TALLINNA TEHNIKAÜLIKOOL
INSENERITEADUSKOND
Tartu kolledž

LAIENDATAVA FUNKTSIONAALSUSEGA BUSSITABLOO PROTOTÜÜP

BUS STOP INFORMATION DISPLAY WITH EXTENDABLE FUNCTIONALITY

RAKENDUSKÕRGHARIDUSTÖÖ

Üliõpilane: Ilja Tihhanovski

Üliõpilaskood: 192997EDTR

Juhendaja: Ago Rootsi, lektor

Tartu 2023

AUTORIDEKLARATSIOON

Olen koostanud lõputöö iseseisvalt.
Lõputöö alusel ei ole varem kutse- või teaduskraadi või inseneridiplomit taotletud.
Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on viidatud.

jaanuar 2023.a

Autor: Ilja Tihhanovski / allkirjastatud digitaalselt /

Töö vastab bakalaureusetöö/magistritööle esitatud nõuetele

jaanuar 2023.a

Juhendaja: Ago Rootsi / allkirjastatud digitaalselt /

Kaitsmisele lubatud

jaanuar 2023.a

Kaitsmiskomisjoni esimees Aime Ruus / allkirjastatud digitaalselt /

Lihtlitsents lõputöö reprodutseerimiseks ja lõputöö üldsusele kättesaadavaks tegemiseks[1]

Mina, Ilja Tihhanovski (sünnikuupäev: 15.10.1977)

1. Annan Tallinna Tehnikaülikoolile tasuta loa (lihtlitsentsi) enda loodud teose „LAIENDATAVA FUNKTSIONAALSUSEGA BUSSITABLOO PROTOTÜÜP“,

mille juhendaja on Ago Rootsi,

1.1 reprodutseerimiseks lõputöö säilitamise ja elektroonse avaldamise eesmärgil, sh Tallinna Tehnikaülikooli raamatukogu digikogusse lisamise eesmärgil kuni autoriõiguse kehtivuse tähtaja lõppemiseni;

1.2 üldsusele kättesaadavaks tegemiseks Tallinna Tehnikaülikooli veebikeskkonna kaudu, sealhulgas Tallinna Tehnikaülikooli raamatukogu digikogu kaudu kuni autoriõiguse kehtivuse tähtaja lõppemiseni.

2. Olen teadlik, et käesoleva lihtlitsentsi punktis 1 nimetatud õigused jäävad alles ka autorile.

3. Kinnitan, et lihtlitsentsi andmisega ei rikuta teiste isikute intellektuaalomandi ega isikuandmete kaitse seadusest ning muudest õigusaktidest tulenevaid õigusi.

---

[1] *Lihtlitsents ei kehti juurdepääsupiirangu kehtivuse ajal, välja arvatud ülikooli õigus lõputööd reprodutseerida üksnes säilitamise eesmärgil.*

/ allkirjastatud digitaalselt /

jaanuar 2023.a

Taltech Tartu kolledž

LÕPUTÖÖ ÜLESANNE

Üliõpilane: Ilja Tihhanovski, 192997EDTR

Õppekava, peaeriala: EDTR17/18 - Telemaatika ja arukad süsteemid

Juhendaja(d): Ago Rootsi, +372 56629821

Lõputöö teema:

(eesti keeles) LAIENDATAVA FUNKTSIONAALSUSEGA BUSSITABLOO PROTOTÜÜP

(inglise keeles) BUS STOP INFORMATION DISPLAY WITH EXTENDABLE FUNCTIONALITY

Lõputöö põhieesmärgid:

Luua andmesidet ja energiavarustust teenusena pakkuvate nutikate bussitabloode võrgustiku toimiv prototüüp. Prototüüp koosneb prototüüpseadmest, näidisandurist, näidis täituri juhtmoodulist ja nende toimimist toetavast tarkvarataristust.

Lõputöö etapid ja ajakava:

Nr	Ülesande kirjeldus	Tähtaeg
1.	Teemakohase kirjanduse läbitöötamine ja eesmärkide, hüpoteeside ning uurimismetoodika täpsustamine	06.2022
2.	Komponentide otsing ja tellimine	06.2022
3.	Esimese prototüübi loomine	07.2022
4.	Iteratiivne prototüübi testimine ja parendamine	10.2022
5.	Tulemuste interpreteerimine	11.2022
6.	Lõputöö käsikirja kokkupanek	12.2022

Töö keel: Eesti keel **Lõputöö esitamise tähtaeg:** 04.01.2023

Üliõpilane: Ilja Tihhanovski / allkirjastatud digitaalselt / jaanuar 2023.a

Juhendaja: Ago Rootsi / allkirjastatud digitaalselt / jaanuar 2023.a

Programmijuht: Aime Ruus / allkirjastatud digitaalselt / jaanuar 2023.a

SISUKORD

EESSÕNA

Antud rakenduskõrgharidusetöö on koostatud Ilja Tihhanovski initsiatiivil tulenevalt autori huvist IoT ja targa linna vastu. Töö teema sõnastati koostöös juhendaga Ago Rootsiga.

Töö eesmärk on luua vahetatavate andurite ja täituritega varustatud nutika bussitabloo ja selle tööd toetava taristu prototüüp. Töö käigus analüüsiti viisi, kuidas saab panna targa linna IoT seadmete võrgustiku lisaks oma põhiülesandele täitma lisaülesandeid säästes niiviisi ressursse ja teenides võrgustiku haldajale tulu. Töö tulemusi soovitakse kasutada edaspidi reaalse seadme väljatöötamiseks ettevõtluse eesmärgil.

Võtmesõnad:
Asjade internet, tark linn, sensorvõrgud, mikrokontrollerid, rakenduskõrgharidusetöö

LÜHENDITE JA TÄHISTE LOETELU

IKT – info- ja kommunikatsiooni tehnoloogia

IoT – värkvõrk või asjade internet (ingl. *Internet of Things*)

LCD – vedelkristallkuvar (ingl. *Liquid Crystal Display*)

LED – valgusdiood (ingl. *Light-Emitting Diode*)

E-Ink – E-tint, elektroonilise paberi tehnoloogia

WiFi – siin traadita andmesidetehnoloogia, IEEE 802.11 sünonüüm

LPWAN – madala energiatarbe ja laia ulatusega traadita sidevõrk (ingl. *Low-Power Wide-Area Network*)

LoRa – patenteeritud LPWAN füüsilise kihi andmeside tehnoloogia (tuletatud ingl. *Long Range*)

LoRaWAN – LoRa võrgu kanalikiht

NFC – lähiväljaside, kontaktivaba lähivälja sidetehnoloogia (ingl. *Near Field Communication*)

GTFS – levinud avaliku transpordivõrgu info jagamise formaat (ingl. *General Transit Feed Specification*)

Tark linn – arenenud linn, mis suudab tänu värkvõrgu tehnoloogiale tagada kestliku majandusliku arengu ja kõrge elatustaseme

ISM – tööstuse, teaduse ja meditsiini (ingl. *industrial, scientific, and medical*) rakenduste jaoks eraldatud litsentsivaba raadiosageduste vahemikud

GSM – 2G mobiilsidet kirjeldav standard (ingl. *Global System for Mobile Communications*). Siin kasutatakse mobiilandmeside tähistamiseks üldiselt

2G – digitaalne mobiilsidevõrkude standard. Praegu kasutusel enamasti vanemate IoT rakenduste poolt.

GPRS – üldine raadio-pakettandmeside teenus (ingl. *General Packet Radio Service*), üks 2G baastehnoloogiatest

UMTS – 2G edasiarendus, tuntakse ka 3G nime all (ingl. *Universal Mobile Telecommunications System*)

LTE – 3G edasiarendus (ingl. *Long Term Evolution*)

5G – mobiilsidevõrkude viies generatsioon

CSV – levinud failiformaat (ingl *comma-separated values*), mida kasutatakse tabelinformatsiooni kodeerimiseks. Kasutatakse GTFS failides.

SoC – integreeritud elektroonikaskeem, kus enamik komponente on koondatud ühele ränikiibile (ingl. *system on a chip*).

HTTP – TCP teeki rakenduskihi protokoll (ingl. *Hypertext Transfer Protocol*). Kirjeldab veebiliiklust.

MQTT – kerge(ingl. *lightweight*) masinatevahelise sõnumivahetuse protokoll, mida saab kasutada piiratud ressursside seadmetes

QoS – MQTT ühenduse tüüp, määrab seda, kuidas on tagatud sõnumi edastamine adressaadini (ingl. *Quality of Service*)

APN – lüüsi aadress (ingl. *Access Point Name*), mille kaudu seade pääseb mobiilsidevõrku

Bluetooth – traadita lühimaa andmeside tehnoloogia

BLE – *Bluetooth Low Energy* – traadita lühimaa andmeside tehnoloogia, kus oluliseks peetakse energiasääst

UART – asünkroonne järjestikliides (ingl. *Universal Asynchronous Receiver-Transmitter*)

API – kirjeldus või viis, kuidas erinevad arvutiprogrammid või nende osad suhtlevad omavahel (ingl. *Application Programming Interface*)

UTC – peamine ajastandard, mida kasutatakse kellade sünkroniseerimiseks (ingl. *Coordinated Universal Time*). Pärineb GMT-st (ingl. *Greenwich Mean Time*)

DST – suveajale ülemineku praktikate kirjeldus (ingl. *Daylight Saving Time*)

ROM – arvuti püsimälu (ingl. *Read-Only Memory*)

SRAM – elektrooniline suvapöördumisega mälu (ingl. *Static Random-Access Memory*)

AES – sümmeetrilise võtmega krüptograafia standard (ingl. *Advanced Encryption Standard*)

SHA-2 – krüptograafia standardite perekond (ingl. *Secure Hash Algorithm*)

RSA – Rivest–Shamir–Adleman – avaliku võtmega krüptograafia süsteem

ECC – avaliku võtmega krüptograafia liik (ingl. *Elliptic-Curve Cryptography*)

I^2C – sünkroonne mitme peaseadme ja mitme alamseadmega andmevahetussiin (ingl. *Inter-Integrated Circuit*)

I^2S – järjestikliides, mida kasutatakse enamasti digitaalse audio edastamiseks

SPI – sünkroonne järjestikliides, kasutatakse lühimaa andmesideks (ingl. *Serial Peripheral Interface*)

cron – ülesannete planeerija unixilaadsetes operatsioonisüsteemides

NTP – seadmete kella sünkroniseerimise protokoll muutuva latentsusajaga pakettandmesidega võrkudes (ingl. *Network Time Protocol*)

SQL – relatsiooniliste andmebaaside haldamise keel (ingl. *Structured Query Language*)

MySQL – levinud avatud lähtekoodiga relatsiooniline andmebaasisüsteem

XML – keel ja formaat suvaliste andmete säilitamiseks ja ülekandmiseks (ingl. *Extensible Markup Language*)

JSON – tänapäeval levinud andmevahetusformaat (ingl. *JavaScript Object Notation*)

Uduarvutus (ingl. *Fog Computing*) – detsentraliseeritud arvutitaristu, kus andmed, arvutused, säilitusruum ja rakendused asuvad andmete allika ja pilve vahel. Selle töö raames uduarvutus tähendab osade arvutuste tegemist bussitabloo moodulites, näiteks

anduri kontroller võib töödelda mõõdiseid selleks, et vähendada edasisaadetavate andmete mahtu. Kontrolleri püsimälu võimaldab andmete säilitamist sideseansside vahel. See võimaldab sideressurssi optimaalsemat kasutamist

