršruta punktu attēlojumu. Papildus tam, kā redzams attēlā numur 2.9. viena no bibliotēkas piedāvātajām funkcijām ir maršruta norāžu attēlošana.

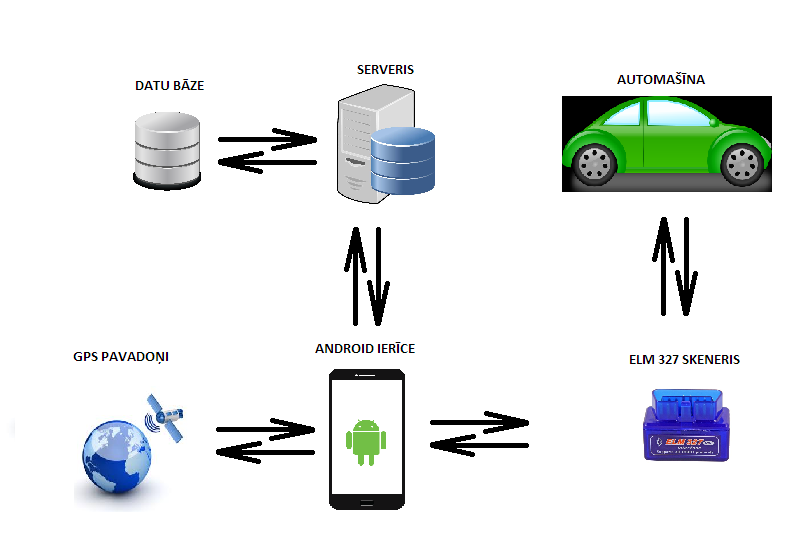
Šī un iepriekšējā bibliotēka tiks izmantota web aplikācijas izstrādē. Pateicoties šīm bibliotēkām, tiks attēlota informācija par transportlīdzekļa kustības stāvokli, būs iespējams apskatīties veikto transportlīdzekļa maršrutu kādā konkrētajā laika posmā, ko izvēlēsies lietotājs, kā arī transportlīdzekļa atrašanās vietu reālajā laikā.

# Programmatūras Izstrāde Un testēšana

Šajā nodaļā tiks aprakstīts risinājuma implementācija, un iespējamās problēmas, ar kurām var saskarties izstrādājot šāda veida programmatūru. Programmatūras izstrāde tiks veikta balstoties uz iepriekšējās nodaļās aprakstītajai zināšanu bāzei, kā arī izmantojot personīgā personīgo pieredzi, strādājot programmēšanas sfērā.

## Sistēmas sastāvdaļu kopējais apraksts

Šī darba ietvaros, darba autors, izstrādāja transportlīdzekļu izsekošanas sistēmu, balstoties iegūtajām zināšanām. Šajā nodaļā tiks aprakstītas pašas svarīgākās sistēmas sastāvdaļas. Sistēmas darbības kopējo shēmu var aplūkot attēlā numur 3.1.



3.1.att. **Kopējā sistēmas darbības shēma**

Sistēmas mezglu savstarpējo darbību ir iespējams aprakstīt diezgan vienkārši. Divas galvenās sistēmas sastāvdaļās ir Web serviss un „Android” vidē izstrādāta aplikācija. Aplikācijas darbības nodrošināšanai ir nepieciešama mobilā ierīce, kurai ir noteikti jāatbalsta 3 tālāk minētās funkcionālās iespējas. Šai ierīcei ir nepieciešams GPS modulis, „Bluetooth” informācijas apmaiņas bezvadu tehnoloģijas ports un mobilā bezvadu interneta atbalsts. Šādu nepieciešamo funkciju klāstu atbalsta lielākā daļa mobilo telefonu, kuriem ir „Android” operētājsistēma un kuri ir ražoti pēc 2011. gada. Papildus, ir nepieciešams ELM 327 OBD-II transportlīdzekļu datordiagnostikas skeneris. „Android” telefonam tika izstrādāta aplikācija, kas caur „Bluetooth” informācijas pārraides portu ir spējīga savienoties ar iepriekš minēto skeneri, un, pateicoties šim, savienojumam iegūst datus no transportlīdzekļa borta datora. Šajā darbā ir plānots iegūt tādus transportlīdzekļa darbības parametrus kā momentānais ātrums, odometra rādījums, vidējais degvielas patēriņš, degvielas bākas aizpildījums, dzinēja apgriezienu skaits, kā arī iespējamo transportlīdzekļa sistēmas darbības traucējumu kļūdu kodus. Paralēli transportlīdzekļa datu iegūšanai, „Android” aplikācija iegūst datus no GPS moduļa. Iegūtie dati ir atrašanās vietas koordinātas (šaurums, platums un augstums), datu iegūšanas laiks, kā arī GPS raidītāja atrašanās vietas datu precizitāte. Iegūstot šos datus, tie tiek nosūtīti uz web servisu, kas saglabā šos datus datubāzes serverī, kā arī apstrādājot tos, izvada tos mājaslapā, gala lietotājam ērti apskatāmā un viegli saprotamā veidā.

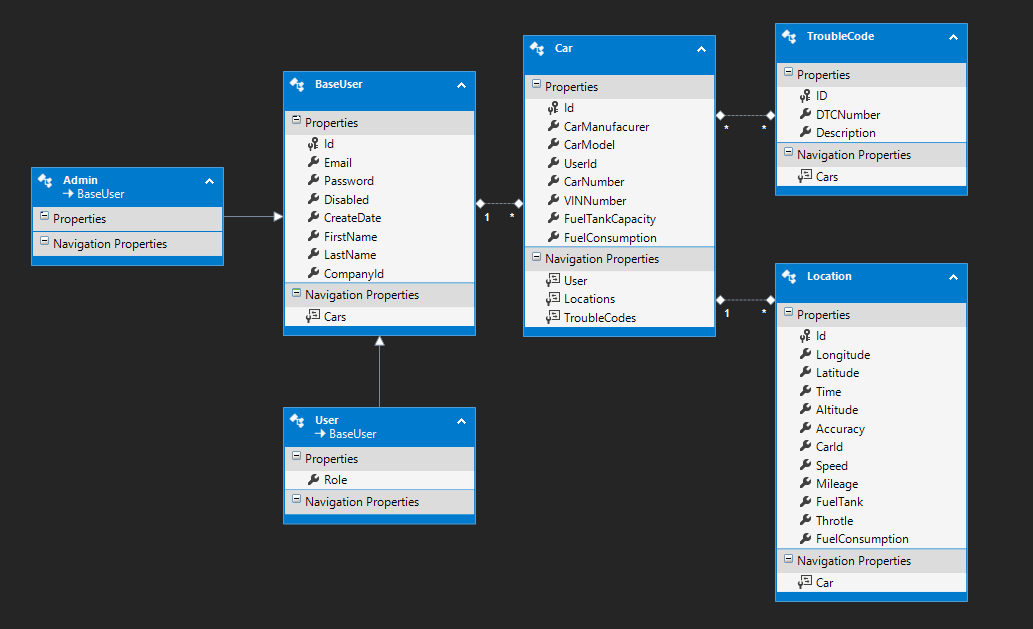
Mājaslapā ir izvietoti grafiki ērtai skaitlisko datu uztverei, kā arī karte, uz kuras ir iespējams novērot transportlīdzekļa pašreizējo atrašanās vietu, kā arī nobrauktā maršruta atainojumu, izvēloties laika periodu. Papildus tam, web aplikācija izvadīs dažāda veida informatīvus grafikus.

Sistēmas darbībai ir nepieciešam datu bāze, kas uzglabā informāciju iegūtu no „Android” aplikācijas.

## Datu bāze

Šajā apakšnodaļā tiks aprakstīta web servisa darbības nodrošināšanai nepieciešamās datu bāzes struktūra.

No 3.2. attēlā redzamās informācijas var redzēt, ka šīs konceptuālās sistēmas izstrādei ir nepieciešamas 4 datu bāzes tabulas. „Admin” un „User” fiziski nav divas atšķirīgas datu bāzes tabulas. Šīs divas entītijas manto „BaseUser” tabulas īpašības, tādejādi, sistēmā var tikt glabāti divu dažādu tipu lietotāji – galvenie sistēmas administratori un lietotāji, kuri savukārt var tikt iedalīti vairākās lomās. „BaseUser” tabula satur standarta lietotāju informāciju. Relācija viens pret vairākiem ar „Car” tabulu nozīmē to, ka katram piereģistrētajam lietotājam ir iespējams piereģistrēt vairākus transportlīdzekļus.



3.2.Att.**Datu bāzes shēma**

„Car” tabulā tiek glabāta pamatinformācija par konkrētu transportlīdzekli, tajā skaitā transportlīdzekļa marka, modelis, Valsts ceļu satiksmes drošības direkcijas reģistrācijas numurs, transportlīdzekļa benzīna bākas tilpums, kā arī transportlīdzekļa ražotāja noteiktais vidējais degvielas patēriņš. Transportlīdzekļa ražotāja noteiktais vidējais degvielas patēriņš ir nepieciešams, lai varētu salīdzināt to ar transportlīdzekļa reālo degvielas patēriņu. Ja gadījumā reālais patēriņš ir lielāks par ražotāja norādīto, tad transportlīdzekļa uzraudzītājam ir jāpievērš pastiprināta uzmanība konkrētā transportlīdzekļa ekspluatācijas nosacījumiem. Transporta degvielas bākas tilpums ir nepieciešams tikai tādēļ, ka ELM 327 datordiagnostikas skeneris spēj nolasīt tikai transportlīdzekļa degvielas bākas aizpildījumu procentos. OBD protokols neatbalsta transportlīdzekļa degvielas bākas kopējā tilpuma noteikšanu.

Katram transportlīdzeklim, kurš pārvietojās pa ceļu ir savs unikālais VIN numurs - saīsinājums no Angļu valodas (Vehical Identification Number). Dažkārt to sauc par šasijas numuru. Šis numurs ir unikāls un tiek piešķirts transportlīdzeklim individuāli - uz pasaules nav divu vienādu transportlīdzekļu ar vienu VIN. Šī iemesla dēļ dažkārt tas tiek salīdzināts ar Personas Kodu vai DNS, kas katram cilvēkam arī ir unikāls un ļauj identificēt konkrēto cilvēku.

Šajā gadījumā izmantojot ELM 327 datordiagnostikas skeneri mēs varam nolasīt transportlīdzekļa VIN numuru un izmantot to kā identifikācijas parametru.

„Trouble code” tabula satur DTC kļūdu kodus un šo kļūdu kodo īsus paskaidrojumus. Šī tabula veido relāciju „daudzi pret daudziem” ar „Car” tabulu, kas nozīmē, ka katram transportlīdzeklim var tikt piešķirti vairāki kļūdu kodi, un katrs kļūdas kods var piemist vairākiem transportlīdzekļiem.

Tabula „Location” satur visus datus, kas tiek sūtīti no „Android” ierīces. Sākumā šie dati nonāk uz serveri, kur tie tiek apstrādāti un pārveidoti nepieciešamajā formātā. Pēc datu pielāgošanas serveris saglabā datus datu bāzē.