Süsteemide süsteem – süsteem, mille komponentideks on omakorda süsteemid. Tavalisest süsteemist oluliselt keerulisem ülesehituse ja käitumise poolest

FreeRTOS – Reaalaja operatsioonisüsteem (RTOS, ingl. *Real Time Operating System*), mida kasutatakse ESP32 mikrokontrolleris

RESTful API – rakendusliides, mis realiseerib REST (ingl. *Representational State Transfer*) arhitektuuri

CRC32 – kontrollsumma arvutamise algoritm

MD5 – räsi (ingl. *hash*) arvutamise algoritm, kasutatakse andmete tervikluse kontrolliks

SISSEJUHATUS

Kõige Internet (ingl. *Internet of Everything*, IoE) on võrk, mis ühendab omavahel inimesi, andmeid, protsesse ja asju [1], [2]. Värkvõrgust (IoT-st) see erineb selle poolest, et IoT ei sisalda inim- ja protsessikomponente [1] esinedes ühena mitmest IoE tehnoloogiatest [2]. IoE lähenemine võib osutuda kasulikuks targa linna kontseptsiooni teostamisel [2]. Üheks mitmest targa linna definitsioonidest võib pidada järgmist: tark linn on asula arenduse visioon, mis käsitleb informatsiooni ja info- ja kommunikatsiooni tehnoloogia (IKT) põimimist linnavarade turvaliseks haldamiseks [2]. Nende linnavarade hulka kuuluvad kohalike omavalitsuste infosüsteemid, koolid, raamatukogud, transporditaristud, haiglad, elektrijaamad, veevärgid ja muud ühiskondlikud teenused [2]. Targa linna mõiste pidevalt areneb ja üks arengusuundadest on erinevate tarka linna moodustavate süsteemide omavaheline ühendamine parema linna juhtimise võimaldamiseks [3]. Tarka linna uuritakse ka TalTechis, kus paikneb targa linna tippkeskus [4].

Ühistransport ja selle seos linnakeskkonnaga on oluline targa linna komponent [5]. Ühistranspordi taristu nutikaks muutmise juures on oluline samm targa bussipaviljoni või nutika bussitabloo kasutuselevõtt [2]. Üha enam peetakse elektroonilist tablood elementaarseks bussipeatuse inventariks [6]. Uuringud kinnitavad, et elektroonilised reaalaja tablood meeldivad reisijatele ja tõstavad usaldust linnatranspordi vastu [7]. Elektrooniliste bussitabloode kasutuselevõtu poolest on Eesti (Tallinn) maailmas esimeste seas [7]. Samas 2020. aasta seisuga olid elektrooniliste tablood vaid u. 9% bussipeatustest Eestis [8].

Laiatarbe mikrokontrollerid muutuvad järjest võimsamateks pakkudes arvestatavat mälumahtu ja arvutusvõimsust [9].

Linnakeskkonna arenduses tuleb vastu võtta otsuseid tuginedes infole. Infot võib saada keskkonda vaadeldes ning selleks saab kasutada erinevaid andureid, millest moodustatakse sensorvõrke. Sensorvõrkude loomise juures tuleb korduvalt lahendada energiavarustuse ja andmevahetuse probleemid [10]. Nutikate bussipeatuste võrgustik, mille loomine igal juhul eeldab nende probleemide lahendamist, võib pakkuda andmevahetust ja energiat teenusena. Lähenedes ülesandele eelkõige kui tehnilisele väljakutsele, autor näeb taolises ressursside jagamises kolm võimaliku kasufaktorit:

1. Ressursside säästmine, olemasoleva energia-, andmevahetus-, ajaressursi parem ärakasutamine, sensorvõrgustike keskkonnajälje vähendamine.

2. Võimalus rahastada nutikate bussitabloode võrgustiku laiendamist ja nii tõsta reisijate heaolu.
3. Tiheda andurite võrgustiku abil saadud infot võib kasutada linna juhtimises.

Plaanitakse asutada ettevõtet, mis pakuks nutikate bussitabloode võrgustiku taristu teenust, seejuures (teatud ettevõtte arengu faasis) nii tarkvaralised kui riistvaralised lahendused, mida taristus kasutatakse, võivad olla avatud. See teeks „sisenemisbarjääri" madalaks. Elektroonilise tabloo olemasolust huvitatud kohalik omavalitsus või lihtsalt elanik saaks tabloo ise kokku panna ja liituda võrguga. See võib aidata kaasa võrgustiku kasvule, tulemusena kasvab ka võrgustiku atraktiivsus sensorvõrkude platvormina. Selline sensorvõrkude loomise juures mitut tekkivat tüüpprobleemi lahendav võrgustik võib osutuda tasuvaks kommertsprojektiks.

Eeltoodust tuleneb selle töö ülesanne. Lõputöö raames luuakse andmesidet ja energiavarustust teenusena pakkuvate nutikate bussitabloode võrgustiku toimiv prototüüp. Prototüüp koosneb prototüüpseadmest, näidisandurist, näidis täituri juhtmoodulist ja nende toimimist toetavast tarkvarataristust.

1. TÖÖÜLESANDE TÄPSUSTUS

Töö keskendub vahetatavate andurite ja täituritega varustatud nutika bussitabloo ja selle jaoks vajaliku taristu väljatöötamise baasküsimustele. Vaadeldakse targa linna olulise osa — nutikate bussitabloode võrgustiku — kasutamist mitte ainult reisijatele vajaliku info edastamiseks vaid ka ümbritseva (linna)keskkonnainfo kogumiseks viisidel, mida ei saa ette arvata taristu loomise hetkel, näiteks erinevate sensorvõrkude loomiseks, eos lahendades osad tüüpilised väljakutsed, mida sensorvõrkude arendajad peavad lahendama, nagu andmevahetus ja energiavarustus.

Töö käigus vastatakse järgmistele küsimustele:

- Kuidas luua nutikate bussitabloode võrgustik kasutades tänapäeva laiatarbes kättesaadavaid komponente ja standardseid tarkvaralisi lahendusi?
- Kuidas optimaalselt lahendada infovahetust bussitabloode võrgustiku erinevate osapoolte vahel?

Töös ei leia käsitlemist järgmised küsimused:

- bussitabloo kasutajakogemus ja ergonoomika,
- prototüübi ilmastiku- ja vandaalikindlus.

Vastused eeltoodud küsimustele on ülimalt olulised reaalse nutika bussitabloo loomisel, kuid nad jäävad välja käesoleva lõputöö mahust.

Tulemusena valmib nutika bussitabloo seadme prototüüp, anduri prototüüp, täituri prototüüp, nende tööd toetava taristu prototüüp.

2. KIRJANDUSE ÜLEVAADE

2020. aasta seisuga olid elektrooniliste tablood vaid u. 9% bussipeatustest Eestis [8]. Suhteliselt vähese leviku üheks põhjuseks peetakse kõrget hinda, mis lisaks tabloo enda ja selle paigalduse hinnale sisaldab ka elektriühenduse väljaehitamise maksumust [8]. Samas on märkimisväärne, et elektroonilised bussitablood olid lisaks Tallinnale ja Tartule ka väiksemates Eesti linnades (Tapa, Valga, Viljandi jt) [8]. Mujal maailmas on uuritud ka nutika bussitaristu loomist maapiirkondade jaoks [11].

Infotabloo oluline aspekt on info ligipääsetavuse tõstmine. Tekst on enamasti esitatud suurema šrifti abil ja helendavad tähed on hästi nähtavad ka pimedas. Tartu bussitablool on ka heliväljund. See kõik on oluline nägemisprobleemidega või ärevushäiretega reisijate jaoks. Väga oluliseks peetakse seda, et reisija saab reaalajas infot busside liiklemise kohta. [8]

2.1. Nutika bussitabloo komponendid

Nutika bussitabloo minimaalsesse koostisosade komplekti kuuluvad infoväljundseadmed, sidepidamist võimaldav aparatuur ja energiaallikas. Info väljastamiseks kasutatakse enamasti ekraani, mõnikord kasutatakse ka kõlarit. Tabloo töötamiseks on vajalik info hankimist võimaldav taristu, mis füüsiliselt paikneb tabloost eemal. Süsteemi võib kuuluda ka mobiilirakendus, mille kaudu võib reisija hankida bussi liiklemist puudutavat infot. Nutikas bussipeatus võib sisaldada ka õhu kvaliteedi monitoorimisseadet, reisija andmesidet võimaldavat varustust (WiFi tugijaam, mobiilsidevõrgu kärje sõlmjaam), USB pistikuid reisijate elektroonika laadimiseks. Selline (näiteks Barcelonas kasutatav) bussipeatus lisaks muule omab kaamerat, loeb reisijaid kokku ja võib anda reisijale teada bussi saabumisest. [2], [12]

Ekraanide puhul kasutatakse LCD tehnoloogiat [2], kuid see võib koosneda LED-idest [13]. Samuti leiavad kasutust ka elektroonilisel paberil põhinevaid ekraanid [14].

Alternatiivseks info väljastamise viisiks on ka heli. Heliväljund tuleb aktiveerida reisijal nupu või kaugjuhtimispuldi abil (pulte võib jagada neid vajavatele reisijatele, nt nägemispuudega inimestele). Heliväljund on võimalik aktiveerida ka sõidukijuhil [8]. Samuti on loodud lahendusi, kus suhtlus bussipeatuse nutika tablooga toimub žestide abil või puuteekraani kaudu. Osade lahenduste puhul on võimalik juhtida tabloo funktsioone mobiilseadmest kasutades NFC tehnoloogiat [2]. Igasugune reisijale

pakutav võimalus füüsiliselt kontakteeruda tablooga teeb süsteemi enda ja selle kasutamise keerulisemaks ja toob mängu väärkasutamise ja vandalismi aspekti [8].

Sidepidamiseks kasutatakse mobiilsidet (GPRS, või uuem) [2], [11] või madala energiatarbega kaugmaa andmesidet (LPWAN), näiteks LoRaWAN või selle modifikatsioone [11], [14], [15]. Igal sidetehnoloogial on omad plussid ja miinused nii tehnilises kui majanduslikus vaates. Mobiilside on levinud tehnoloogia ja see pakub head andmesidekiirust. LPWAN tehnoloogiad erinevad madalama energiatarbe poolest. Paljud LPWAN võrgud kasutavad litsentsivabu sagedusi, nende kasutamise eest ei pea operaatorile maksma [16]. Teatud olukorras võib LPWAN tagada parema ja kindlama andmeside madalama energiatarbe ja odavama hinnaga [11]. Samas energiasääst ja odavus tuleb andmesidekiiruse ja edastatava info hulga arvelt, litsentsivabadus ei anna kindlust, et süsteem on jätkusuutlik pikema aja jooksul. Teatud olukordades võivad need piirangud saada otsustavaks [16].

Energia hankimine ja säästmine on tähtis aspekt, sest see mõjutab elektrooniliste infotabloode levikut eelkõige paigaldamise hinna ja ülalpidamiskulude kaudu [8]. Näiteks Eestis on lahendatud energiavarustuse probleemi elektrivõrguga liitumise abil [8]. Kasutatakse ka päikesepaneele koos akuga [14], aga pakutakse ka käsivändaga elektrigeneraatorit [11]. Viimast mainitud energiahankimise viisi kasutatakse ühtlasi ka reisija olemasolu tuvastamiseks [11]. Teaduskirjanduses on välja toodud ka võimalus kasutada tuulegeneraatoreid [2].