## „Android” izsekošanas sistēmas aplikācija

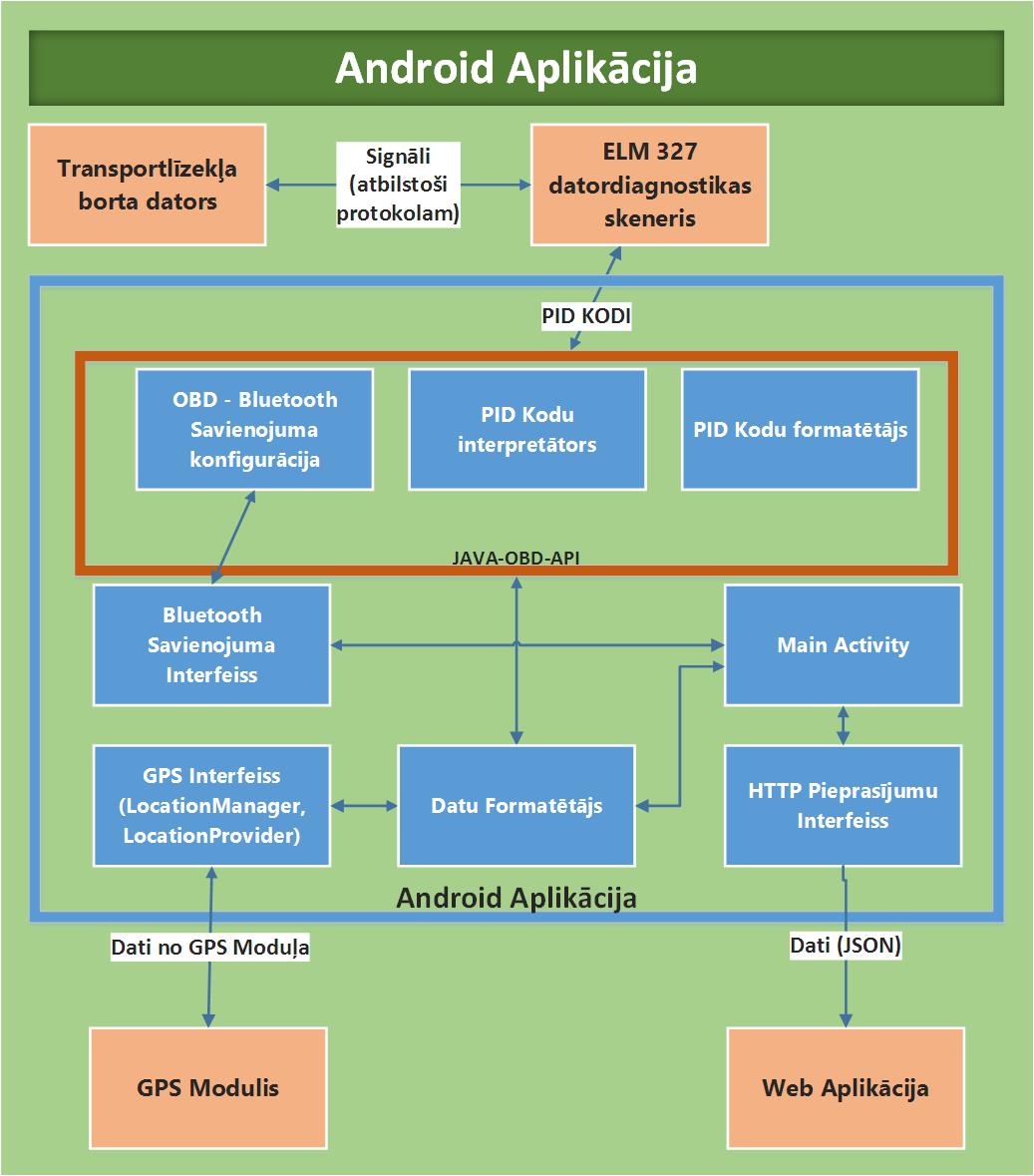
Šī bakalaura darba ietvaros, tika izstrādāta aplikācija Java valodā, kura darbojas „Android” vidē. Šīs Aplikācijas darbībai ir nepieciešams nodrošināt GPS moduļa, „Bluetooth” informācijas apmaiņas bezvadu tehnoloģijas porta un mobilā bezvadu interneta pieejamību.

Šādu nepieciešamo funkciju klāstu atbalsta lielākā daļa mobilo telefonu, kuriem ir „Android” operētājsistēma un kuri ir ražoti pēc 2011. gada.

Papildus, tika uzprogrammēta sasaiste ar ELM 327 OBD-II transportlīdzekļu datordiagnostikas skeneri. „Android” telefonam tika izstrādāta aplikācija, kas caur „Bluetooth” informācijas pārraides portu ir spējīga savienoties ar iepriekš minēto skeneri, un, pateicoties šim, savienojumam iegūst datus no transportlīdzekļa borta datora.

Šajā darbā ar „Android” transporta izsekošanas aplikāciju ir plānots iegūt tādus transportlīdzekļa darbības parametrus kā momentānais ātrums, odometra rādījums, vidējais degvielas patēriņš, degvielas bākas aizpildījums, dzinēja apgriezienu skaits, kā arī iespējamo transportlīdzekļa sistēmas darbības traucējumu kļūdu kodus.

Paralēli transportlīdzekļa datu iegūšanai, „Android” aplikācija iegūst datus no GPS moduļa. Iegūtie dati ir atrašanās vietas koordinātas (šaurums, platums un augstums), datu iegūšanas laiks, kā arī GPS raidītāja atrašanās vietas datu precizitāte. Iegūstot šos datus, tie tiek nosūtīti uz web servisu, kas saglabā šos datus datubāzes serverī, kā arī apstrādājot tos, izvada tos mājaslapā, gala lietotājam ērti apskatāmā un viegli saprotamā veidā.



..att. „**Android” Aplikācijas blokshēma**

Attēlā numur 3.3. ir redzama „Android” izsekošanas aplikācijas blokshēma. Blokshēmā ir attēlota kopēja aplikācijas struktūra. Aplikācijas savienojumu ar OBD skeneri nodrošina „JAVA-OBD-API” bibliotēka. Tā ģenerē PID kodus, nosūtīšanai uz skeneri, kā arī apstrādā saņemtos no skenera PID kodus. Ar to nodarbojas PID kodu interpretators. PID kodu formatētājs formatē saņemtos datu dažādās mērvienībās. OBD Savienojums tiek izveidots pateicoties „Android” „Bluetooth” interfeisam, un šī savienojuma galējā konfigurācija notiek „JAVA-OBD-API” bibliotēkā. Galvenajā aktivitātē tiek veidoti savienojumi ar GPS, HTTP, Bluetooth interfeisiem un „JAVA-OBD-API” bibliotēku. Pirms datu nosūtīšanas pa HTTP interfeisu – tie tiek formatēti atbilstoši pieprasījumiem un formatēti JSON formātā. 2.pielikumā ir iespējams novērot dažādus aplikācijas koda fragmentus.



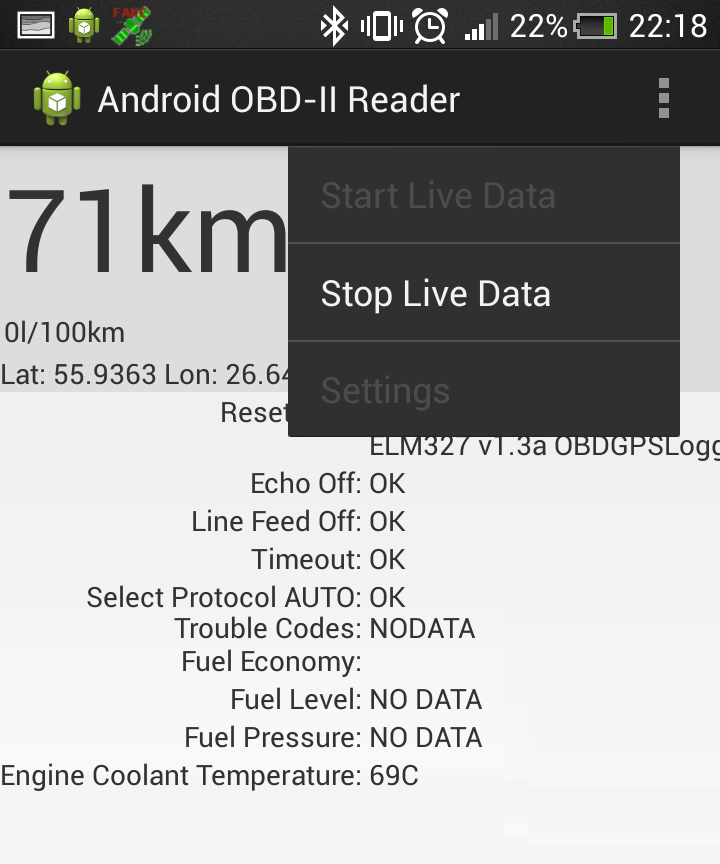
3.4.att.**Aplikācijas galvenais logs**

Attēlā 3.3. ir attēlota aplikācija galvenais skats. Šajā skatā tiek attēlota visa ievāktā informācija no GPS un ELM 327 moduļa. Informācija šajā skatā patstāvīgi tiek atjaunota, un atjaunošanās biežumu ir iespējams uzstādīt aplikācijas iestatījumos.

Šis skats izvada visus nepieciešamos datus, piemēram, pašreizējo ātrumu, dzinēja apgriezienu skaitu, momentāno degvielas patēriņu, atrasēšanās vietas koordinātas. Attēlā numur 3.3. nav attēlots transportlīdzekļa odometra rādījums. Šī problēma ir radusies tādēļ, ka, šī attēla uzņemšanas laikā, sistēma darbojās simulācijas režīmā izmantojot „OBD Sim” aplikāciju, kura neatbalsta transportlīdzekļa odometra rādījumu simulāciju. Testējot aplikāciju reālā darba vidē (nolasot datus no eksistējoša transportlīdzekļa) šie dati tika sekmīgi iegūti.

Šajā skatā tiek izvadīti arī kļūdu kodi, kas rodas transportlīdzekļa noteiktu agregātmezglu bojājumu, vai arī funkcionālo traucējumu gadījumā. Kļūdu kodu paskaidrojumi netiek izvadīti šajā skatā, bet tie tiek nosūtīti uz serveri un attēloti web aplikācijā, iekļaujot kļūdu kodu paskaidrojumus.

Komandas „Reset OBD”, „Line Echo Off”, „Timeout”, „Line Echo Off”, arī netiek nosūtītas uz serveri. Šīs komandas attēlo OBD-II porta sasaistes konfigurācijas datus. „Reset OBD” komandas izpildes rezultātā tiek izvadīts skenera modeļa nosaukums, kā arī skenera programmatūras versija. „Line Echo Off”, „Timeout”, „Line Echo Off” komandu nozīme un izvadītie dati tika aprakstīti nodaļā 2.5.



3.5.att.**Aplikācijas izvēlne**

Aplikācijas izvēlnē lietotājam tiek piedāvātas trīs izvēles iespējas – „Start Live Data”, „Stop Live Data” un „Settings”. Šī izvēlne vizuāli ir atrādīta 3.4. attēlā.

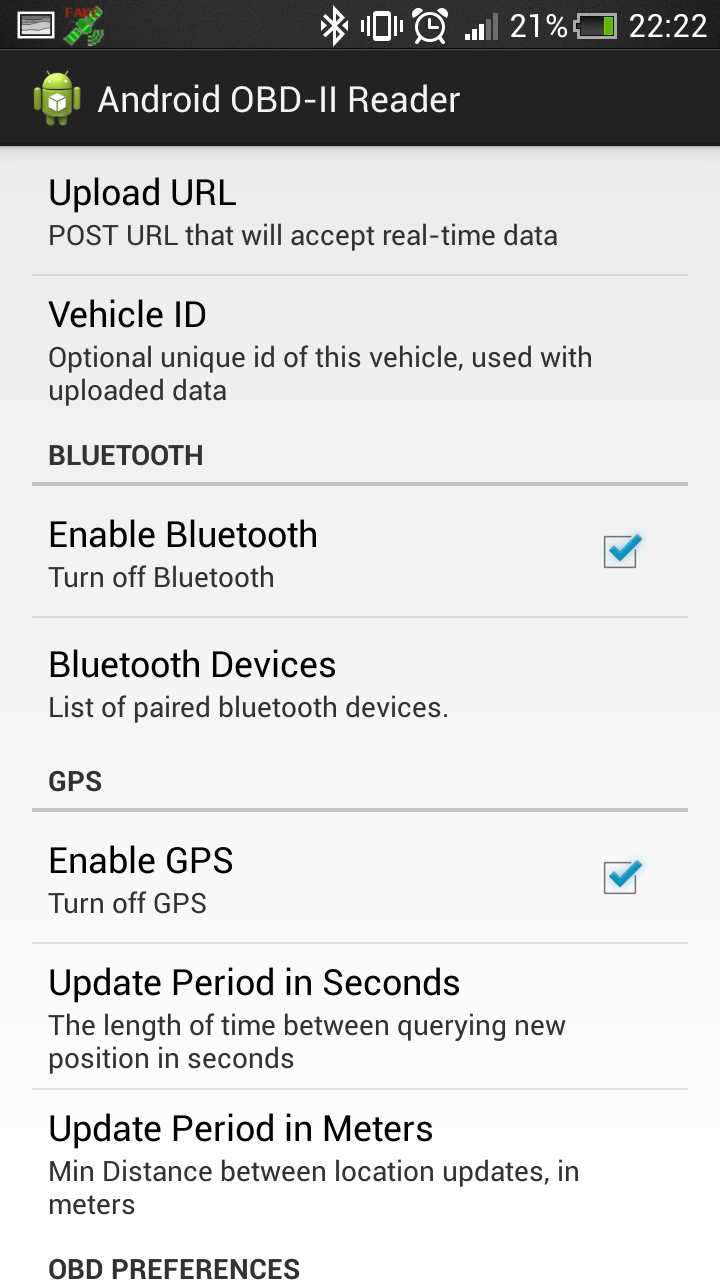
Aplikācijas lietotājam zvēloties „Start Live Data” izvēlni, Aplikācija sāks datu ieguvi no autodiagnostikas skenera, kā arī no GPS moduļa. Paralēli tam, iegūtie dati tiek sūtīti uz serveri, kur tie tiek saglabāti tālākai apstrādei. Lai šī funkcionalitāte darbotos bez problēmām, lietotājam, pirms tas uzsāk darbību, ir jāsavieno ierīce, uz kuras darbojas šī aplikācija, ar skeneri izmantojot „Bluetooth” savienojumu.