2.2. Erinevad vaated nutikale bussitabloole

Kasutajakogemus

Toimiva nutika bussitabloo oluline omadus on reisijale pakutav kasutamiskogemus. Peale info ligipääsetavust peetakse oluliseks lisateenuseid, nagu reisija nutiseadmete laadimine, täiendava info kuvamine, ajaviitmise võimalused [2]. Ka seadme vastupidavus vandalismile ja võime sulanduda linnakeskkonda on oluline kestva positiivse kasutajakogemuse osa [7].

Keskkond

Kahjulike ainete sisaldus õhus (lämmastikoksiidi, süsihappegaasi, peenosakeste sisaldus) ja müra bussijaamades võib olla kõrgem kui linnas keskmiselt [7].
Juba praeguseks on kavandatud bussijaamu, mis mõõdavad kahjulike ainete kontsentratsiooni õhus, müra, aga ka temperatuuri, õhuniiskust või mööduvate autode arvu [2].

Seosed ja andmevahetus

Nutikas bussitabloo kasutab infot busside planeeritud ja reaalse liikumise kohta [14]. Eestis busside (planeeritud) liiklemise info andmed on avalikud ja neid vahendab transpordiamet GTFS formaadis [17]. Kuna bussitabloo toimimist juhtiva arvuti, tüüpiliselt mikrokontrolleri võimekus on piiratud, korraldatakse osa andmete töötlusest tavaliselt pilves asuva infrastruktuuri abil, näiteks Microsoft Azure pilves asuvas serveris [14].

Bussipeatuse kasutuse info võib olla sisendiks bussiplaani optimeerimiseks st infovahetus bussipeatuse ja teiste infosüsteemi osade vahel toimub mõlemas suunas [11] ja üksikud tablood võivad toetada uduarvutust toimides osaliselt iseseisvalt ja jagades arvutusülesandeid erinevate süsteemiosade vahel nii, et iga osa ja süsteem tervikuna toimiks võimalikult efektiivselt isegi teatud osade ajutise võrgust väljalangemise korral [11].

Autor ei leidnud lahendust, kus ühe praktilise otstarbe täitmiseks loodud targa linna IoT seadmete võrgustik kavandatakse sellisena, et seda saab kasutada (ja peab kasutama) taristuna teiste seadmete toetamiseks ja peab sellist ideed uudseks.

3. MATERJAL JA METOODIKA

Nutikate bussitabloode võrgustiku vaadeldi kui küberfüüsilist süsteemi. Selle toimimist mõjutavad olulisel määral küberneetiline (bussigraafikud, busside reaalse asukoha info) ja füüsiline komponent (bussitabloo seadmed).

Olulist rolli mängivad ka sotsiaalsed aspektid. Süsteemi teenuseid tarbivad reisijad, selle erinevad komponendid kuuluvad erinevatele organisatsioonidele (nt bussigraafikuid haldab riik, bussid kuuluvad eraettevõtetele, bussipeatusi haldab kohalik omavalitsus) jne. Süsteemi kavandamisel tuli arvestada asjaoluga, et osapoolte koostoimeprotokollide hulk ei ole lõplik.

Ülesande lahendamist alustati analüüsiga, mida jätkati kogu ülesande lahendamise jooksul, vajadusel muutes varasemaid disainiotsuseid uute väljakutsete ilmnemisel.

Analüüsi käigus eraldati kaks abstraktsiooni taset, millel opereeriti:

- Nutikate bussitabloode võrgustiku tase, kus üheks agendiks on nutikas bussitabloo
- Üksiku bussitabloo tase, kus agentide näiteks on ekraanimoodul, andurite ja täiturite moodulid jne.

3.1. Ülesande teoreetiline analüüs ja disainivalikute põhjendused

Analüüsis läheneti ülesandele agenditehnika võtteid kasutades.

Süsteemis osalevad agendid

GAIA analüüsi [18] tulemusena kirjeldati bussitabloo võrgustikus osalevaid agente, nende poolt täidetavaid ülesandeid ja suhteid agentide vahel:

- **Bussitabloo**
    - kuvab reisijale bussiaegu vastavalt bussiplaanile ja võimalusel vastavalt tegelikule bussi liiklemise infole
    - vahendab omanikelt tulnud info täituritele (ja anduritele)
    - vahendab anduritelt tulnud mõõtmistulemused omanikele
    - varustab andurid ja täiturid energiaga
    - kogub ja saadab süsteemi haldurile telemeetriaandmeid
- **Bussitabloo külge ühendatud andur**

    - Kogub ja saadab bussitabloole mõõtmistulemusi ja telemeetriaandmeid kokkulepitud kujul
    - Saab bussitabloo käest käske kokkulepitud kujul
- **Bussitabloo külge ühendatud täitur**
    - Saab bussitabloo käest käske kokkulepitud kujul ja täidab neid
    - Kogub ja saadab bussitabloole telemeetria andmeid
- **Reisija**
    - Saab bussitabloo käest informatsiooni (ekraani kaudu)
    - Suhtleb bussitabloo külge ühendatud täituritega ja anduritega
- **Ühistranspordi infosüsteem**
    - Annab ligipääsu bussisõiduplaanidele
- **Anduri või täituri omanik**
    - Annab täituritele (ja anduritele) käske
    - Võtab vastu ja töötleb anduritelt (ja täituritelt) tulnud mõõtmistulemusi ja telemeetria andmeid
- **Buss** (bussifirma kaudu?)
    - Annab bussitabloole infot busside reaalse liiklemise kohta
- **Süsteemi haldur**
    - Kogub bussitabloode poolt tulnud telemeetria andmeid
    - Reageerib tõrgetele ja vajadusel kõrvaldab rikkeid
    - Paigaldab ja eemaldab bussitablood peatustesse
    - Lisab ja eemaldab andureid ja täitureid vastavalt vajadusele

Agendid võivad omavahel suhelda erinevatel viisidel.

Bussitabloo moodulid asuvad üksteisest väga lähedal ning nende seost võib nimetada „väga tihedaks" (bussitabloo varustab moodulid energiaga, mooduli ainuke viis suhelda välismaailmaga on tabloo kaudu). Füüsilisel tasemel seda seost võib lahendada ühendades neid omavahel kaablitega. On võimalik ka juhtmeta ühendus (elektromagnetilise induktsiooni kasutamine energia edastamiseks [19] ja traadita andmeside), see võib olla kasulik ilmastikukindluse tõstmiseks, aga toob kaasa ka erinevaid probleeme alustades energia ülekande efektiivsusest ja lõpetades suuremate turvariskidega.

Reisija osalemist süsteemis selles töös piiritletakse ekraanile kuvatava info passiivse tarbimisega. Töö edasiarendustena võidakse luua andureid ja täitureid, mille kaudu reisija võib süsteemi teiste osalejatega suhelda.

Ühistranspordi infosüsteem vastutab selle eest, et kehtiv busside liiklemise plaan on kättesaadav teatud veebiaadressi kaudu.

Bussifirmad, kellele bussid kuuluvad, võivad omada erinevaid lahendusi, mis võimaldavad bussi reaalse asukoha jälgimist. Nende väljund ei pruugi olla standardiseeritud. Mõistliku lahendusena nähakse süsteemi võimekust kasutada pistikprogramme, mida arendatakse vajaduse tekkimisel uute bussifirmade liitumisel nutikate bussitabloode võrgustikuga.

Süsteemi haldur või administraator võib muuta süsteemi füüsilist konfiguratsiooni. Selleks ta omab füüsilist ligipääsu seadmetele (andurite ja täiturite füüsiline lisamine ja eemaldamine peab olema takistatud kolmandatele isikutele).

Bussitablood ühendatakse „välismaailmaga" (ühistranspordi infosüsteem, moodulite omanikud jne) LPWAN või mobiilse andmeside abil. Mõlema andmeside tehnoloogia (eriti LPWAN) puhul omab kriitilist tähtsust edastatavate andmete hulga vähendamine. LPWAN standardid piiravad edastatavat andmemahtu. Mobiilandmeside puhul aga iga edastatav bait enamasti maksab. Samuti edastatava andmehulkade vähendamise kasuks räägib mikrokontrollerite järjest paranev, aga siiski suhteliselt vähene andmete töötlemise võimekus. Andmeedastuse vähendamine vähendab ka süsteemi keskkonnajälge.

Eeltoodu tingib vajadust optimeerida andmevahetust bussitabloo ja välismaailma vahel. Üheks selleks sobivaks viisiks on veel ühe agendi – serveri – lisamine süsteemi. Server vahendab IoT seadmete suhtlust välismaailmaga ning samal ajal teeb IoT seadmete seost välismaailmaga kindlamaks ja lihtsamaks.

Server kui bussitabloode võrgustiku keskpunkt

Transpordiamet avaldab GTFS formaadis peatuste infot koos sõiduplaanidega [17]. Füüsiliselt on tegemist küllaltki mahukate CSV failidega, mis sisaldavad kogu Eesti bussipeatuste infot normaliseeritud kujul. Andmebaasi saab kätte ühe kokkupakitud failina, selle suurus on ca 24 MB. Lasta igal bussitablool sellist faili alla laadida ja töödelda ei ole otstarbekas. Peatusi on palju ja massiline allalaadimine paneb transpordiameti serveri liigse koormuse alla. Samuti üks konkreetne peatus vajab väga väikese osa andmebaasist (u. 1-2 kB andmeid). On teada, et andmebaasi uuendatakse ainult kord ööpäevas. [20]

Loomulikuks lahenduseks on võtta kasutusele keskne sõlm piisava mälumahu, arvutusvõimsuse- ja kindla internetiühendusega ning kasutada seda selleks, et

- perioodiliselt alla laadida ühistranspordi infosüsteemi avaandmed,
- töödelda neid sobivateks igale konkreetsele bussipeatusele vajalike andmeid sisaldavateks pakkideks
- säilitada töödeldud infot
- vajadusel kiiresti väljastada infot bussipeatusele

Samuti selline keskne sõlm ehk server sobib hästi ka andurite info töötlemiseks, säilitamiseks ja edastamiseks andurite omanikele.

Tänapäeva pilvelahendused võimaldavad selliseid sobiva (ja vajadusel kergesti muudetava) võimsusega virtuaalseid servereid luua kiiresti ja mugavalt.

Server käitub vahekihina:

- bussitabloo ja bussiplaanide registri vahel:
    - kaitstes bussiplaanide serverit massilise andmete allalaadimise eest,
    - kaitstes bussitablood liigse andmete allalaadimise ja töötlemise eest;
- bussitabloo ja bussifirma vahel:
    - säästes bussifirmat vajadusest teada, mis seadmele saata infot, et buss läheneb
    - säästes bussitablood vajadusest „osata" suhelda iga bussifirma bussi asukoha edastamise lahendusega
- bussitabloo ja anduri või täituri omaniku vahel
    - säästes andurite omanike vajadusest vastu võtta mõõtmistulemusi reaalajas,
- bussitabloo ja muu välismaalima vahel
    - piirates interneti sõlmede arvu, millega bussitabloo suhelda võib, vähendades sel viisil süsteemi potentsiaalset ründepinda.

Sellest, kuidas server täidab oma ülesandeid, räägitakse lähemalt alapeatükis 4.1.

Bussitabloo seadme arhitektuur

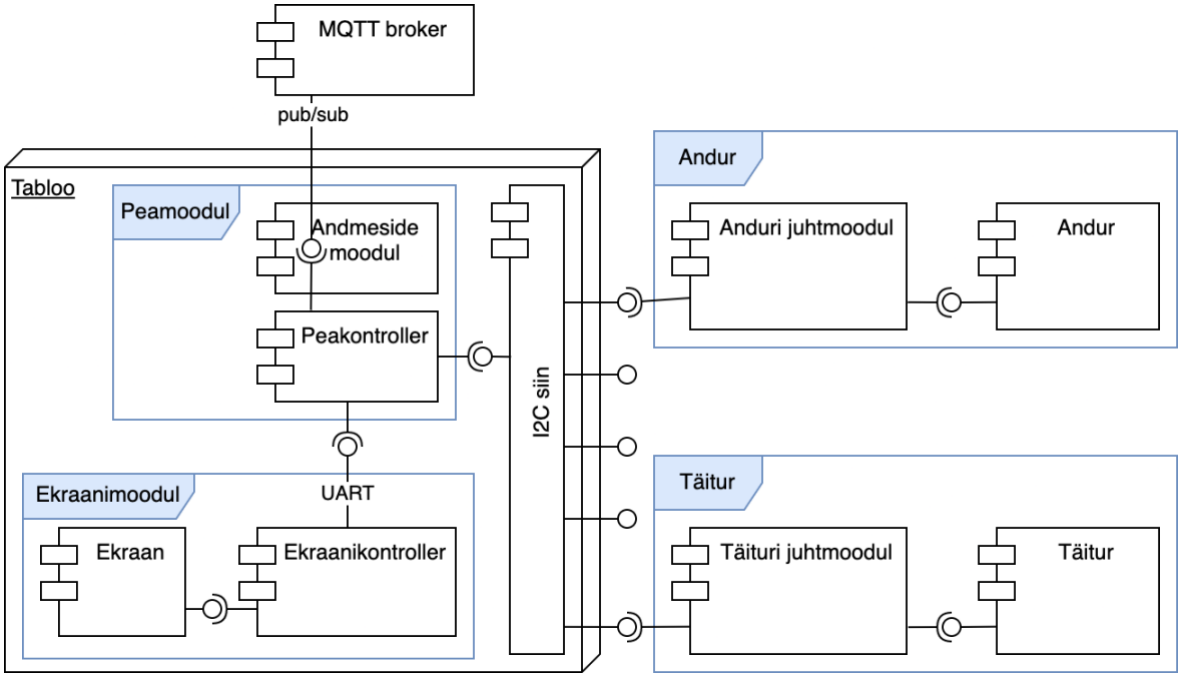
Seadet kavandati koosnevaks moodulitest, seadme näidisskeem on joonisel 3.1:

- peakontroller vastutab andmevahetuse eest ja juhib teiste moodulite tööd;
- ekraanimoodul vastutab bussi info kuvamise eest;

- iga andurimoodul lahendatakse iseseisva agendina, see järgib etteantud andmeside reegleid ning saab iseseisvalt koguda ja eeltöödelda tajurilt tulevat informatsiooni;
- lisaks anduritele võib süsteemi lisada ka täiturite juhtmooduleid, neid võib juhtida peakontrolleri kaudu, vajadusel nad võivad käituda ka autonoomselt.

Iga moodul bussitabloo koosseisus käitub intelligentse agendina ja vastutab oma ülesannete täitmise eest. Moodulid suhtlevad omavahel kasutades eelnevalt defineeritud rollipõhiseid protokolle. Näiteks peakontroller saadab ekraanimoodulile peatuse bussiplaani, jooksva kellaaja infot, aga ei ütle ette, kuidas täpselt seda infot peab kuvama. Osade andurite ja täiturite puhul peakontroller ei pruugi isegi „teada", mis seadmega tegemist on, lihtsalt edastades infot ühel või teisel suunal.

Mooduleid saab vahetada ja nad võivad teatud piirideni toimida iseseisvalt ja teistest moodulitest eraldi. Näiteks ekraanimoodul saab kuvada eeldatavat bussigraafikut ka peakontrolleri rivist väljalangemise korral. Anduri või täituri lisamine või eemaldamine ei pea kuidagi mõjutama ekraanimoodulit või teisi andureid/täitureid.



Joonis 3.1 Tabloo komponendid. Helesinisega on märgistatud antud töö mõistes moodulid. Moodulid käituvad autonoomsete agentidena kapseldades enda sisse oma komponente.

Selline ülesehitus teeb võimalikuks seadme paindliku komplekteerimise vastavalt konkreetse peatuse vajadustele. Näiteks maal asuv peatuse tabloo, mis hangib energiat

lokaalselt päikesepaneelilt või tuulegeneraatorilt, võib olla varustatud e-paberi tehnoloogial põhineva ekraaniga. Oma võimaluste ja omaduste poolest erineb see ekraan oluliselt tüüpilisest linnapildist tuttavast (kaugelt ja pimedas paremini loetavast ent samas energiakulu poolest nõudlikumast) monokroomse LED ekraaniga varustatud tabloost. Seejuures maal asuvat ja linnas olevat tablood võib juhtida täpselt samasugune peakontroller ning mõlemad tablood võivad majutada nii samasuguseid kui ka erinevaid mooduleid.

Ideaalis bussitabloo komponentide vahetamine ei eeldaks süsteemi ümberprogrammeerimist, ümberseadistamist või isegi kõigi osade väljalülitamist. Üks komponent asendatakse teisega, lülitatakse sisse ja ta teeb oma tööd teisi mitte segades. Teisisõnu ka bussitabloo näol on tegemist agentide kooslusega ja bussitabloo võrgustikust võib rääkida kui süsteemide süsteemist.

Sellise iseseisva ja samal ajal sisemistest reeglitest kinni pidava intelligentse käitumise saavutamiseks iga moodul kasutab teatud liiki arvutit. Ühelt poolt see tõstab iga süsteemiosa keerukust ja kogusüsteemi hinda. Teiselt poolt see oluliselt lihtsustab peakontrolleri tööd ja tõstab selle universaalsust tehes kogusüsteemi lihtsamini hoomatavaks ja kergemini hallatavaks. Selle asemel et üritada luua kõikehõlmav kõigi võimalike anduritega suhtlev kontroller, defineeritakse piisavalt universaalne ja üldine peakontrolleri ja mooduli vahelise suhtlemise protokoll. Uue moodulitüübi lisandumisel tuleb selle jaoks realiseerida see protokoll, olemasolevaid süsteemi agente see muudatus reeglina ei puuduta.

Sellisel viisil ehitatud süsteemis peakontroller või server ei pea arvestama konkreetse anduri või täituri iseärasustega, ta suhtleb sellega kui abstraktse agendiga ning selle abstraktsusekihi loob iga anduri või täituri ja ülejäänud süsteemi vahele see vahelüli-arvuti.

„Mooduli mõiste" kui anduri ja täituri mõistete üldistus

Eelnevalt loetletud anduri ja täituri ülesanded paljuski kattuvad ja andurist või täiturist võib bussitabloo kontekstis mõelda kui moodulist, mis täidab mõned või kõik järgmistest ülesannetest:

- Saadab peakontrollerile mõõtmistulemusi
- Saadab peakontrollerile telemeetriaandmeid
- Saab peakontrollerilt käske kokkulepitud kujul (ja täidab neid)

Seejuures on mõeldav olukord, kus ka andur täidab käske (näiteks taaskäivitub või muudab oma seadistusi) ja telemeetriast ning mõõtmistulemustest võib mõelda kui lõppseadme poolt saabuva tagasiside liikidest.

Selle eest, mis toimub mooduli sees, vastutab selle arendaja/omanik. Teised agendid ei pea „teadma" sellest, kas ja kuidas moodul hangib, töötleb ja säilitab mõõtmistulemusi või täidab käske. Tegelikkuses muidugi leidub näiteid, kus selline ideaalne maailmapilt ei pea paika. Agent võib „tahtmatult või tahtlikult reegleid rikkuda" (nt rikkiläinud moodul võib põhjustada lühiseid, omada mitteunikaalset aadressi, moodul võib üritada saata liiga palju andmeid jpm). Ebasoovitud käitumise vältimiseks või peatamiseks vajab peakontroller lisaõigusi, näiteks lisamoodulikohtade toide võib olla väljalülitatav. Süsteemi ebasoovitud käitumise korral peakontroller võib rikkis moodulid välja lülitada. Võib mõelda ka viisidele, kuidas peakontroller saab seda teha autonoomselt või reageerides serveri poolt tulnud käsule. Süsteemihaldur võib omada lihtsat võimalust lülitada moodulid välja füüsiliselt. See võib „päästa rikkiläinud" süsteemi võimaldades selle korrasolevate osade edasist tööd.

Süsteemi abstraktsioonitasemed

Analüüsi ja arenduse lihtsustamiseks jaotati süsteem alljärgnevateks abstraktsioonitasemeteks:

- Võrgustiku tase
- Üksiku tabloo tase
- Mooduli omaniku virtuaalne võrgustik

Kuigi moodul kuulub konkreetsele süsteemis osalejale, ei suhtle ta omanikuga otse, vaid peakontrolleri vahendusel. See vabastab mooduli omaniku andmeside probleemide lahendamisest. Samuti ei pea mooduli omanik lahendama energiavarustuse probleeme. Üldjuhul moodul peab vaid vastama teatud tingimustele oma energiatarbe ja edastatavate andmemahtude osas.

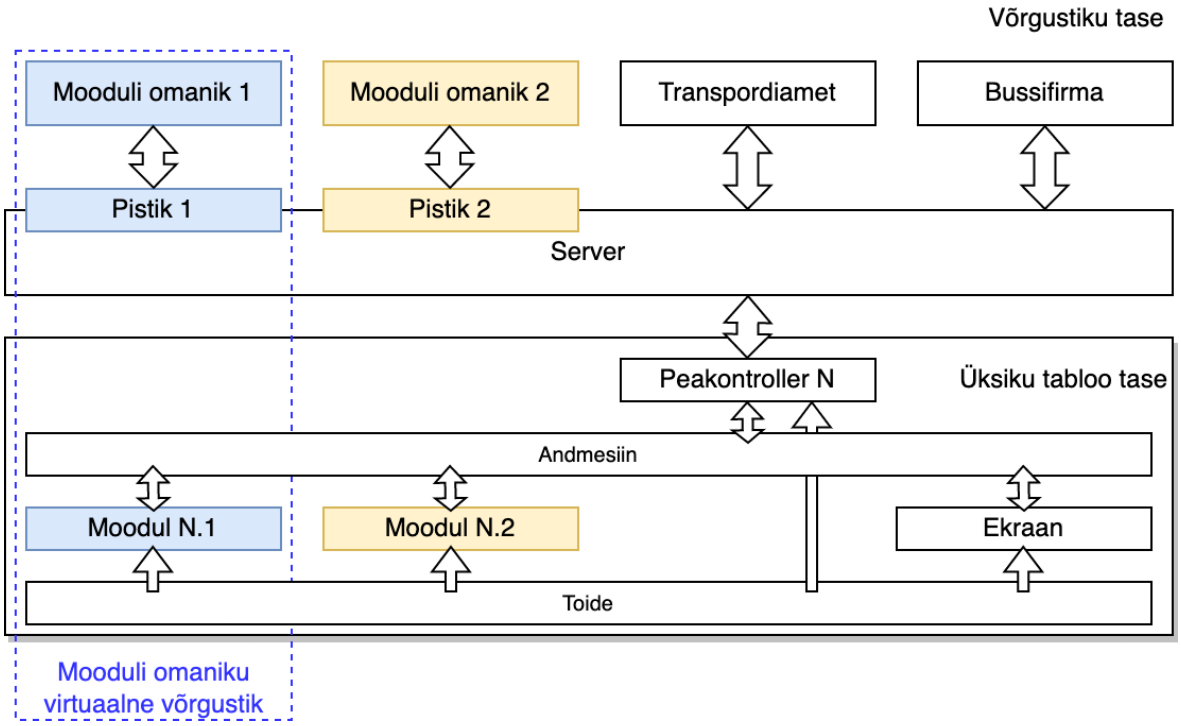
Mooduli omanik ei suhtle ka peakontrolleriga, vaid serveriga. Selleks lepitakse kokku protokollid, mille vahendusel suhtlus toimub. Protokolle võib realiseerida serveri koosseisus toimiva pistikprogrammide abil. Serverit lahendatakse nii, et ta võimaldab pistikprogrammide kasutamist.