Paralēli aplikācijas savienojumam ar skeneri lietotājam ir jānodrošina interneta savienojuma pieejamība. Šajā konkrētajā situācijā, vadoties pēc autora personiskajiem uzskatiem un pieredzes, pats piemērotākais interneta savienojuma veids ir mobilais internets, vai arī GPRS savienojums ar tīklu. No tā izriet, ka visērtākais ierīces veids, kas ir derīgs savienojumam ar internetu, varētu būt mobilais telefons vai arī planšetdators ar 3G tīkla pārraides iespējām. Ieteikums lietotājam ir izpētīt mobilo sakaru operatoru piedāvājumus, kuri attiecas tieši uz mobilā tīkla nodrošināšanu un izmaksām, un izvēlēties konkrētajā situācijā piemērotāko pakalpojumu.

Kā pēdējais no faktoriem, sekmīgai datu apmaiņai, par kuru ir jāparūpējas aplikācijas lietotājam, ir GPS moduļa darbības nodrošināšana. Izpildot visas 3 iepriekšminētās prasības, tiek garantēta aplikācijas stabila darbība.

Lietotājam izvēloties „Stop Live Data” funkciju, datu ieguve, apstrāde un nosūtīšana uz serveri tiks apstādināta.

Lietotājam izvēloties „Settings” izvēlni, aplikācija inicializēs jaunu skatu, kurā lietotājs varēs veikt izmaiņas programmatūras darbība izmainot dažādus nosacījumus programmas iestatījumos.



..att.**Aplikācijas iestatījumi Nr.1**

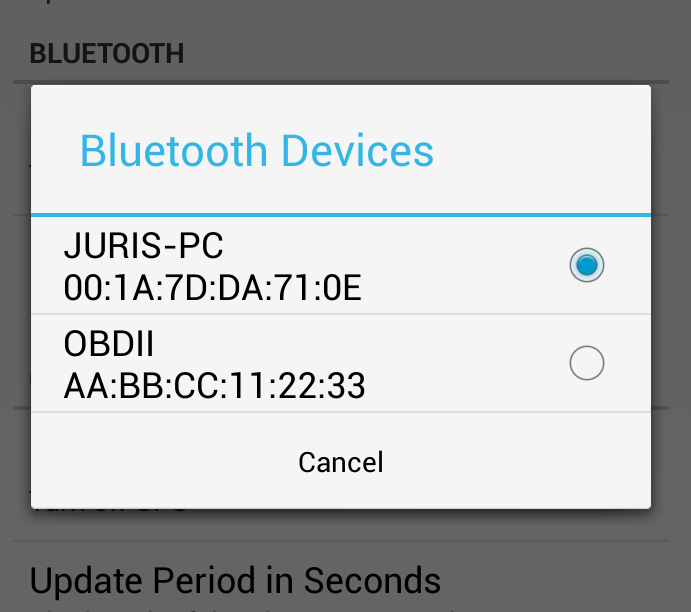
Attēlā 3.5. ir attēlots programmas iestatījumu skats, kuru iespējams inicializēt, izvēloties „Settings” izvēlni galvenajā aplikācijas izvēlē.

„Upload URL” izvēlne ļauj lietotājam patstāvīgi izvēlēties saikni, uz serveri, uz kuru tiek sūtīti dati. Šī saikne galvenokārt tika izmantota aplikācijas izstrādes laikā, kad darba autoram bija radusies nepieciešamība pārslēgties starp dažādiem serveriem. Reālajā situācijā saiknes izmaiņas būtu nepieciešamas tikai tādā gadījumā, ja katra kompānija izvietotu web servisu uz sava servera. Ja web serviss tiek bāzēts kādā kopējā, visiem lietotājiem kopējā serverī tad saite varētu būt konstanta.

„Vehicle ID” izvēlne ļauj ievadīt transportlīdzeklim serverī piešķirto identifikācijas numuru, lai, nosūtot datus uz serveri, serveris varētu noteikt, kuram lietotāju grupas transportlīdzeklim pieder šie dati. Šis identifikācijas numurs tiek izmantots gadījumā, ja transportlīdzeklis, no kura tiek nolasīti dati, neatbalsta transportlīdzekļa VIN numura noteikšanu. Daudzi transportlīdzekļu ražotāji bloķē VIN numura noteikšanu dažādu drošības apsvērumu dēļ. Pēc darba autora domām, VIN numura izmantošana identifikatora vietā ir labāks risinājums, jo šādā gadījumā tiktu nodrošināta lielāka drošība tādā ziņā, ka lietotājam, kurš sekotu transportlīdzekļa darbībai web aplikācijā, būtu pilnīgi skaidrs, ka dati nāk no nepieciešamā transportlīdzekļa, jo lietotājs nespētu to izmainīt. Kā jau iepriekš tika minēts, katram transportlīdzeklim, neatkarīgi no markas un ražotāja, ir savs personīgais identiskais VIN numurs.

Sadaļā „Bluetooth” lietotājam ir piedāvātas 2 konfigurācijas iespējas – „Enable Bluetooth” un „Bluetooth devices”. Atzīmējot „Enable Bluetooth” aplikācija aktivizēs Bluetooth funkcionalitāti, kā arī tā tiks automātiski aktivizēta katrā aplikācijas palaišanas reizē.

„Bluetooth devices” sadaļā lietotājs var izvēlēties bloetooth ierīci, ar kuru tas grib savienoties, kā tas parādīts 3.6 attēlā.



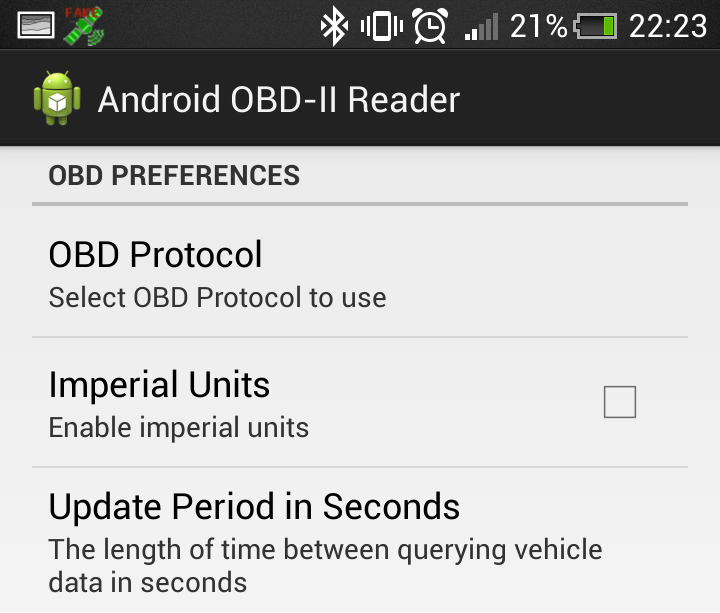
..att.**”Bluetooth” savienojuma izvēle**

Atšķirt OBD-II savienojumu ir salīdzinoši vienkārši – tās nosaukumā tiks iekļauti vārdu salikums „OBD”, vai arī „ELM 327”. Savienojuma laikā lietotājam var tikt pieprasīts ievadīt drošības kodu savienojuma noslēgšanai. Šis kods, lielākajai daļa ierīču ir – 0000, 1234 vai arī 6789. Lai nezaudētu savienojumu starp ierīcēm, attālums starp tām nedrīkst būt lielāks par 10 metriem.

Sadaļā „GPS” lietotājam tiek piedāvātas 3 konfigurācijas iespējas. „Enable GPS” funkcija līdzīgi kā „Bluetooth” sadaļā aktivizē vai deaktivizē GPS moduļa darbību. Ja šī funkcija ir aktivizēta, tā aktivizēs un deaktivizēs GPS moduli automātiski katru reizi, kad aplikācija tiks ieslēgta vai izslēgta. Pēc autora domām tas ir ļoti svarīgi, jo „Bluetooth” un GPS moduļi izmanto lielu elektroenerģijas daudzumu un, nejauši aizmirstot tos izslēgt, ierīces akumulators izlādēsies ļoti ātri.

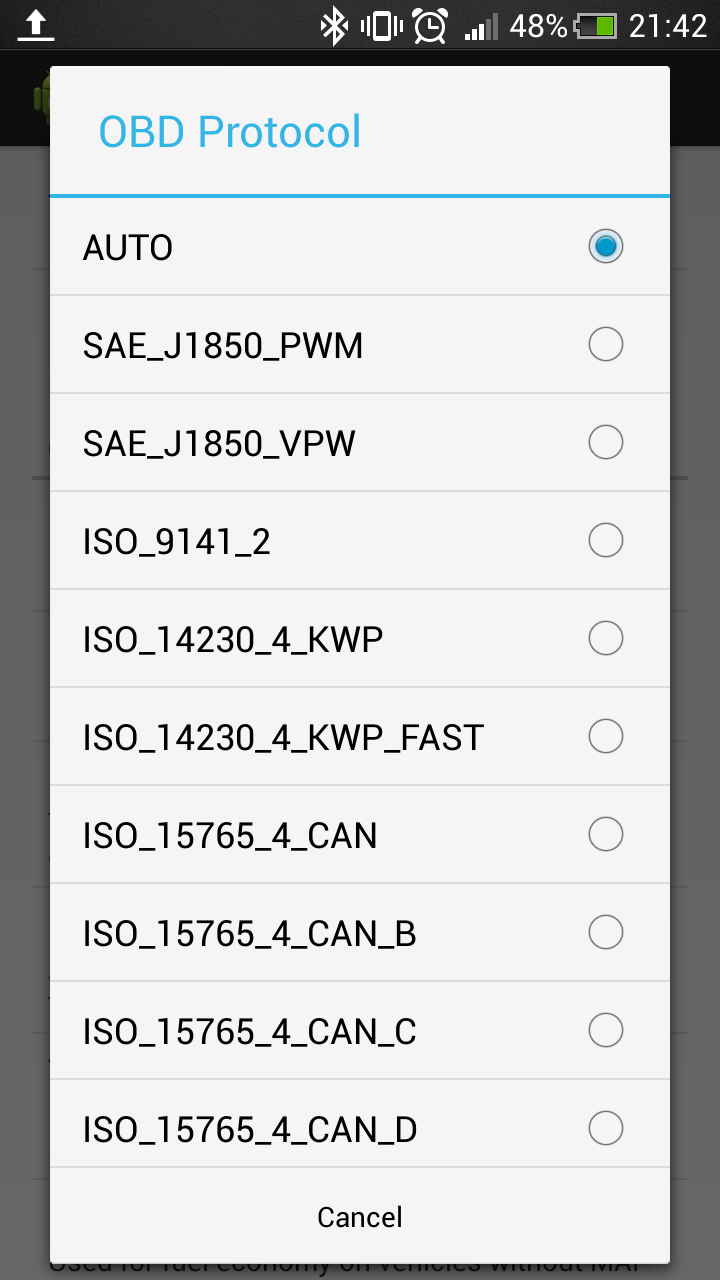
„Update period in seconds” apakšsadaļā lietotājs var iestatīt, GPS moduļa datu atjaunošanas laika intervālu. Jo lielāks intervāls tiks iestatīts, jo mazāka būs datu apmaiņa ar serveri, respektīvi, izmaksas par datu apmaiņu arī tiks samazinātas, bet līdz ar to samazināsies iegūto datu kvalitāte. Darba autors iesaka izvēlēties GPS moduļa atjaunošanās laika periodu intervālā no 5 līdz 15 sekundēm.