Joonis 3.2 näitab, et mõlema abstraktsioonitaseme „sees" toimub mitmekesine andmevahetus sellel tasemel tegutsevate erinevate agentide vahel. Server ja peakontrollerid toimivad lüüsidena abstraktsioonitasemete vahel. Selline lähenemine

võimaldab eraldada võrgustiku loomist kaheks omavahel nõrgalt seotud ülesandeks, mida omakorda võib jaotada alamülesanneteks:

- Võrgustiku tase
    - Osapoolte vahelise suhtlemise protokollid
    - Andmete edastamise ja töötlemise küsimused
- Üksiku bussitabloo tase
    - Seadmete vahelise suhtlemise protokollid
    - Energiavarustuse küsimused

Üksikut bussitablood võib vaadelda ühe agendina võrgustiku tasemel.



Joonis 3.2 Nutikate bussitabloode võrgustiku analüüsi abstraktsioonitasemed

Erinevate bussitabloode küljes olevatest samale omanikule kuuluvatest moodulitest saab mõelda kui virtuaalsest moodulite võrgustikust ehk mooduli omaniku abstraktsioonitasemest.

Andmeside tehnoloogia valik seadme jaoks

Reeglina bussipeatuses puuduvad andmesidekaablid ja nende paigaldamine on kulukas. Tänapäeval on olemas mitmed traadita andmeside tehnoloogiad, millest vaadeldi mobiilandmesidet ja LPWAN näitena LoRaWAN-i. Valikus lähtuti eelkõige prototüübi loomise lihtsusest, taristu ülalpidamise kuludest ja edastatavatest andmemahtudest ning andmeside kiirustest. Ühe või teise tehnoloogia kasutamise võimalusi võib

mõjutada ka seadmete paiknemise geograafia (nt osades maailma piirkondades võib puududa mobiilandmeside, sageduste kasutamise piirangud on riigiti erinevad jms), seega keskenduti Eesti oludele.

LoRaWAN

LoRaWAN on proprietaarsel füüsilisel kihil LoRa põhinev võrgukihi protokoll. See kasutab ISM sagedusi ja võimaldab andmesidekiiruseid 0.3 kuni 50 kbit/s. [21]

LoRaWAN eelistena võib tuua väiksemat energiatarvet, suhteliselt soodsate vahenditega privaatse võrgu loomise võimalust ning üldjuhul võimalust kasutada andmesidet seal, kus pole mobiilsidevõrku. [21]

LoRaWAN puudusteks võrreldes mobiilsidevõrguga on infrastruktuuri ehitamise vajadus, oluliselt väiksemad kiirused ja piirangud edastatava andmemahu suhtes. Sarnased piirangud on ka teistel LPWAN tehnoloogiatel [16]. Samuti võib puuduseks osutuda ISM sageduste kasutamine, kui tehnoloogia leiab laiemat kasutust ja võõrad seadmed hõivavad olulise eetriaja osa ning võivad hakata ettearvamatul kombel segama andmevahetust.

Mobiilne andmeside

Mobiilne andmeside (selle all siin mõeldakse nii üldkasutatavaid GSM, UMTS, LTE, 5G kui ka samal taristul põhinevaid IoT standarte nagu LTE-M) on levinud kogu Eestis ja sõltuvalt kasutatud tehnoloogilisest põlvkonnast pakub kiiruseid 64 kbit/s 2G puhul kuni 10 Gbit/s 5G puhul [22]. Mobiilandmeside on levinud ja küps tehnoloogia, mida laialt kasutatakse ka IoT seadmetes. [23].

Selle eelisteks on lai levik (Eestis) eriti linnades, oluliselt kõrgemad kiirused ja andmeedastusmahud, puuduv andmeside eest vastutava taristu ehitamise ja ülalpidamise vajadus. Tihti võib valida mitme omavahel konkureeriva teenusepakkuja vahel, see tagab teenuse vastuvõetavat hinna ja kvaliteedi suhet.

Puudusteks võib pidada sõltuvust operaatori hinnapoliitikast ja tehnoloogia valikust, on kohti, kus levi puudub. Mainitakse seadmete kõrgemat energiatarvet, kuid energiatarve sõltub olulisel määral konkreetsest seadmest ja see on väga erinev ka erinevatel LPWAN seadmetel.

Mobiilse andmeside valikut nutika bussitabloo prototüübi loomise jaoks tingisid eelkõige selle kasutuselevõtu lihtsus, lihtsam ligipääs internetti ja suurem andmeedastuskiirus, samuti ennustatav ülalpidamise kulu ja side kvaliteet.

3.2. Prototüübi riistvaralised komponendid

Komponentide valikut tingisid lähteülesanne ja autori kogemused. Valmiv bussitabloo prototüüp peab koosnema hinnalt soodsatest komponentidest, see peab võimaldama andmevahetust anduritega ja serveri kaudu kolmandate osapooltega.
Riistvaraliselt minimaalne bussitabloo prototüüp koosneb süsteemi juhtivast mikrokontrollerist või arvutist, andmesidet võimaldavast komponendist, toiteallikast ja ekraanist, kuhu kuvatakse bussiinfo. Lisaks tuleb luua võimaluse ühendada seadmega erinevad andurid ja täiturite juhtmoodulid.

Mikrokontroller

Õppetöö käigus töö autor puutus kokku Arduino seeria, ESP8266 ja ESP32 mikrokontrolleritega, seega valikut tehti neist. Tabel 3.1 võrdleb vaadeldud mikrokontrollerite selle töö kontekstis olulisi omadusi. Sobivaimaks osutus ESP32. Selle SoC omaduste ja hinna suhe tundus olevat parim. Paljude muude mikrokontrollerite hulgas paljulubav tundub RP2040 mikrokontroller, kuid seda ei uuritud lähemalt kogemuse puudumise tõttu.

ESP32 pakub muu hulgas järgmist [24]:

- Ülimalt madalate energiatarbe režiimide kasutamise võimalus
- Juhtmevaba andmevahetuse tugi: WiFi, Bluetooth, BLE
- Erinevate andmeside standardite tugi: SPI, I^2C, I^2S, UART
- Riistvaraline krüptograafia: AES, SHA-2, RSA, ECC
- Kahetuumaline 32-bitine mikroprotsessor taktsagedusega 240 MHz
- 448 kB ROM, 520 kB SRAM, lisaks erinevad muut- ja püsimälu tüübid (Flash, RTC, eFuse)

Asjaolu, et ESP32 toetab Arduino programmeerimisteeki, teeb sellel põhinevate prototüüpide arendust kiireks ja lihtsaks. ESP32 arendusplaatide hind algab mõnest eurost.

Prototüübi arenduses kasutati arendusplaati ESP-WROOM-32 [25] ning sisseehitatud mobiilse andmeside mooduliga SIM800L arendusplaati ESP32-WROVER-B [26].

Kaaluti ka mikroarvuti (nt Raspberry Pi) kasutuselevõttu, aga mikroarvuti hind on reeglina oluliselt kõrgem ja osa selle funktsionaalsusest ei pruugi kasutust leida (HDMI, Ethernet, USB). [27] Samuti mikrokontrolleri kasuks räägib selle kasutamise võimalus ilma arendusplaadita (reaalsetes seadmetes) ja sellega saavutatav kasutatava ruumi, energiatarbe ja hinna vähendamine.

Tabel 3.1 Vaadeldud mikrokontrollerite ja mikroarvuti omadused. Hinnad ei sisalda transpordi maksumust

Omadus	**Arduino Nano [28], [29]**	**ESP32 [24]**	**SparkFun Thing Plus - RP2040 [30]**	**Raspberry PI 4 B [27], [31]–[33]**
Protsessori taktsagedus	20 MHz	2 x 240 MHz	2 x 133 MHz	4 x 1,5 GHz
Püsimälu	ROM 48 kB EEPROM 256 B	ROM 448 kB eFuse 96 B Flash 4 MB või rohkem	Flash 2 - 16 MB	Flash
Muutmälu	SRAM 6 kB	SRAM 520 kB RTC 8 + 8 kB	SRAM 264 kB	SDRAM 1 – 8 GB
UART	1	3	2	6
I2C	1	2	2	6
Voolu tarbimine	7,4 – 24,7 mA	1 µA – 240 mA	11 – 42 mA	540 – 1280 mA
Maksumus, EUR	10,40	2,08 – 17,90	18,48	45,80 – 97,92
Muud omadused		WiFi, Bluetooth, krüptograafia		HDMI, USB, Ethernet

Ekraanimooduli riistvara

Reeglina nutikate bussitabloode koosseisus kasutatakse monokroomseid LED paneele [13], taustvalgustuseta passiivseid vedelkristallekraane, elektroonilist paberit [14], [34] ja ka tavalisi LCD ekraane [35]. Igal tehnoloogial on omad plussid ja miinused. Näiteks elektroonilist paberit või passiivseid LCD ekraane iseloomustab madal energiatarve, aga reeglina on nad väiksemad ja ei ole pimedas nähtavad. Samuti nad jäävad aeglaseks külmas. Monokroomsete LED paneelide eredus (st nähtavus pimedas ja otseses päikesevalguses) on eeskujulik, aga nende energiatarve on suur ja resolutsioon on väike.

Prototüübi jaoks valiti HUB-75 ühendusega RGB LED maatrikspaneelid [36]. Nad pakuvad piisavalt hea ereduse ja kontrastsuse, võimaluse paneele omavahel ühendades panna kokku piisavalt suure ekraani. Osa sellistest paneelidest saab juhtida otse ESP32 kontrollerist, see teeb homogeensemaks tabloo prototüübi riistvaralise arhitektuuri. Prototüübis kasutatud paneeli miinusteks võib lugeda ilmastikukindluse puudumist, suhteliselt suurt energiatarvet ja suhteliselt suuri nõudeid kontrollerile.

Alternatiivse lahendusena katsetati elektroonilise paberi moodulit [37]. Kuna katsetatud moodul on väga väike, seda ei saa pidada muuks kui kontseptsiooni tõestuseks.

3.3. Moodulite näidised

Prototüübi töö illustreerimiseks loodud mõned näidismoodulid. Kuna tabloode võrgustik hakkab pakkuma teiste poolt loodud moodulite majutamise teenust, näidismoodulite loomisel keskenduti eelkõige liidesele mooduli ja peakontrolleri (ja selle kaudu mooduli omaniku) vahel. Näidismoodulite funktsionaalsust jäeti primitiivseks selleks, et pöörata tähelepanu liidesele.

- Õhutemperatuuri ja -niiskuse andur põhineb DHT anduril ja ESP32 mikrokontrolleril
- LED tuld vilgutav täitur - üksik ESP32, mis saades käsku, vilgutab sisseehitatud LED-i.

3.4. Server

Kasutatakse DigitalOcean pilveserverit järgmise tarkvaraga:

- operatsioonisüsteem: Ubuntu Linux [38],
- veebiserver: Apache 2 [39],
- andmebaasiserver: MySQL [40],
- MQTT vahendaja: Mosquitto [41],
- skriptide keel: PHP [42],
- korduvate ülesannete käivitamiseks: cron [43].

3.5. Arenduskeskkond

Seadme tarkvara arendati C++ programmeerimiskeeles ja kasutati selleks ESP32 jaoks loodud Arduino teeki [44]. Kasutati ka muid erinevaid teeke. Serveris käivitatavaid skripte kirjutati PHP-s [40]. Mõlema jaoks kasutati MS Visual Studio Code arenduskeskkonda [45] koos mikrokontrollerite tarkvara arendust toetava PlatformIO

pistikprogrammiga [46]. Versioonihalduseks kasutati Giti [47] ja koodi repositooriumi majutati Githubis [48].

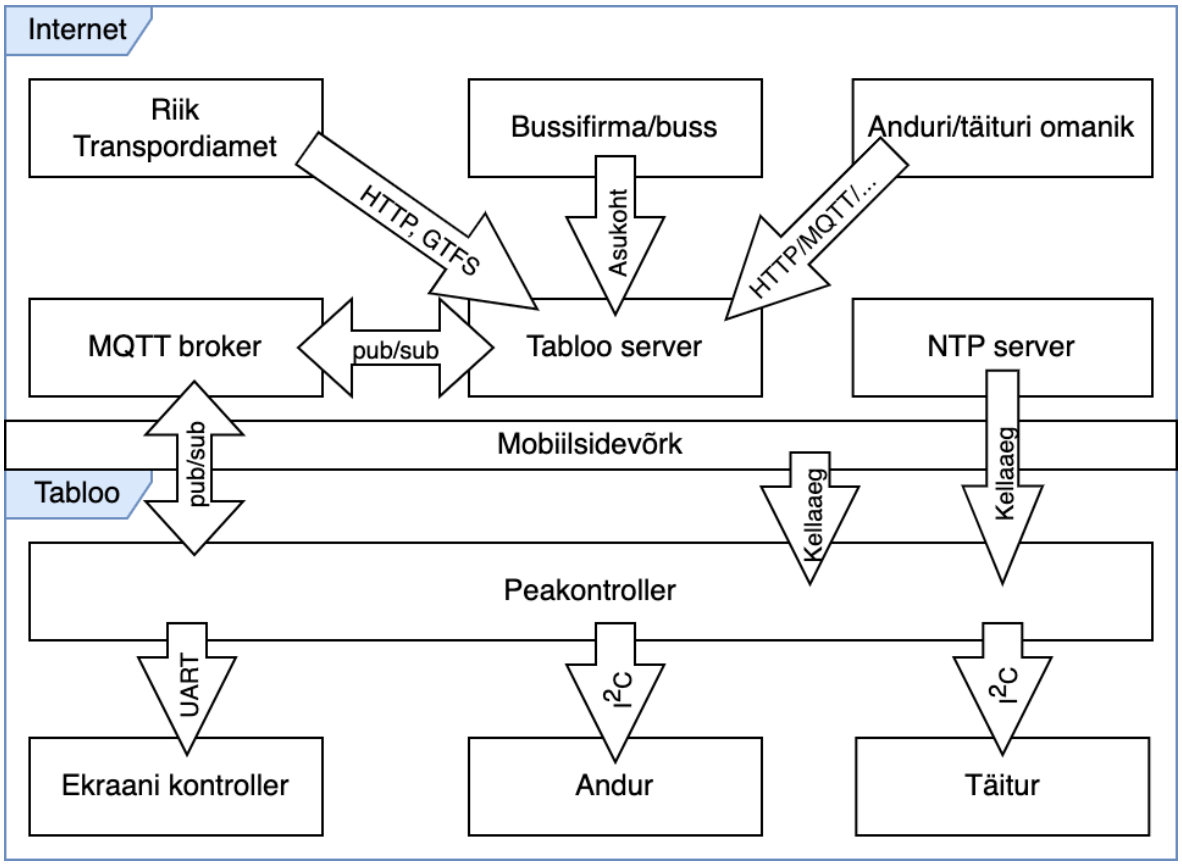
Andmebaasi olemi-suhte diagrammide loomiseks kasutati MySQL Workbench [49]. Muude jooniste ja diagrammide koostamiseks kasutati Diagrams.net veebitarkvara [50]. Arenduses ja katsetes kasutati MQTT klienti MQTT Explorer [51].

4. TULEMUSED

Esmajärjekorras arendati serverilahendus, mis võimaldas info hankimist ühistranspordi andmebaasist, selle töötlemist ja edastamist seadmesse.

Paralleelselt tehti katseid, et leida optimaalne info edastamise viis, katsetati erinevaid teeke ja seadmeid.

Seadme arendust alustati ESP-WROOM-32 arendusplaadiga kasutades andmesideks WiFi-t, seejärel mindi üle ESP32-WROVER-B arendusplaadile 2G mobiilse andmeside võimalusega. 2G võib pidada aegunud tehnoloogiaks, kuid prototüübi arenduse seisukohalt ei olnud vahet 2G ja uuemate mobiilandmeside põlvkondade vahel ja valik tehti arendusplaadi soodsaima hinna alusel.



Joonis 4.1. Nutika bussitabloo üldine andmevoogude skeem

Rakenduskihi andmesideprotokolli valiti HTTP ja MQTT vahel. Seadme andmevahetus serveriga alguses toimis üle HTTP, kuid seejärel mindi üle MQTT kasutamisele, kuna sellel on teatud eelised võrreldes HTTP-ga. IoT rakenduste jaoks algselt arendatud MQTT kasutamine nõuab vähem tarkvaraarendust info usaldusväärse edastamise

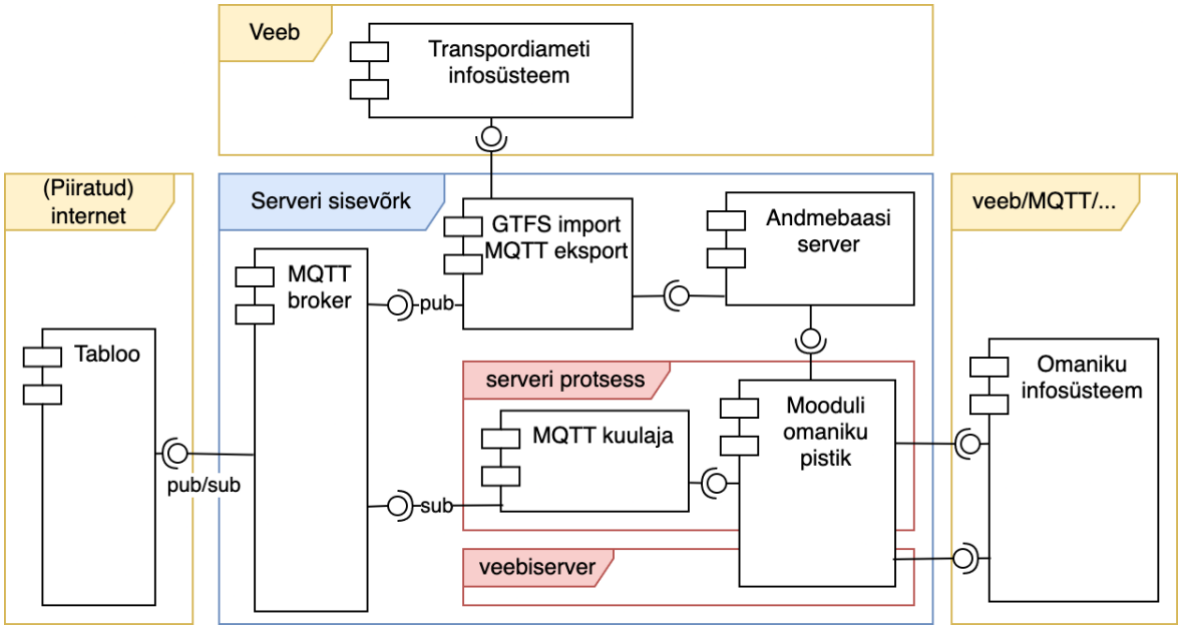
tagamiseks. Oluline on ka asjaolu, et suurema pakettide arvu korral MQTT nõuab vähem andmesidemahtu sama info edastamiseks. See on tähtis IoT seadme piiratud ressursside ja tasulise andmeside puhul. [52], [53]

Taristus kulgevate andmevoogude üldine skeem on toodud joonisel 4.1. Skeemilt on näha, et kogu bussitabloo suhtlus välismaailmaga (välja arvatud kellaaja päringud) toimub MQTT protokolli vahendusel. Jääb lahtiseks, kuidas võivad toimuda kontrollerite tarkvara uuendused, kuid ka neid on võimalik teostada MQTT abil [54].

MQTT asünkroonsus ei võimalda reaalaja andmete edastamist, sellepärast kellaaja infot otsustati pärida GSM võrgult. Katsetamise käigus selgus, et see meetod ei ole alati usaldusväärne [55] ja dubleeriva kellaaja kättesaamise meetodina tuleb kasutada NTP protokolli.

4.1. Server

Server käitub keskse sõlmena andmevahetuses bussitabloo taristu ja ülejäänud maailma vahel. See teisendab erinevate osapoolte andmed sobivatesse formaatidesse, säilitab andmed analüüsiks ja päringuteks sobival kujul. Serveri abil toimub bussitabloode juhtimine, andurite ja täiturite suhtlemine nende omanikega. Serveri komponente kujutab joonis 4.2.

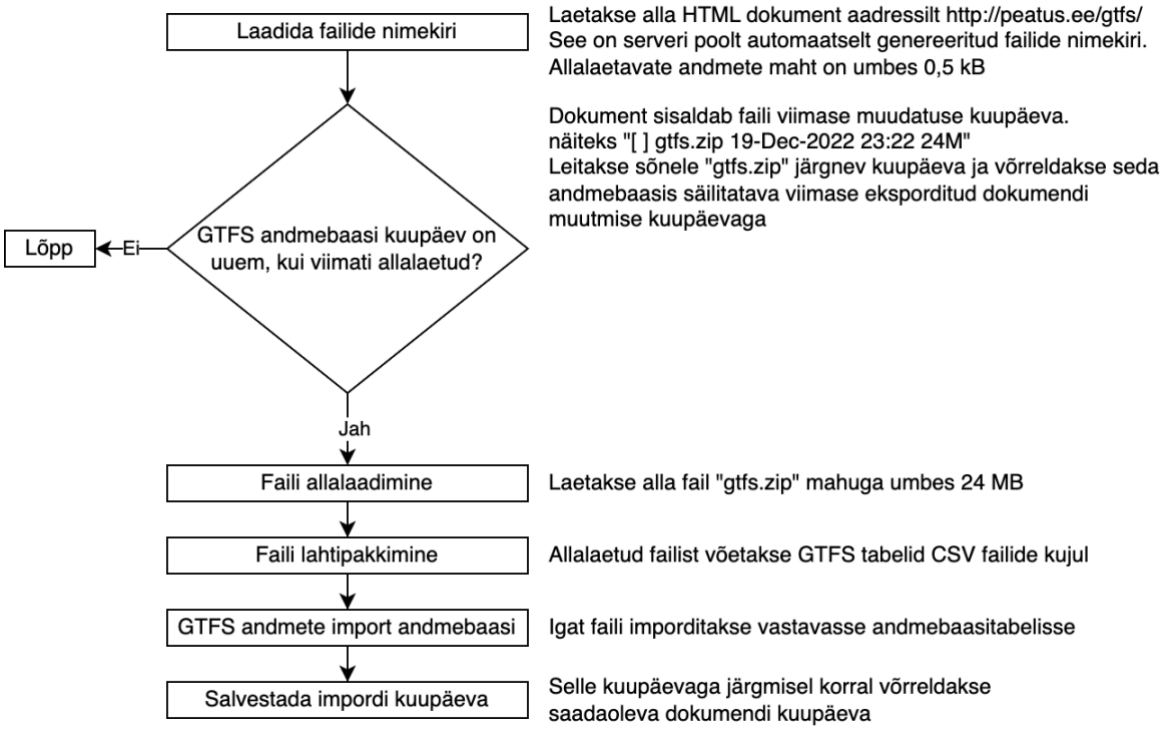


Joonis 4.2 Serveri tähtsamad komponendid ja nende interaktsioonid serveriväliste komponentidega. Komponendid võivad omakorda koosneda komponentidest.