„Update period in meters” ļauj atjaunot GPS rādījumus pēc noteiktas distances izmaiņas. Šī parametra konfigurācija ir salīdzinoši labi pielietojama pilsētas apstākļos, kad transportlīdzeklis atrodas sastrēgumā, vai arī atrodas lēnā kustībā. Šāda veida situācijā aplikācija nosūtīs apstrādei uz serveri nelietderīgus datus, kas savukārt apgrūtinās šo datu uztveršanu gala lietotājam. Respektīvi, iepriekš aprakstītās situācijas gadījumā, šī funkcija samazinās arī datu apmaiņas apjomu, kas savukārt novedīs pie naudas līdzekļu ekonomijas.



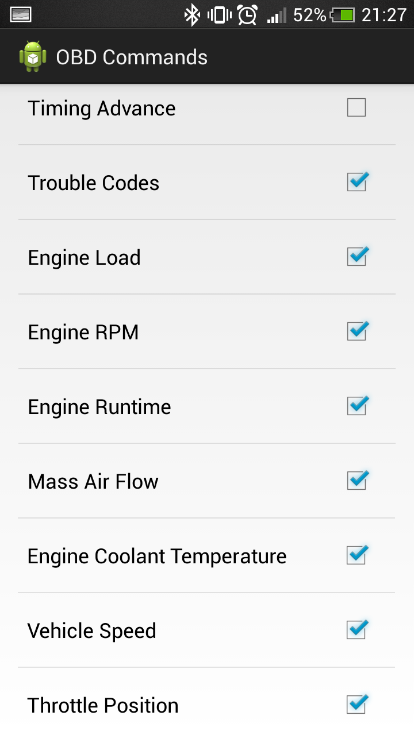
3.8.att. **Aplikācijas iestatījumi Nr.2**

OBD iestatījumu sadaļā, kas ir izsmeļoši ilustrēta 3.7. attēlā, ir trīs konfigurācijas apakšnodaļas.



3.9.att.**OBD-II protokolu saraksta fragments**

„OBD Protocol” apakšnodaļā ir iespējams izvēlēties OBD-II protokolu, kurš tiks lietoti savienojumam ar ELM 327 skeneri. Daļējs OBD-II protokolu saraksts ir attēlots attēlā 3.8. Darba autors iesaka izmantot automātisko protokola noteikšanu. Šī funkcija savienojuma laikā pārbauda visu protokolu atbalstu un izvēlas to, kurš tiek atbalstīts konkrētajā transportlīdzeklī.



3.10.att.**OBD komandu saraksta fragments**

„Imperial units” apakšnodaļā mēs varam izvelēties, kādā mērvienību sistēmā tiks attēloti iegūtie dati. Šī opcija neietekmē datus, kas tiek sūtīti uz serveri. Visi dati, kas tiek sūtīti no aplikācijas uz serveri, ir attēloti metriskajā sistēmā.

„Update periods in seconds” opcija, ļauj lietotājam izvēlēties intervālu, ar kuru tiks nolasīti dati no OBD-II skanera. Skenera datu atjaunošanas periodu ir ieteicams izvēlēties vienādu ar GPS moduļa datu atjaunošanās periodu.

Papildus iestatījumos ir iespējams izvēlēties komandas, kuras tiks attēlotas galvenajā programmas skatā. Daļējs izvadāmo datu saraksts ir attēlots 3.9. attēlā. Šī funkcija ir ļoti noderīga transportlīdzekļa OBD-II funkcionalitātes diagnostikai, lai varētu laicīgi noteikt, kādus datus ir iespējams nolasīt un kādu datu nolasīšanu transportlīdzeklis neatbalsta.

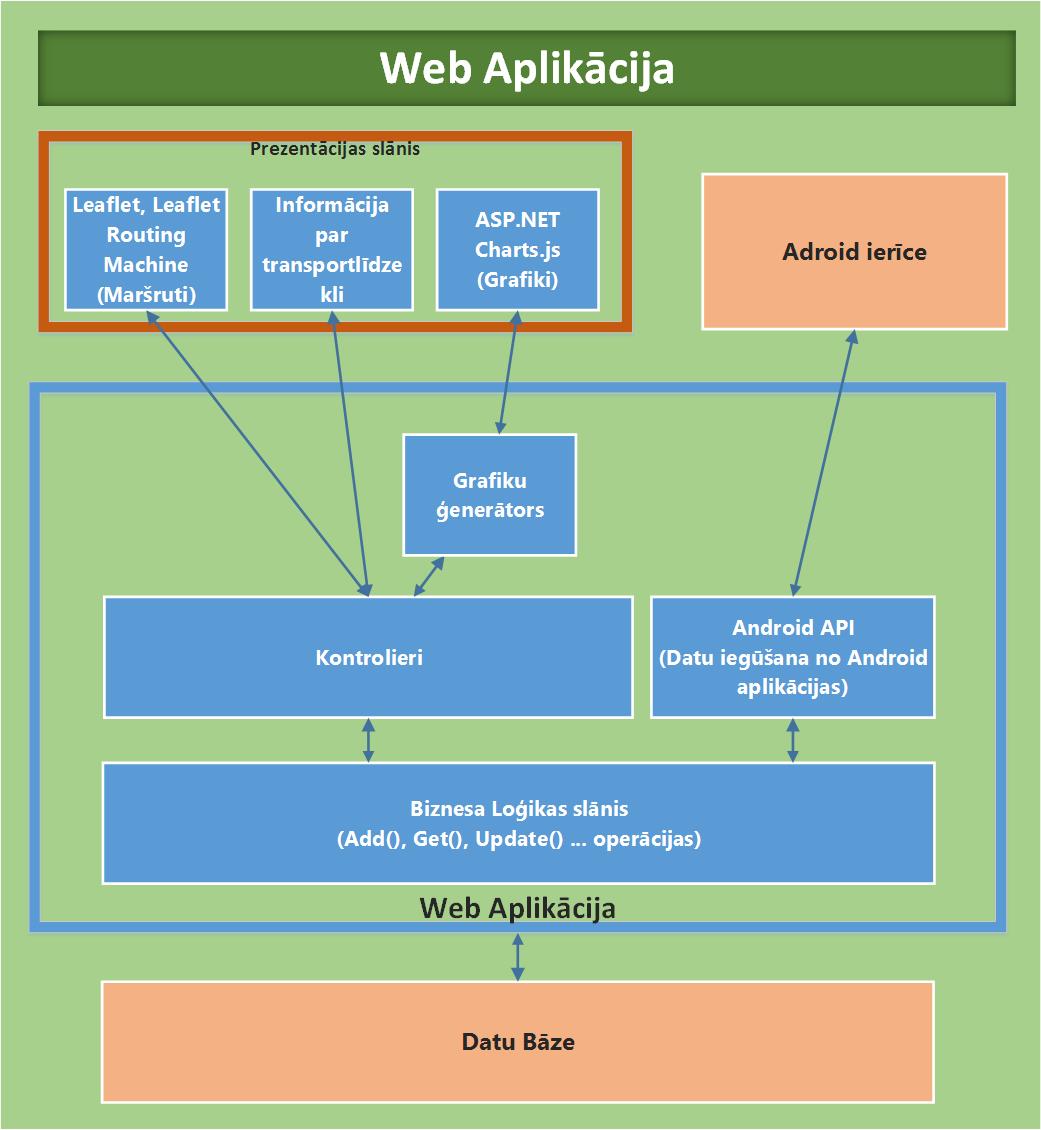
Aplikācijas savienojums ar ELM 327 tika iegūts pateicoties atvērtā koda bibliotēkai „OBD-JAVA-API”. Šī bibliotēka tika izveidota JAVA programmēšanas valodā un tādēļ, to veiksmīgi varēja implementēt aplikācijā, jo „Android” operētājsistēma un aplikācijas, kas darbojas tajā atbalsta JAVA programmēšanas valodu. Bibliotēka padarīja vieglāku PID kodu pārvēršanu bitu simbolu rindā, kā arī bibliotēka apstrādāja no ELM 327 saņemto datu plūsmu un pārveidoja to cilvēkam saprotamā formātā.

Datu nosūtīšana uz serveri tika īstenota izmantojot HTTP Post pieprasījumus. Pirms tam kā aplikācija sūta datus uz serveri, tā pārveido tos JSON formātā. JSON (JavaScript ObjectNotation) datu formāts tika izmantots, sava kompaktuma dēļ, tādejādi samazinot sūtāmo datu apjomu un, respektīvi, interneta savienojuma trafika izmantošanu.

## Web aplikācija transportlīdzekļa izsekošanai

Šī darba ietvaros tika izstrādāta web aplikācija, datu apmaiņas un apstrādes, lietotājam pieņemamā un viegli saprotamā formā, nodrošināšanai. Aplikācija ne tikai apstrādā saņemtos datus izvadot tos lietotājam saprotamā veidā, galvenokārt grafiku un karšu veidā, bet arī nodrošina savienojumu ar lietotāja ierīci, uz kuras atrodas iepriekšējā nodaļā aprakstītā aplikācija, un informācijas apmaiņu ar to un datu saglabāšanu datu bāzē.

Aplikācija vienlaikus var apstrādāt vairākus lietotāju profilus un katrā profilā ir iespējams piereģistrēt vairākus transportlīdzekļus.



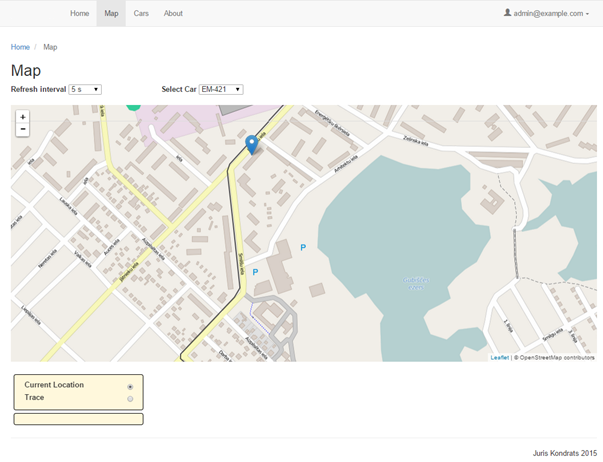
..att. **Web Aplikācijas Blokshēma**

Attēlā numur 3.3. ir redzama Web aplikācijas blokshēma. Blokshēmā ir attēlota kopēja aplikācijas struktūra. Aplikācijas pamata loģika slēpjas biznesa slānī. Šajā slānī tiek veiktas visas algoritmiskās un loģiskās darbības, kā arī datu apmaiņu ar datu bāzi. Web aplikācijai ir četras galvenās pamatfunkcijas – datu saņemšana, datu saglabāšana, datu apstrāde un datu izvade lietotājam saprotamā veidā.

Aplikācija apstrādā saņemtos izsekošanas datus no android ierīces izmantojot Android API interfeisu. Šis interfeiss saņemtos datus saglabā datu bāzē, izmantojot biznesa loģikas slāni.

Biznesa loģikas slānis atbild gan par saņemto no „Android” aplikācijas datu saglabāšanu, gan par Web interfeisa datu saglabāšanu (lietotāju informācija, transportlīdzekļu pamatinformācija, u.c.) un apstrādi.

Kontrolieri veido slāni starp biznesa loģikas slāni un Web interfeisu. Tas ir kā tilts starp datu plūsmu no web interfeisa uz biznesa loģiku un otrādi. Lai aplikācijā varētu veiksmīgi attēlot dažādus grafikus tiek izmantots Grafiku ģenerators. Grafiku ģenerēšanai tiek izmantota bezmaksas „HighCharts” bibliotēka. Aplikācijas koda fragmentus ir iespējams apskatīties 3. pielikumā.