Info hankimine ühistranspordi andmebaasist

Ühistranspordiregistri avaandmed on kirjeldatud transpordiameti poolt spetsifikatsioonis [20] ja kujutavad endast tekstifailide kogumi, kus struktureeritud ning omavahel seostatud kujul toodud andmed ühistranspordi marsruutide, ajatabelite, peatuste jms kohta. Mõistlikuks peeti kogu info importimist andmebaasi automaatselt luues vastava struktuuriga tabeleid selleks, et hiljem saaks kasutada relatsioonilise andmebaasi võimsust vajaliku info hankimiseks SQL päringute abil. Kogu ühistranspordi andmebaasi töötlemine on suhteliselt ajamahukas, MySQL andmebaasi imporditud andmetest ühe peatuse info kokkupanek võtab mõnekümned millisekundid.

Joonis 4.3 kirjeldab ühistranspordi andmebaasi uuenduse algoritmi. Importi käivitatakse *cron*-i abil. Algoritm on koostatud nii, et vältida juba imporditud andmete uuesti allalaadimist.



Joonis 4.3 GTFS andmete uuendamise algoritmi plokkskeem

Impordi skripti kood on kättesaadav tabloo repositooriumist [56] kataloogis /server/importer. Info liikumise skeem on toodud joonisel 4.5.

Andmebaas

Lisaks bussiliinide infole MySQL andmebaas säilitab järgmist infot:

- Lubatud peatuste nimekiri – peatused, mille infot võivad bussitablood näidata

- Moodulite andmed – identifikaatorid (ning tulevikus mooduliga suhtlemise protokolli kirjeldus, näiteks pärimise sagedus jmt), mooduli omaniku info
- Lubatud moodulite andmed peatuste kaupa – igas peatuses võib olla oma andurite komplekt
- Anduritest saadud mõõdiste toorandmed.

Andmebaasi struktuur on toodud lisas 1.

Kasutajaliidesed

Suhtlemine kasutajaliidestega realiseerib (osaliselt) RESTful API. Tabel 4.1 loetleb serveri prototüübi jaoks loodud API lõpp-punkte. Kasutajaliidesed ja API on arenduses ja lõpp-punktide nimekiri võib veel muutuda.

Tabel 4.1 Serveri API lõpp-punktid

API lõpp-punkt	**Selgitus**
Avalik API	Ei nõua sisselogimist
authorities	Omavalitsuste nimekiri
areas/<authority>	Piirkondade nimekiri omavalitsuse kaupa. <authority> väärtust saab võtta nimekirjast, mida tagastab lõpp-punkt „authorities". Väärtus peab olema *urlencoded* vormingus
stops/<area>	Peatuste nimekiri piirkonna kaupa. <area> võimalikud väärtused on toodud „areas" lõpp-punkti väljundis.
Mooduli omaniku API	Autentimine API võtme abil
moduletypes	Moodulite tüüpide nimekiri, lisamine, muutmine, kustutamine
modules	Moodulite nimekiri, lisamine, muutmine, kustutamine
mymodulesoutput	Moodulite poolt tulnud töötlemata väljund. Mooduli omanik näeb ainult oma moodulite väljundit
modulemessage	Käsu saatmine, ainult meetod POST, erinevad atribuudid. Võib saata käske ainult oma moodulitele
Halduri API	Vajalik autentimine
moduleowners	Moodulite omanikud
enabledstops	Tabloodega varustatud peatused
modulemessage	Käsu saatmine, ainult meetod POST, erinevad atribuudid. Haldur võib saata käsu igale moodulile
output	Moodulite poolt tulnud töötlemata väljund. Mooduli omanik näeb ainult oma moodulite väljundit

Bussitabloo serveri veebiaadressil [57] on kättesaadav algeline veebiliides, mille kaudu saab vaadelda peatuste andmeid. Veebiliidese koodiga saab tutvuda repositooriumis kaustas /server/web. API kood asub repositooriumi kaustas /server/api [56].

Infovahetus bussifirmadega

Infovahetuse peaeesmärgiks on bussiliikluse kohta tegeliku info saamine selleks, et täpsustada bussiliini tabloodel kuvatavat järgmise bussi saabumise infot (näiteks, kui buss hilineb). Loomulikuna tundub selle info saamine busside operaatori käest. Selleks võib luua liidese (näiteks veebi API lõpp-punkti, kuhu operaator postitab bussi liiklemise infot).

Võimalik aga ka lahenduse loomine, kus bussid varustatakse RFID siltidega ja bussitablood vastavate lugejatega, mis käituvad loodava süsteemi kontekstis anduritena. Nende poolt tulnud info alusel võib server ise jälgida busside peatustesse saabumist (ja sealt lahkumist).

On olemas ka kesksed targa linna infosüsteemid, mis agregeerivad busside asukoha infot. Näiteks Telia teenuse abil võib jälgida Tartus liikuvate busside asukohti [58]. Teenuse API info alusel võib server ennustada busside hilinemisi ja vajadusel saata vastavad sõnumid bussi marsruudil asuvatele tabloodele.

Infovahetus moodulite omanikega

Infovahetus moodulite omanikega toimub kahes suunas. Omanik võib saata oma seadmele või seadmetele käske. Moodulite poolt tulevad mõõdised ja telemeetriaandmed võivad olla päritud omaniku poolt (näiteks üle veebi) või neid võidakse lükata (ingl. *push*) omaniku infosüsteemi (näiteks MQTT abil).

Ülalkirjeldatud funktsionaalsust realiseeritakse pistik-komponentide abil. Pistik kujutab endast skriptide kogumit, mis vastutavad info edastamise eest.

Prototüübi jaoks realiseeriti pistikud kahe näidismooduli jaoks. Nende kood asub tabloo repositooriumis [56] kaustas /server/plugins.

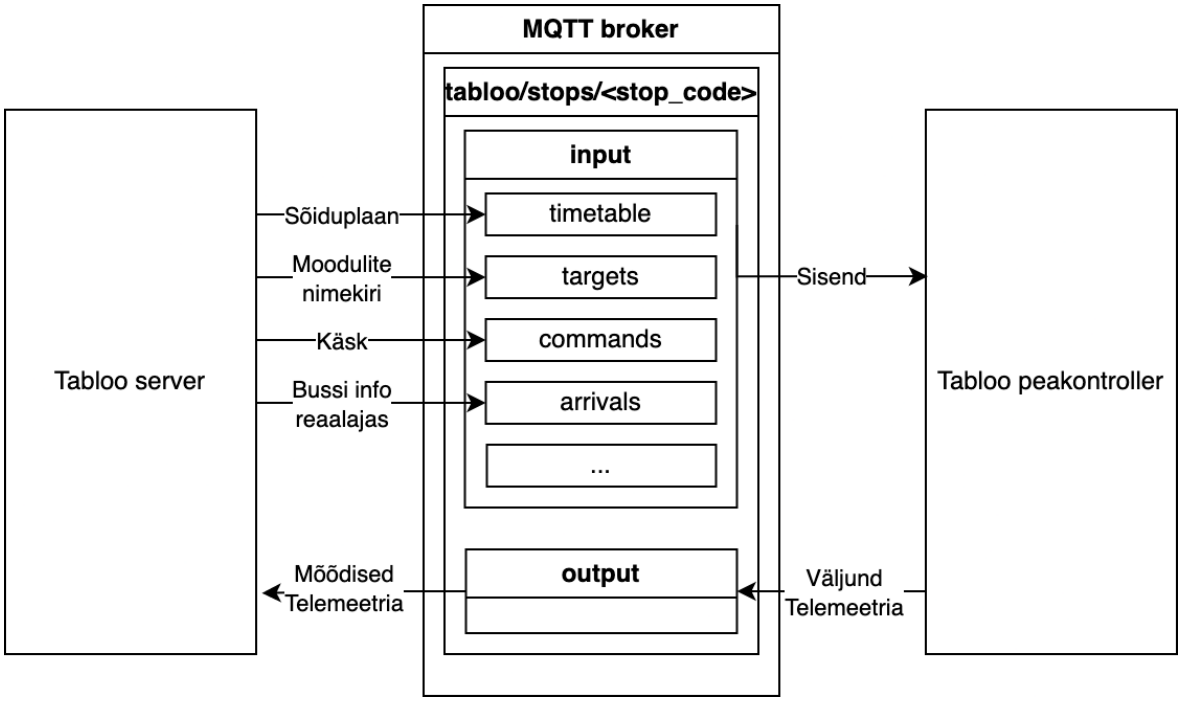
4.2. MQTT vahendaja

Bussitablood suhtlevad serveriga MQTT vahendaja (ingl. *broker*) vahendusel. MQTT jagab vahendatavat infot hierarhilistesse teemadesse (ingl. *topics*). Iga süsteemiga

liidetud bussitabloo jaoks luuakse teema, mille nimi on tuletatud bussitabloo koodist ja mille alamteemad kasutatakse tabloo sisendi ja väljundi jaoks.

Joonis 4.4 näitab MQTT vahendaja teemade struktuuri ja info liikumise suundi vahendaja, serveri ja tabloo peakontrolleri vahel. Iga tabloo jaoks luuakse teema (ingl. *topic*), mille nimi on tuletatud peatuse koodist. Selles on kaks alamteemat:

- input – selle alamteemadesse tulevad sõnumid serverilt. Alamteemad on vajalikud info struktureerimiseks. Tabloo jaoks alamteema nimi ei ole oluline.
- output –server tellib (ingl. *subscribe*) sealt andmeid, mida postitab tabloo.