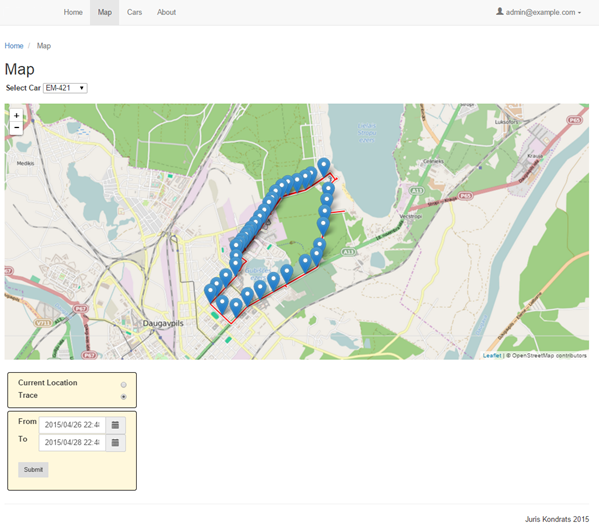


3.12.att.**Pašreizējā atrašanās vietas skats**

Attēlā 3.10. ir attēlota pašreizējā transportlīdzekļa atrašanās vieta. Ir jāprecizē, ka šī atrašanās vieta ir pēdējā transportlīdzekļa nolasītā transportlīdzekļa atrašanās vieta. Lai attēlotu šo informāciju reālajā laikā, lietotājam ir nepieciešams nodrošināt pastāvīgu datu savienojumu ar serveri. Šajā gadījumā ir nepieciešams izvērtēt un izvēlēties piemērotāko mobilā interneta provaideru un atbilstošu tarifu plānu. Mūsdienās, pēc darba autora personīgajiem novērojumiem, apskatot lielākās kompānijas, kas nodrošina 3G tīkla pieejamību Latvijā, ir vērojama tendence piemērot abonēšanas maksu mobilā interneta lietošanai. Tādā veidā ir iespējams nodrošināt pastāvīgu mobilās aplikācijas savienojumu ar serveri.

Kā vēl vienu transportlīdzekļa atrašanās vietas reālajā laikā noteikšanas variantu var minēt SMS servisu. Šis serviss ir balstīts uz to, kā lietotājs var nosūtīt pieprasījumu, uz ierīci, kurā ir ievietota SIM karte un automātiski saņemt atpakaļ atbildi ar pašreizējās atrašanās vietas koordinātēm, bet šajā darbā netiks apskatītas šāda veida funkcionālās iespējas.

Kā redzams attēlā 3.10. Lietotājs var izvēlēties starp viņa profilā piereģistrētājiem transportlīdzekļiem, kā arī izvēlēties laika intervālu, ar kuru atjaunosies dati.



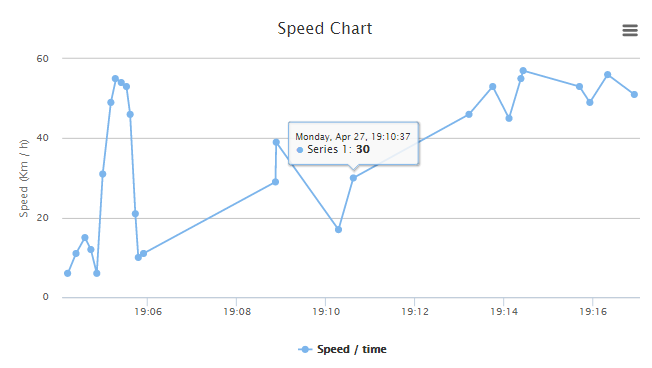
..att.**Maršuta attēlošanas skats**

Attēlā numur 3.11. ir iespējams novērot transportlīdzekļa pārvietojuma maršrutu noteiktajā laika periodā. Datu attēlojuma laika periodu ir iespējams izvēlēties, norādot nepieciešamos sākuma un beigu datumus.

Tāpat kā iepriekš apskatītajā skatā, lietotājam ir iespēja izvēlēties, transportlīdzekli, kurš ir piereģistrēts konkrētā lietotāja profilā.

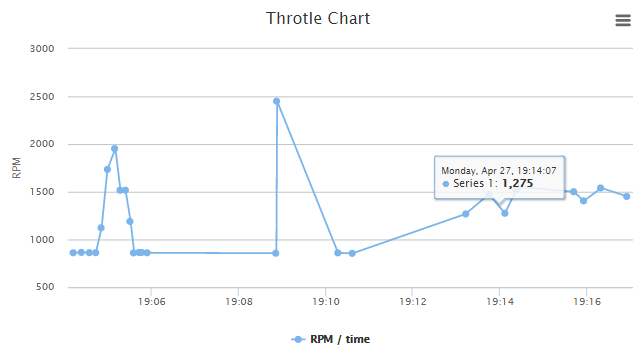
Attēlā 3.10 un attēlā 3.11 kartes attēlošanai tika izmantota „Leaflet” bibliotēka, kas ir uzrakstīta JavaScript programmēšanas valodā. Šī GIS sistēma ir pielietojama bezmaksas.

Attēlā 3.11 papildus „Leaflet” bibliotēkai tika izmantota „Leafleat-routing-machine” bibliotēka, kas nodrošina transportlīdzekļa pārvietojuma maršruta attēlošanu kartē. Tāpat kā „Leaflet” bibliotēka – šī bibliotēka ir uzrakstīta JavaScript programmēšanas valodā un tā arī ir pieejama brīvā bezmaksas lietošanā.



3.14.att. **Ātruma grafiks izvēlētajā laika periodā**

Attēlā 3.12 ir redzams viens no grafikiem, kas tika izvadīts informācijas apkopošanai un pārveidošanai lietotājam vieglāk uztveramā formā. Šis grafiks attēlo ātruma izmaiņas atkarībā no laika perioda. Lai uzzinātu precīzu ātrumu kādā no konkrētajiem punktiem, lietotājs var uzvest peles kursoru uz vēlamo punktu un nolasīt precīzu ātruma un laika vērtību. Grafiks ļauj lietotājam izanalizēt transportlīdzekļa vadītāja braukšanas paradumus. Tas ļauj noteikt, vai transportlīdzekļa vadītājs pārkāpj ātruma ierobežojumus, vai arī pārvietojas pārāk lēni.

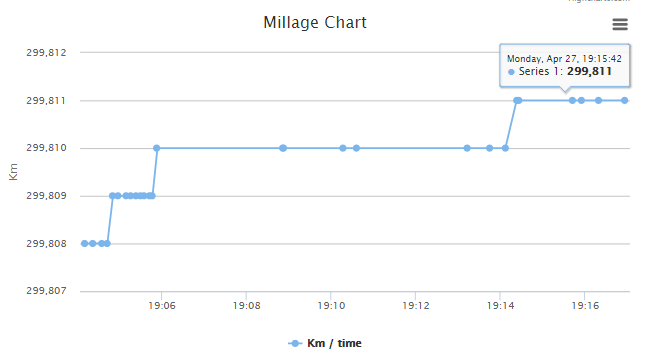


3.15.att. **Dzinēja apgriezienu skaits izvēlētajā laika periodā**

Attēlā 3.13, tāpat kā attēlā 3.11 ir redzams viens no grafikiem, kas tika izvadīts informācijas apkopošanai un pārveidošanai lietotājam vieglāk uztveramā formā. Šis grafiks attēlo dzinēja apgriezienu skaitu izvēlētajā laika periodā. Lai uzzinātu precīzu dzinēja apgriezienu skaitu kādā no konkrētajiem punktiem, lietotājs var uzvest peles kursoru uz vēlamo punktu un nolasīt precīzu dzinēja apgriezienu skaitu un laika vērtību. Zinot dzinēja apgriezienu skaita rādītājus, lietotājs var noteikt un izanalizēt transportlīdzekļa vadītāja braukšanas paradumus.

Palielināts dzinēja apgriezienu skaits liecina par agresīvu braukšanu, kā arī paaugstināts apgriezienu skaits kaitē transportlīdzekļa dzinējam un transmisijai, jo tādā gadījumā paātrinās dažādu motora un transmisijas detaļu nolietošanās, respektīvi, palielinās transportlīdzekļa apkalpošanas izmaksas.

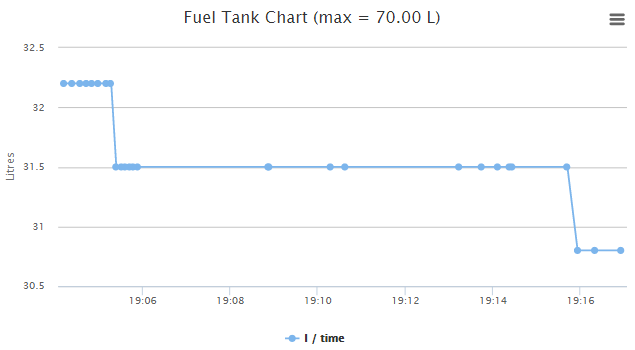
Zinot šos datus, sistēmas lietotājs var izdarīt secinājumus un aprunāties ar transportlīdzekļa vadītāju sniedzot tam padomus, vai arī pielietot citus pasākumus transportlīdzekļa darba kvalitātes uzlabošanai.



3.16.att.**Noskrējiena grafiks izvēlētajā laika periodā**

Attēlā 3.14 ir redzams viens no grafikiem, kas tika izvadīts informācijas apkopošanai un pārveidošanai lietotājam vieglāk uztveramā formā. Šis grafiks attēlo transportlīdzekļa noskrējienu izvēlētajā laika periodā. Lai uzzinātu precīzu noskrējienu kādā no konkrētajiem punktiem, lietotājs var uzvest peles kursoru uz vēlamo punktu un nolasīt precīzu noskrējiena un laika vērtību.

Šis grafiks ļauj lietotājam pārliecināties, ka transportlīdzekļa odometra rādītāji netiek sagrozīti. Patstāvīgi augošs grafiks liecina par to. Daudziem transportlīdzekļiem ir iespējams samazināt odometra rādītājus, tādejādi noslēpjot transportlīdzekļa lietošanu savām vajadzībām.



3.17.att. **degvielas bākas tilpuma grafiks izvēlētajā laika periodā**.

Attēlā 3.15 ir redzams viens no grafikiem, kas tika izvadīts informācijas apkopošanai un pārveidošanai lietotājam vieglāk uztveramā formā. Šis grafiks attēlo transportlīdzekļa bākas tilpuma izvēlētajā laika periodā. Lai uzzinātu precīzu bākas tilpumu kādā no konkrētajiem punktiem, lietotājs var uzvest peles kursoru uz vēlamo punktu un nolasīt precīzu bākas tilpuma un atbilstošu laika vērtību.

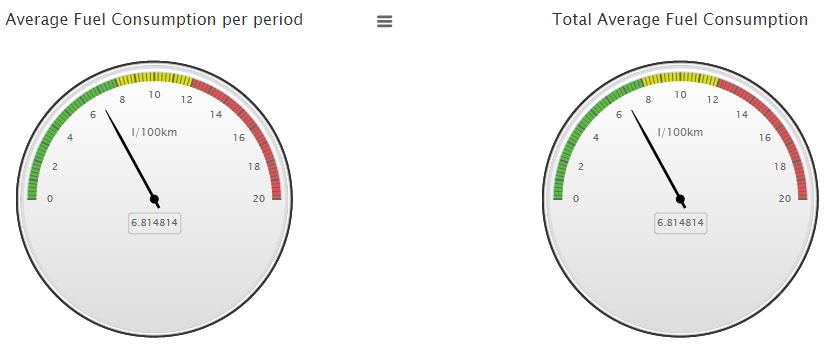
Šajā grafikā lietotājs var rūpīgi sekot degvielas bākas izmaiņām. Pēc autora pieredzes, bieži vien daudzas transporta firmas, saskaras ar problēmu, kad uzņēmumā esošajiem transportlīdzekļiem, ļaunprātīgi tiek nolieta degviela. Šo parādību ir ļoti sarežģīti izsekot, tāpēc šis grafiks varētu ātri un ērti parādīt šāda veida pārkāpumu. Ja grafikā, īsā laika periodā ir ļoti liels degvielas kritums, kas ir lielāks par vidējo degvielas kritumu, lietotājs to varētu saredzēt un tālāk izveidot atbilstošu rīcības plānu. Grafika strauja virzība uz augšu, nozīmē to ka transportlīdzekļa bākā tika pielieta degviela.

Šajā grafikā lietotājs varētu saskatīt arī nelielas svārstības. Iemesls šīm svārstībām varētu būt tas, ka degvielas līmenis, gravitācijas un kustības kinētiskās enerģijas svārstību ietekmē, transportlīdzeklim atrodoties kustībā, var svārstīties nelielā amplitūdā, un šīs svārstības var ietekmēt degvielas līmeņrāža sensoru.



3.18.att. **Vidējais ātrums izvēlētajā laika periodā un visā laika periodā**

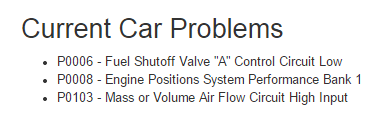
Attēlā 3.16. ir attēlots transportlīdzekļa vidējais ātrums izvēlētajā laika periodā, kā arī kopējais vidējais ātrums. Tādā veidā lietotājs var apskatīt datus par vidējo ātrumu kādā konkrētā laika periodā, un bez problēmām salīdzināt tos ar vidējo braukšanas ātrumu visā laika periodā, un, respektīvi, izdarīt secinājumus, vai vidējais ātrums samazinās, vai arī palielinās, tādejādi, izvērtējot transportlīdzekļa vadītāja darbību, kā arī, izvēlēto maršrutu un pieņemt atbilstošu lēmumu esošās situācijas labošanai.



3.19.att. **Vidējais degvielas patēriņš izvēlētajā laika periodā un visā laika periodā**

Attēlā 3.17. ir attēlots transportlīdzekļa vidējais degvielas patēriņš izvēlētajā laika periodā, kā arī kopējais degvielas patēriņš sākot no transportlīdzekļa reģistrācijas transportlīdzekļa izsekošanas aplikācijā. Tādā veidā lietotājs var apskatīt datus par vidējo degvielas patēriņu kādā konkrētā laika periodā, un bez problēmām salīdzināt tos ar vidējo degvielas patēriņš visā laika periodā. Zinot šos datus, lietotājs var izdarīt secinājumus, vai degvielas patēriņš samazinās, vai arī palielinās, tādejādi, izvērtējot transportlīdzekļa vadītāja darbību, un pieņemt atbilstošu lēmumu esošās situācijas labošanai.

Degvielas patēriņa palielinājumu veicina vairāki faktori, sākot no vadītāja braukšanas manieres beidzot ar riepās esošā gaisa spiedienu, kā arī dažādu dzinēja un transmisijas tehnisko problēmu dēļ.



3.20.att. **Transporta kļūdu kodu saraksta fragments**.

3.19. attēlā ir attēlots transportlīdzekļa darbības kļūdu fragmenta attēlojums reālajā laikā. Tā kā aplikācijas testēšanas laikā reālajā vidē, transportlīdzeklim, uz kura tika testēta aplikācija, nebija konstatētas darbības kļūdas, attēlā redzamās kļūdas tika simulētas pateicoties „OBDSim” aplikācijai.

Redzamajā attēlā tiek izvadīts DTC kļūdas koda numurs, kā arī īss kļūdas koda apraksts kopējās ainas noteikšanai. Izmantojot izvadīto DTC kļūdas kodu ir iespējams atrast plašāku informāciju izmantojot interneta resursus.

# Novērtējums un Rekomendācijas

Šī bakalaura bakalaura darba ietvaros, darba autors patstāvīgi izstrādāja transporta izsekošanas sistēmas konceptu. Šis koncepts sastāv no divām aplikācijām - web aplikācijas un „Android aplikācijas”. Šīs abas aplikācijas apmierina izvirzītās tām prasības, kā arī to funkcionalitāte izpilda noteiktos uzdevumus.

Neskatoties uz to, ka aplikācijas apmierina tām izvirzītās prasības, un aplikāciju testu laikā tās rādija stabilu darbību, autors atzīst, ka ir nepieciešams veikt nelielus uzlabojumus, lai aplikācija varētu būt ērti izmantojama, jebkuram lietotājam. Šie uzlabojumi, galvenokārt, ir nepieciešami tādās jomās, kā lietotāja interfeiss, datu apmaiņas drošība un aplikācijas kļūdu apstrāde. Iepriekš minētie aspekti nebija iekļauti šī bakalaura darba ietvaros.

Aplikāciju izstrāde darba autoram aizņēma 2/3 no visa laika, kas bija patērēts bakalaura darba izveidei. Lielākās problēmas autoram sagādāja „Android” aplikācijas izstrāde, jo darba autoram līdz šim nebija pieredzes mobilo aplikāciju veidošanā „Android” operētājsistēmas vidē.

„Android” izsekošanas aplikācija tika testēta reālajā darba vidē (braucošā transportlīdzeklī). Testa laikā, kas ilga apmēram pus stundu, aplikācija rādīja stabilu darbību. Kritiskas kļūdas, kuras varētu apturēt programmatūras darbību, netika sasniegtas. Visi dati, kas tika nosūtīti uz serveri, veiksmīgi sasniedza to, un tika saglabāti datu bāzē. Web aplikācijā varēja novērot transportlīdzekļa, testa laikā veikto maršrutu, kā arī dažādus informatīvos grafikus, kuri aprakstīti 3.4. apakšsadaļā. Testa laikā iegūto datu fragments ir norādīts 1. pielikumā.

Darba autors šo aplikāciju, galvenokārt, iesaka uzņēmumiem, kuriem ir savs autoparks. Šī aplikācija šāda veida uzņēmumiem dod iespēju rūpīgāk kontrolēt savā īpašumā esošos transportlīdzekļus.

Aplikācija ļauj sekot katram transportlīdzeklim individuāli, nosakot tā tehnisko stāvokli un izmantošanas īpašības. Tādā veidā ir iespējams kontrolēt arī transportlīdzekļa vadītāju. Ar aplikācijas palīdzību ir iespējams kontrolēt, vai transportlīdzeklis tiek lietots ārpus darba vajadzībām. Tā pat ir iespējams kontrolēt, vai netiek nelegāli piesavināti tādi līdzekļi, kā degviela. Zinot datus par transportlīdzekļa lietošanas datus iespējams novērtēt arī transportlīdzekļa vadītāja darbību, lietojot transportlīdzekli, atbilstoši paslavēt vai paaugstināt atalgojumu, vai arī pretējā gadījumā norādīt uz kļūdām vai arī vai arī sniegt padomus, efektivitātes uzlabošanai. Transportlīdzekļa pielietojuma efektivitātes uzlabošana ir ļoti svarīga. Paaugstinot transportlīdzekļa pielietojuma efektivitāti, ir iespējams pagarināt transportlīdzekļa mūžu, kā arī samazināt izmaksas par remontu un degvielu. Pēdējais faktors ir īpaši svarīgs, balstoties uz pašreizējām degvielas cenām.

Tāpat, šo aplikāciju var izmantot arī privātpersonas, sava personīgā vai ģimenes auto kontrolēšanai. Pēc autora domām, tāda veida kontrole ir svarīga gadījumos, kad ģimenē transportlīdzekli lieto kāds no pusaudžiem, kuriem autovadītāja aplikācija ir neilgu laiku. Tādā veidā vecāki var kontrolēt maršrutus, pa kuriem pārvietojas bērni, kā arī kontrolēt ātruma režīmus, un balstoties uz šiem datiem plānot savu rīcību un nostāju.

„Android” aplikācijas darbības nodrošināšanai autors iesaka izmantot mobilo tālruni, uz kura darbojas „Android” operētājsistēma ar versiju ne mazāku paar 2.3(Gingerbread). Lai nodrošinātu nepārtrauktu savienojumu ar internetu, autors iesaka izvēlēties mobilo sakaru operatoru, kurš piedāvā bezlimita datu plūsmu. Kā OBD-II skeneri, darba autors iesaka izvēlēties ELM 327 skeneri. Šī skenera cena ir, sākot ar 5 dolāriem. Tāpat būs darbības laikā būs nepieciešams savienojums ar transportlīdzekļa elektroenerģijas kontaktu, jo aplikācija izmanto ļoti daudz portu, un moduļu, kas savukārt prasa lielu enerģijas daudzumu, līdz ar to telefonā pieejamais akumulators (ja telefons nav pieslēgts ārējam barošanas tīklam) var ļoti ātri izlādēties.

# SECINĀJUMI

Bakalaura darba ietvaros tika izpētīta un izanalizitēta Globālās Pozicionēšanas Sistēmas datu iegūšanas iespējas izmantojot mobilo ierīci, kurā ir iebūvēts Globālās pozicionēšanas sistēmas signāla uztveršanas modulis. Paralēli tam, tika izpētīta un aprakstīta Globālās Pozicionēšanas Sistēmas darbības principi un signālu aprite starp virszemes pavadoņiem un Globālās pozicionēšanas sistēmas signālu uztveršanas moduļiem.

Pēc tam, kad tika izpētīta Globālās Pozicionēšanas Sistēmas darbība un tās implementācijas iespējas „Android” operētājsistēmas aplikāciju vidē, darba autors izpētīja un izanalizēja transportlīdzekļa datordiagnostikas iespējas. Pētījumu rezultātā var darba autors var secināt to, ka viens no vienkāršākajiem un ērtākajiem transportlīdzekļa darbības datu ieguves veidiem ir datu ieguve izmantojot OBD-II portu.

Ar šī porta un dažādu skeneru ierīču palīdzību (konkrētajā darba izpildē tika aprakstīts un pielietots ELM 327 datordiagnostikas skeneris) ir iespējams nolasīt tādus transportlīdzekļa darbības parametrus kā dzinēja apgriezienu skaitu minūtē, transportlīdzekļa noskrējienu, degvielas bākas tilpumu un patēriņu, transportlīdzekļa ātrumu, transportlīdzekļa kļūdu kodu sarakstu, un daudzus citus rādījumus.

Iegūstot iepriekš aprakstīto informāciju, darba autors izstrādāja transportlīdzekļu izsekošanas sistēmas konceptu.

Viena no šī koncepta sastāvdaļām ir aplikācija, kas paredzēta darbam „Android” operētājsistēmas vidē un kura spēj nolasīt globālas pozicionēšanas moduļa (programmatūras testiem tika izmantots mobilais tālrunis ar iebūvētu GPS moduli) kā arī datus no OBD-II skanera (kā OBD-II skeneri tika izmantots ELM 327 skeneris ar ”Bluetooth” datu apmaiņas interfeisu, ērtāka savienojuma nodrošināšanai ar ierīci). Iegūstot šos datus, aplikācija nosūta tos uz serveri tālākajai apstrādei.

Darba autors atzīst, ka OBD-II ports tiek atbalstīts sākot no 2000. gada sākuma līdz ar to darbā, izstrādātais transportlīdzekļa izsekošanas sistēmas koncepts nav piemērots transportlīdzekļiem, kuri ir vecāki par 2000. gadu.

Kā otrā transportlīdzekļu izsekošanas sistēmas koncepta sastāvdaļa tika izstrādāta aplikācija, kas darbojas uz servera, saņem, apstrādā un saglabā datubāzē reāla laikā no transportlīdzekļa saņemtus datus.

Papildus tam ar aplikācijas palīdzību ir iespējams apskatīties transportlīdzekļa pašreizējo atrašanās vietu, kā arī kustības maršrutu kādā noteiktā laika posmā. Papildus, lietotājs var apskatīties datus par transportlīdzekļa rādījumiem un transportlīdzekļa darbības laikā sastaptajām problēmām

Izstrādātais transportlīdzekļu izsekošanas sistēmas koncepts tika testēts reālajā laikā, Daugavpils pilsētas apkārtnē izmantojot VW Passat 2001. gadā izlaistu automašīnu. Autora Izvirzītie mērķi saistībā ar šo sistēmu tika sasniegti, un koncepta sastāvdaļu implementācija tika aprakstīta 3.nodaļā.

Autors secina, ka visi šajā darbā izvirzītie mērķi tika sasniegti. Autora Ieteikumu un rekomendācijas koncepta lietošanā un pilnveidošanā ir izklāstīti 4.nodaļā.

# Literatūras saraksts

[1]Vilde A., Rucičš Ā., Viesturs D. Globālās pozicionēšanas tehnoloģijas lauksaimniecībā. – Jelgava: LLU Lauksaimniecības tehnikas zinātniskais institūts, 2008. – 47 lpp.

[2]Zvirgzds J. Globālās pozicionēšanas sistēmas pastāvīgo bāzes staciju izveide Latvijā. RTU zinātnisko rakstu krājums Ģeomātika I, 2005. gads. 51.lpp-57.lpp.

[3]Zvirgzds J. Ģeodzēziskie mērījumi, izmantojot GPS bāzes staciju sistēmu LATPOS, Geodetic Measurements in GPS Base Station Network LATPOS of Latvia. RTU zinātnisko rakstu krājums Ģeomātika, 11. sēj. 2007. gads 81. – 89lpp.

[4]Zvirgzds J. GPS reālā laika mērījumu precizitāte LatPos sistēmā. RTU zinātnisko rakstu krājums Ģeomātika, sērija 11, sējums 3, 2008. gads 29.–36.lpp.

[5]Balodis J., Silabriedis G., Caunīte M., Janpaule I., Balodis K., Rubans A., Aleksejenko I., Otto R., Reiniks M., Mitrofanovs I., Plotņikovs S., Zvirgzds J. EUPOS®-RIGA Network Elevation Analyses. Proceedings of the International Symposium on Global Navigation Satellite. - Rīga, Latvija: Berlin, Germany, 2009. pp. 68.-72

[6]Vilde A., Rucičš Ā., Viesturs D. National Research Council (U.S.). Committee on the Future of the Global Positioning System: GPS sistēmas nākotnes komiteja. – Jelgava: LLU Lauksaimniecības tehnikas zinātniskais institūts, 2008. – 47 lpp.

[7]National Academy of Public Administration (1995). The global positioning system: a shared national asset: recommendations for technical improvements and enhancements. National Academies Press. p. 16. ISBN 0-309-05283-1. Retrieved August 16, 2013., Chapter 1, p. 16

[8]Balodis J., Caunīte M., Janpaule I., Kenyeres A., Rubans A., Silabriedis G., Rosenthal G., Zariņš A., Zvirgzds J., Ābele M. EUPOS and SLR Contribution to GOCE Mission // ESA Living Planet Symposium 2010 Proceedings, Norvēģija, Bergena, 28. jūnijs-2. jūlijs, 2010. lpp. 1.-7

[9]C. R. Anderson Jr and F. G. Lindzey, “Estimating cougar predation rates from GPS location clusters,” J. Wildl. Manage., vol. 67, no. 2, p. 307, 2003.

[10]Козловский Е. [Искусство позиционирования](http://www.vokrugsveta.ru/vs/article/2941/) // [Вокруг света](https://ru.wikipedia.org/wiki/%D0%92%D0%BE%D0%BA%D1%80%D1%83%D0%B3_%D1%81%D0%B2%D0%B5%D1%82%D0%B0_(%D0%B6%D1%83%D1%80%D0%BD%D0%B0%D0%BB)). — М., 2006. — № 12 (2795). — С. 204-280.

[11]Глобальные Навигационные Спутниковые Системы (GNSS). Как это работает?: [http://gps-club.ru/gps\_think/detail.php?ID=20187], 02.09.2012.

[12]M. Hebblewhite and D. T. Haydon, “Distinguishing technology from biology: a critical review of the use of GPS telemetry data in ecology.,” Philos. Trans. R. Soc. Lond. B. Biol. Sci., vol. 365, no. 1550, pp. 2303–2312, 2010.

[13]D. Jennings, S. Cormack, A. J. Coutts, L. J. Boyd, and R. J. Aughey, “Variability of GPS units for measuring distance in team sport movements,” Int. J. Sports Physiol. Perform., vol. 5, no. 4, pp. 565–569, 2010.

[14]Шебшаевич В. С., Дмитриев П. П., Иванцев Н. В. и др. Сетевые спутниковые радионавигационные системы / под ред. В. С. Шебшаевича. — 2-е изд., перераб. и доп. — М.: Радио и связь, 1993. — 408 с. — [ISBN 5-256-00174-4](https://ru.wikipedia.org/wiki/%D0%A1%D0%BB%D1%83%D0%B6%D0%B5%D0%B1%D0%BD%D0%B0%D1%8F:%D0%98%D1%81%D1%82%D0%BE%D1%87%D0%BD%D0%B8%D0%BA%D0%B8_%D0%BA%D0%BD%D0%B8%D0%B3/5256001744).

[15]J. Liu, B. Priyantha, T. Hart, H. S. Ramos, A. a. F. Loureiro, and Q. Wang, “Energy efficient GPS sensing with cloud offloading,” Proc. 10th ACM Conf. Embed. Netw. Sens. Syst. - SenSys ’12, p. 85, 2012.

[16]Левин В.К. Отечественные суперкомпьютеры семейства МВС: [http://parallel.ru/mvs/levin.html], 27.05.2011.

[17]Y. Zheng, Q. Li, Y. Chen, X. Xie, and W.-Y. Ma, “Understanding mobility based on GPS data,” in Proceedings of the 10th international conference on Ubiquitous computing - UbiComp ’08, 2008, no. 49, p. 312.

[18]A. R. Rodgers, R. S. Rempel, and K. F. Abraham, “A GPS-based telemetry system,” Wildl. Soc. Bull., vol. 24, pp. 559–566, 1996.

[19]R. J. Aughey, “Applications of GPS technologies to field sports,” International Journal of Sports Physiology and Performance, vol. 6, no. 3. pp. 295–310, 2011.

[20]M. J. Rycroft, “Understanding GPS. Principles and applications,” Journal of Atmospheric and Solar-Terrestrial Physics, vol. 59, no. 5. pp. 598–599, 1997.

[21]Aлександров И. Космическая радионавигационная система НАВСТАР (рус.) // Зарубежное военное обозрение. — М., 1995. — № 5. — С. 52-63. — [ISSN](https://ru.wikipedia.org/wiki/ISSN) [0134-921X](http://www.sigla.ru/table.jsp?f=8&t=3&v0=0134-921X&f=1003&t=1&v1=&f=4&t=2&v2=&f=21&t=3&v3=&f=1016&t=3&v4=&f=1016&t=3&v5=&bf=4&b=&d=0&ys=&ye=&lng=&ft=&mt=&dt=&vol=&pt=&iss=&ps=&pe=&tr=&tro=&cc=UNION&i=1&v=tagged&s=0&ss=0&st=0&i18n=ru&rlf=&psz=20&bs=20&ce=hJfuypee8JzzufeGmImYYIpZKRJeeOeeWGJIZRrRRrdmtdeee88NJJJJpeeefTJ3peKJJ3UWWPtzzzzzzzzzzzzzzzzzbzzvzzpy5zzjzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzztzzzzzzzbzzzzzzzzzzzzzzzzzzzzzzzzzzzvzzzzzzyeyTjkDnyHzTuueKZePz9decyzzLzzzL*.c8.NzrGJJvufeeeeeJheeyzjeeeeJh*peeeeKJJJJJJJJJJmjHvOJJJJJJJJJfeeeieeeeSJJJJJSJJJ3TeIJJJJ3..E.UEAcyhxD.eeeeeuzzzLJJJJ5.e8JJJheeeeeeeeeeeeyeeK3JJJJJJJJ*s7defeeeeeeeeeeeeeeeeeeeeeeeeeSJJJJJJJJZIJJzzz1..6LJJJJJJtJJZ4....EK*&debug=false).

[22]Y. Yoshihara, T. Suzuki, K. Tominaga, K. Aoki, and H. Nishijima, “Development of an On-Board PM Sensor for the OBD System Based on an Electrochemical Polarization,” SAE Pap., no. 2011–01–2059, pp. 1406–1411, 2011.

[23]N. N. Hasan, A. Arif, U. Pervez, M. Hassam, and S. S. Ul Husnain, “Micro-controller based on-board diagnostic (OBD) system for Non-OBD vehicles,” in Proceedings - 2011 UKSim 13th International Conference on Modelling and Simulation, UKSim 2011, 2011, pp. 540–544.

[24]E. Brothers, “Heavy-duty OBD is coming.,” Fleet Equip., vol. 35, no. 2, pp. 16–19, 2009.

[25]T. Ochs, H. Schittenhelm, A. Genssle, and B. Kamp, “Particulate Matter Sensor for On Board Diagnostics (OBD) of Diesel Particulate Filters (DPF),” SAE Int. J. Fuels Lubr., vol. 3, no. 1, pp. 61–69, 2010.

[26]J. Zaldivar, C. T. Calafate, J. C. Cano, and P. Manzoni, “Providing accident detection in vehicular networks through OBD-II devices and android-based smartphones,” in Proceedings - Conference on Local Computer Networks, LCN, 2011, pp. 813–819.

[27]H.-J. Yun, S.-K. Lee, and O.-C. Kwon, “Vehicle-generated data exchange protocol for Remote OBD inspection and maintenance,” in Computer Sciences and Convergence Information Technology (ICCIT), 2011 6th International Conference on, 2011, pp. 81–84.

[28]S. H. Baek, D. W. Jeong, Y. S. Park, H. S. Kim, M. J. Kim, and J. W. Jang, “Implementation vehicle driving state system with OBD-II, MOST network,” in 17th Asia-Pacific Conference on Communications, APCC 2011, 2011, pp. 709–714.

[29]L. De Oliveira Costa and F. M. Rossin, “Optimizing the On Board Diagnostic System (OBD) to Monitor for Reduction of the SCR Catalyst Conversion Efficiency using the NOx Sensor,” in XIX Congresso e Exposição Internacionais de Tecnologia da Mobilidade, 2010, vol. 7.

[30] J. Hu, F. Yan, J. Tian, P. Wang, and K. Cao, “Developing PC-based automobile diagnostic system based on OBD system,” in Asia-Pacific Power and Energy Engineering Conference, APPEEC, 2010.

[30]M. Finkbeiner, A. Inaba, R. Tan, K. Christiansen, and H.-J. Klüppel, “The New International Standards for Life Cycle Assessment: ISO 14040 and ISO 14044,” Int. J. Life Cycle Assess., vol. 11, no. 2, pp. 80–85, 2006.

[31]T. Jokela, N. Iivari, J. Matero, and M. Karukka, “The Standard of User-Centered Design and the Standard Definition of Usability : Analyzing ISO 13407 against ISO,” Design, vol. 46, pp. 53–60, 2003.

[32]A. Herman, M.-C. Wu, D. Cabush, and M. Shost, “Model Based Control of SCR Dosing and OBD Strategies with Feedback from NH 3 Sensors,” 2009.

[33]D. Le Nguyen, M.-E. Lee, and A. Lensky, “The design and implementation of new Vehicle Black Box using the OBD information,” in Computing and Convergence Technology (ICCCT), 2012 7th International Conference on, 2012, pp. 1281–1284.

[34]P. Siebenbrunner, R. Fischperer, J. Bachler, G. Vitale, and H. Hulser, “OBD of De-NOx-Systems-requirements for software development and calibration for 2010 and beyond,” in SAE Technical Papers, 2008.

[35]D. Wang, K. Huang, W. Liu, Z. Lin, and Y. Wang, “OBD system oxygen sensor degradation monitoring and mechanism analysis,” in Proceedings - 3rd International Conference on Measuring Technology and Mechatronics Automation, ICMTMA 2011, 2011, vol. 2, pp. 740–744.

[36]M. Andersson, P. Ljung, M. Mattsson, M. Löfdahl, and A. Lloyd Spetz, “Investigations on the possibilities of a MISiCFET sensor system for OBD and combustion control utilizing di.erent catalytic gate materials,” Top. Catal., vol. 30–31, pp. 365–368, 2004.

[37] Leaflet as a modern open-source JavaScript library: [http://leafletjs.com/], 02.09.2014.

[38]Leaflet Routing Machine: [http://www.liedman.net/leaflet-routing-machine/tutorials/], 02.09.2014.

# Pielikumi

## 1.Pielikums

**Aplikācijas testa laikā iegūto datu tabula.**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Id** | **Longitude** | **Latitude** | **Time** | **Speed** | **Mileage** | **FuelTank** | **Throtle** | **FuelCon** |
| 1547 | 26.55567 | 55.89113 | 2015-04-27 22:04:12.000 | 6 | 299809 | 46 | 861 | 6.1 |
| 1548 | 26.55554 | 55.89171 | 2015-04-27 22:04:23.000 | 11 | 299809 | 46 | 865 | 6.6 |
| 1549 | 26.55563 | 55.89095 | 2015-04-27 22:04:35.000 | 15 | 299810 | 46 | 862 | 6.6 |
| 1550 | 26.55556 | 55.89059 | 2015-04-27 22:04:43.000 | 12 | 299810 | 46 | 862 | 6.6 |
| 1551 | 26.55561 | 55.89083 | 2015-04-27 22:04:51.000 | 6 | 299810 | 46 | 1123 | 7 |
| 1552 | 26.55584 | 55.891 | 2015-04-27 22:04:59.000 | 31 | 299810 | 46 | 1735 | 8 |
| 1553 | 26.55689 | 55.89155 | 2015-04-27 22:05:10.000 | 49 | 299811 | 46 | 1953 | 8.1 |
| 1554 | 26.55835 | 55.89315 | 2015-04-27 22:05:17.000 | 55 | 299811 | 46 | 1516 | 7.9 |
| 1555 | 26.55928 | 55.89376 | 2015-04-27 22:05:24.000 | 54 | 299811 | 45 | 1517 | 7.9 |
| 1556 | 26.56037 | 55.89451 | 2015-04-27 22:05:31.000 | 53 | 299812 | 45 | 1189 | 7.8 |
| 1557 | 26.56119 | 55.89502 | 2015-04-27 22:05:36.000 | 46 | 299812 | 45 | 860 | 6.1 |
| 1558 | 26.5626 | 55.89643 | 2015-04-27 22:05:43.000 | 21 | 299812 | 45 | 864 | 5.9 |
| 1559 | 26.56314 | 55.89661 | 2015-04-27 22:05:47.000 | 10 | 299813 | 45 | 864 | 6 |
| 1560 | 26.56222 | 55.89604 | 2015-04-27 22:05:54.000 | 11 | 299813 | 45 | 861 | 5.9 |
| 1561 | 26.58447 | 55.90529 | 2015-04-27 22:08:52.000 | 29 | 299813 | 45 | 856 | 5.9 |
| 1562 | 26.5847 | 55.9054 | 2015-04-27 22:08:53.000 | 39 | 299814 | 45 | 2450 | 7.8 |
| 1563 | 26.58575 | 55.89805 | 2015-04-27 22:10:17.000 | 17 | 299814 | 45 | 860 | 6.4 |
| 1564 | 26.58779 | 55.8993 | 2015-04-27 22:10:37.000 | 30 | 299814 | 45 | 855 | 6.4 |
| 1565 | 26.57172 | 55.88567 | 2015-04-27 22:13:13.000 | 46 | 299814 | 45 | 1268 | 7.5 |
| 1566 | 26.56614 | 55.88311 | 2015-04-27 22:13:45.000 | 53 | 299814 | 45 | 1475 | 6.8 |
| 1567 | 26.56056 | 55.88254 | 2015-04-27 22:14:07.000 | 45 | 299815 | 45 | 1275 | 6.5 |
| 1568 | 26.55683 | 55.88064 | 2015-04-27 22:14:23.000 | 55 | 299815 | 45 | 1517 | 6.7 |
| 1569 | 26.5561 | 55.88128 | 2015-04-27 22:14:26.000 | 57 | 299815 | 45 | 1551 | 6.7 |
| 1570 | 26.54801 | 55.88063 | 2015-04-27 22:15:42.000 | 53 | 299815 | 45 | 1501 | 6.6 |
| 1571 | 26.54516 | 55.88186 | 2015-04-27 22:15:56.000 | 49 | 299816 | 44 | 1405 | 6.7 |
| 1572 | 26.54518 | 55.88305 | 2015-04-27 22:16:20.000 | 56 | 299816 | 44 | 1540 | 6.7 |
| 1573 | 26.55226 | 55.88641 | 2015-04-27 22:16:56.000 | 51 | 299816 | 44 | 1451 | 6.8 |
| **Average** | | | | **35.55** |  | **45.18** | **1256.889** | **6.814815** |

## 2.pielikums

**„Android” aplikācijas koda fragmenti**

**Metode POST pieprasījuma nosūtīšanai uz serveri**

public void makePostRequest(String url, String data){

StrictMode.ThreadPolicy policy = new StrictMode.ThreadPolicy.Builder().permitAll().build();

StrictMode.setThreadPolicy(policy);

HttpClient httpClient = new DefaultHttpClient();

HttpPost httpPost = new HttpPost(url);

*//kodējam POST datus*

try {

httpPost.setHeader("Accept", "application/json");

httpPost.setHeader("Content-type", "application/json");

httpPost.setEntity(new StringEntity(data));

} catch (UnsupportedEncodingException e) {

e.printStackTrace();

}

*//veidojam POST pieprasījumu.*

try {

HttpResponse response = httpClient.execute(httpPost);

} catch (ClientProtocolException e) {

e.printStackTrace();

} catch (IOException e) {

e.printStackTrace();

}

}

**Metode Datu sagatavošani sūtīšanai uz serveri**

public void sendDataToServer(Map<String, String> resultMap){

if(mGpsIsStarted && mLastLocation != null && isGPSDataChanged){

dateHelperClass = new DateHelperClass();

*//izveidojam datu moduli*

JSONObject jsonObject = new JSONObject();

try {

jsonObject.accumulate("Longitude", mLastLocation.getLongitude());

jsonObject.accumulate("Latitude", mLastLocation.getLatitude());

jsonObject.accumulate("Time", dateHelperClass.GetDate(mLastLocation.getTime()));

jsonObject.accumulate("Altitude", mLastLocation.getAltitude());

jsonObject.accumulate("Accuracy", mLastLocation.getAccuracy());

jsonObject.accumulate("CarId", carInfo.CarID);

jsonObject.accumulate("Speed", resultMap.get(AvailableCommandNames.SPEED.toString()));

jsonObject.accumulate("Mileage", resultMap.get(AvailableCommandNames.MILIAGE.toString()));

jsonObject.accumulate("FuelTank", resultMap.get(AvailableCommandNames.FUEL\_LEVEL.toString()));

jsonObject.accumulate("IsTurnedOn", "true");

jsonObject.accumulate("Throtle", resultMap.get(AvailableCommandNames.ENGINE\_RPM.toString()));

jsonObject.accumulate("TroubleCodes", resultMap.get(AvailableCommandNames.TROUBLE\_CODES.toString()));

jsonObject.accumulate("FuelConsumption",

resultMap.get(AvailableCommandNames.FUEL\_ECONOMY.toString()));

}catch (Exception e)

{

e.printStackTrace();

}

*//veidojam pieprasījumu*

makePostRequest("http://001-site1.smarterasp.net/Location/RetriveDataFromAndroid", jsonObject.toString());

isGPSDataChanged = false;

}

}

**Metodes OBD servisa startēšanai**

public void startService() throws IOException {

*// atrodam Bluetooth ierīci*

final String remoteDevice = prefs.getString(ConfigActivity.BLUETOOTH\_LIST\_KEY, null);

if (remoteDevice == null || "".equals(remoteDevice)) {

Toast.makeText(ctx, "No Bluetooth device selected", Toast.LENGTH\_LONG).show();

stopService();

throw new IOException();

} else {

final BluetoothAdapter btAdapter = BluetoothAdapter.getDefaultAdapter();

dev = btAdapter.getRemoteDevice(remoteDevice);

// Izveidojam Bluetooth savienojumu

Log.d(TAG, "Stopping Bluetooth discovery.");

btAdapter.cancelDiscovery();

showNotification("Tap to open OBD-Reader", "Starting OBD connection..", R.drawable.ic\_launcher, true, true, false);

try {

startObdConnection();

} catch (Exception e) {

// kļūdas gadījumā apstādinām servisu

stopService();

throw new IOException();

}

}

boolean imperialUnits = prefs.getBoolean(ConfigActivity.IMPERIAL\_UNITS\_KEY,

false);

ArrayList<ObdCommand> cmds = ConfigActivity.getObdCommands(prefs);

}

private void startObdConnection() throws IOException {

Log.d(TAG, "Starting OBD connection..");

isRunning = true;

try {

*// atrodam Bloetooth parrneri un savienojamies ar to*.

sock = dev.createRfcommSocketToServiceRecord(MY\_UUID);

sock.connect();

} catch (Exception e1) {

Log.e(TAG, "There was an error while establishing Bluetooth connection. Falling back..", e1);

Class<?> clazz = sock.getRemoteDevice().getClass();

Class<?>[] paramTypes = new Class<?>[]{Integer.TYPE};

try {

Method m = clazz.getMethod("createRfcommSocket", paramTypes);

Object[] params = new Object[]{Integer.valueOf(1)};

sockFallback = (BluetoothSocket) m.invoke(sock.getRemoteDevice(), params);

sockFallback.connect();

sock = sockFallback;

} catch (Exception e2) {

Log.e(TAG, "Couldn't fallback while establishing Bluetooth connection. Stopping app..", e2);

stopService();

throw new IOException();

}

}

// konfigurējam savienojumu ar skeneri.

queueJob(new ObdCommandJob(new ObdResetCommand()));

queueJob(new ObdCommandJob(new EchoOffObdCommand()));

queueJob(new ObdCommandJob(new EchoOffObdCommand()));

queueJob(new ObdCommandJob(new LineFeedOffObdCommand()));

queueJob(new ObdCommandJob(new TimeoutObdCommand(62)));

*// saņemam protokolu no iestatījumiem*

String protocol = prefs.getString(ConfigActivity.PROTOCOLS\_LIST\_KEY,"AUTO");

queueJob(new ObdCommandJob(new SelectProtocolObdCommand(ObdProtocols.valueOf(protocol))));

*// iegūstam datus*

queueJob(new ObdCommandJob(new AmbientAirTemperatureObdCommand()));

queueCounter = 0L;

## 3.Pielikums

**Web aplikācijas koda piemēri**

**Metode datu saņemšanai no Android ierīces**

[HttpPost]

public ActionResult RetriveDataFromAndroid(LocationViewModel locationVM)

{

var location = Mapper.Map<LocationViewModel, Location>(locationVM);

locationService.Add(location);

List<string> codesArray = String.IsNullOrEmpty(locationVM.TroubleCodes) ? new List<string>() :

locationVM.TroubleCodes.Split(new string[] { "\\n" }, StringSplitOptions.None).ToList();

carService.AddTroubleCodesToCar(carService.GetById(Convert.ToInt32(locationVM.CarId)), codesArray);

return Json(new { succes = true }, JsonRequestBehavior.AllowGet);

}

**Metodes leaflet kartes initializācijai**

var map = L.map('map'){

L.tileLayer('http://{s}.tile.osm.org/{z}/{x}/{y}.png', {

attribution: '© OpenStreetMap contributors'

}).addTo(map);

var routeControl = L.Routing.control({

waypoints: locationArray

}).addTo(map);

$(document).on('click', '#trace-submit', function () {

loadTraceDate();

});

}

**Metode Vidējā degvielas patēriņa ģenerēšanai**

public static Highcharts AverageFuelConsumption(Car car, IQueryable<Location> locations, string chartName)

{

Highcharts chart = new Highcharts(chartName)

.InitChart(new Chart

{

PlotBorderWidth = 0,

PlotShadow = false

})

.SetTitle(new Title { Text = chartName == "chart4" ? "Total Fuel Consumption" : "Fuel Consumption per period" })

.SetPane(new Pane

{

StartAngle = -90,

EndAngle = 90,

Background = new[]

{

new BackgroundObject

{

BackgroundColor = new BackColorOrGradient(new Gradient

{

LinearGradient = new[] { 0, 0, 0, 1 },

Stops = new object[,] { { 0, "#FFF" }, { 1, "#333" } }

}),

BorderWidth = new PercentageOrPixel(0),

OuterRadius = new PercentageOrPixel(109, true)

},

new BackgroundObject

{

BackgroundColor = new BackColorOrGradient(new Gradient

{

LinearGradient = new[] { 0, 0, 0, 1 },

Stops = new object[,] { { 0, "#333" }, { 1, "#FFF" } }

}),

BorderWidth = new PercentageOrPixel(1),

OuterRadius = new PercentageOrPixel(107, true)

},

new BackgroundObject(),

new BackgroundObject

{

BackgroundColor = new BackColorOrGradient(ColorTranslator.FromHtml("#DDD")),

BorderWidth = new PercentageOrPixel(0),

OuterRadius = new PercentageOrPixel(105, true),

InnerRadius = new PercentageOrPixel(103, true)

}

}

})

.SetYAxis(new YAxis

{

Min = 0,

Max = 20,

//MinorTickInterval = "auto",

TickColor = ColorTranslator.FromHtml("#666"),

Labels = new YAxisLabels

{

Step = 2,

//Rotation = "auto"

},

Title = new YAxisTitle { Text = "l/100km" },

PlotBands = new[]

{

new YAxisPlotBands { From = 0, To = 8, Color = ColorTranslator.FromHtml("#55BF3B") },

new YAxisPlotBands { From = 8, To = 12, Color = ColorTranslator.FromHtml("#DDDF0D") },

new YAxisPlotBands { From = 12, To = 20, Color = ColorTranslator.FromHtml("#DF5353") }

}

})

.SetSeries(new Series

{

Name = "Fuel Consumption",

Data = new Data(new object[] { locations.Select(s => s.FuelConsumption).Average() })

});

return chart;

}