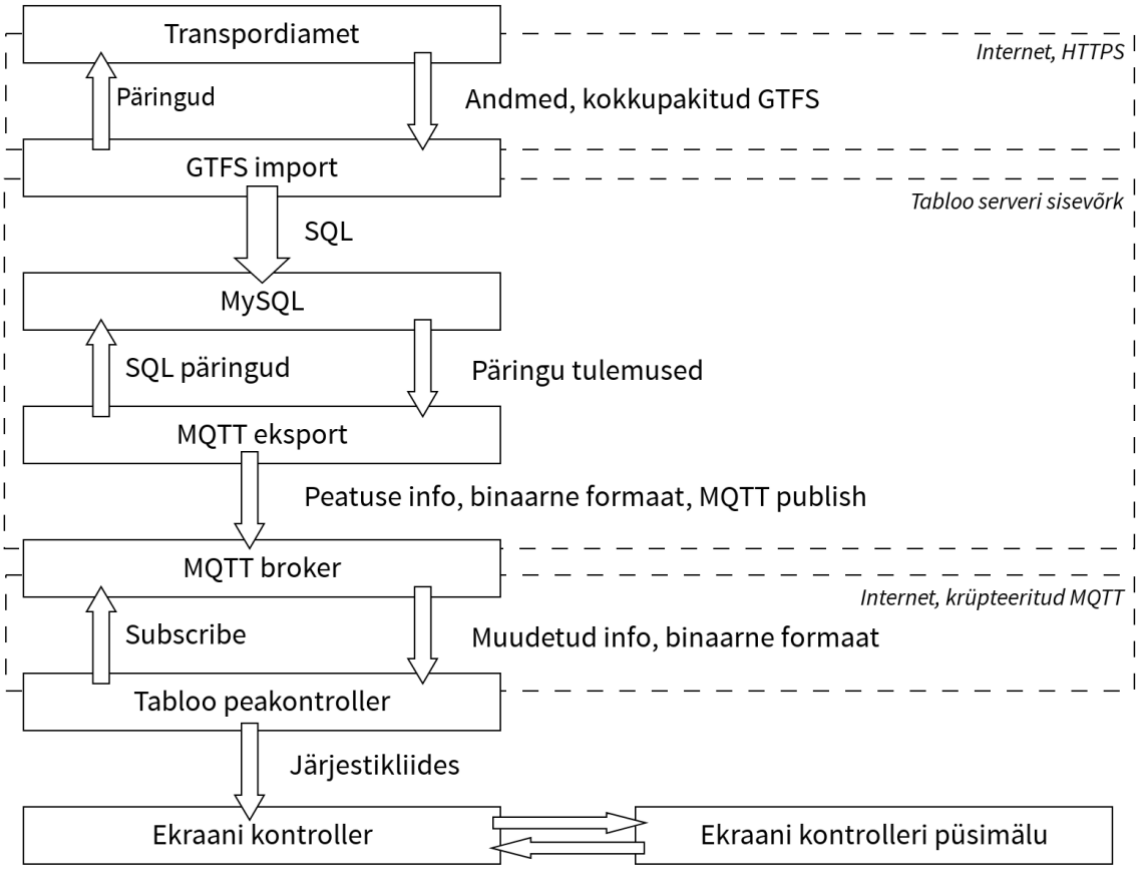
Joonis 4.4 MQTT andmevood.

Prototüüp kasutab vabavaralist MQTT vahendajat Mosquitto, mis on paigaldatud samasse pilveserverisse, kus asub MySQL andmebaas ja andmeid töötlevad skriptid. MQTT vahendaja võib asuda ka mujal, kui selleks tekib vajadus (nt parema andmeedastuse võimekuse vm järele). Miski ei piira ka mõne suure IoT infrastruktuuri teenusepakkuja MQTT vahendaja kasutamist.

4.3. Andmesideprotokollid tabloo tasemel

Andmevahetuseks süsteemi osade vahel kasutatakse erinevaid andmesideprotokolle, osa neist võib defineerida ka tulevikus lisades süsteemi vastavaid pistikprogramme (näiteks andmevahetuseks moodulite omanikuga). Näitena võib tuua joonisel 4.5

kujutatud andmete liikumist ja teisendamist alustades Transpordiameti ühistranspordi andmebaasist ja lõpetades selle info tarbijale ehk reisijale üle andva ekraani kontrolleriga. Võimalusel kasutatakse levinud protokolle ja tehnoloogiaid (HTTP, SQL, MQTT, CSV, JSON).



Joonis 4.5 Andmete teekond transpordiametist nutika bussitabloo ekraani kontrollerini. Meediumid, protokollid, formaadid.

Arenduse käigus tuli valida formaat serveri ja tabloo vahel ning tabloo erinevate komponentide vahel liikuvate andmete struktureerimiseks.

Tabloo poole liigub jooksev kellaaeg, bussisõiduplaan, käsud erinevate tabloo moodulite jaoks. Tagasi saadetakse mõõdiseid ja telemeetria andmeid. Valikut tehti binaarse formaadi ja JSON vahel.

Tänapäeval on levinud andmete ülekandmine JSON formaadis. Sellel on mitmed eelised, see on inimese jaoks kergesti loetav ja arusaadav, kompaktsem võrreldes XML-ga. Selle formaati kasutuselevõtu kaaluti, kuid IoT kontekstis on JSON-il teatud puudused, mis said määravaks:

- binaarne formaat võimaldab sama informatsiooni edasi anda palju väiksema baitide arvu abil ka ilma pakkimisalgoritmide kasutuseta;
- JSON töötlemine nõuab oluliselt rohkem ressursse (kompileeritud koodi maht, arvutused info teisendamiseks)

Binaarse formaadi eelised tulevad hästi välja piiratud ressurssidega (energia, arvutusvõimsus, andmeside) seadmete puhul. Selle kasutamine tundus olevat mõistlikum vaatamata selle põhilisele puudusele (ei ole inimesele loetav, vähem paindlik).

Serverist tulnud sõnumi struktuuri kujutab joonis 4.6. Sõnumil on kahebaidine päis, millele järgneb sõnumi sisu. Sõnumi pikkust tuletatakse MQTT enda paketi andmetest.

Päise esimene bait defineerib sõnumi saaja:

- 0 – sõnum peakontrollerile.
- 1 – sõnum ekraanimoodulile.
- 2 – 255 –lõppseadme I^2C aadress

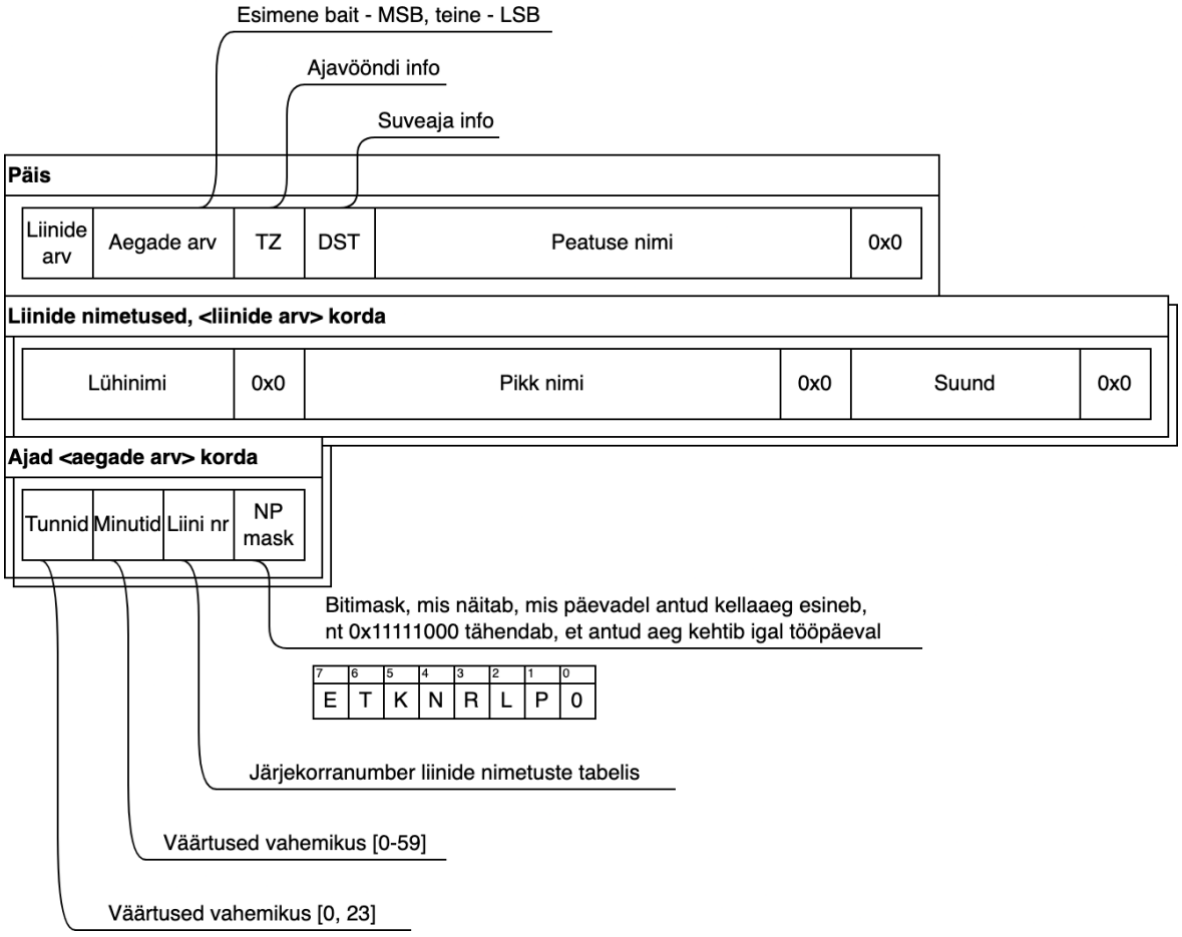
Siht	Tüüp	Sõnumi sisu
1 bait	1 bait	muutuv pikkus

Joonis 4.6 MQTT vahendajalt saabuva sõnumi struktuur

Reaalsuses I^2C kasutab reeglina aadressid 0 kuni 127, seega süsteem võimaldab kuni 126 mooduli ühendamist korraga. See arv on suurem, kui mõistlik maksimaalne seadmete arv, mida saab füüsiliselt ühe bussitabloo külge ühendada.

Päise teine bait tähistab sõnumi tüüpi. Hetkel on defineeritud järgmised tüübid:

- 0 – reserveeritud
- 1 – bussisõiduplaan
- 2 – kellaaeg (MQTT vahendajat seda tüüpi sõnumid ei läbi)
- 3 – käsk
- 4 – moodulite nimekiri



Joonis 4.7 Ühe peatuse bussisõiduplaani formaat

Samu sõnumitüüpe kasutatakse ka tabloo abstraktsioonitasemel peakontrolleri andmevahetuses moodulitega.

Joonis 4.7 illustreerib bussisõiduplaani formaati. Seda realiseerib moodul „BusStopData.h", mis asub tabloo repositooriumis [56] kataloogis /client/modules/lib/BusStopData.

Süsteemis tegutsevad agendid vajavad oma tööks jooksva kellaaja infot. Nutika bussitabloo ekraanil reisija soovib näha, kui palju aega on jäänud järgmise bussi väljumiseni. Mõõdised jõuavad anduri omanikuni prognoosimatu viivitusega ja erinevate mõõtetulemuste omavaheliseks sünkroniseerimiseks neid tuleb esitada aegridadena.

Tabloo peamoodul hangib mobiilsidevõrgust või NTP serverilt jooksvat kuupäeva ja kellaaega ning edastab seda infot teistele moodulitele.

Kuna tablood võivad asuda erinevates ajavööndides, süsteem opereerib UTC aegadega [59]. Kohaliku ajavööndi aja kuvamiseks ekraanimoodulile saadetava bussisõiduplaani päis sisaldab ajavööndi ja suveaja infot.

Kuupäevaga ja kellaajaga manipuleerivaid funktsioone ning formaadi täpne kirjelduse koondab moodul „TablooTime.h" tabloo repositooriumis [56] kataloogis /client/modules/lib/TablooCloud. Kellaaja päringut realiseerib vastava andmevahetuse mooduli (eeltoodud kataloogis failid „TablooGSM.h" või „TablooWiFi.h") meetod „networking_request_datetime".

Selleks, et peamoodulil oleks info muude selle külge ühendatud moodulite kohta, sellele edastatakse moodulite nimekiri. Iga nimekirja bait tähistab ühte moodulit antud tabloo koosseisus. Aadresse 0 ja 1 (peamoodul ja ekraanimoodul) ei edastata, kuna eeldatakse, et nad on alati olemas.

Neljas realiseeritud sõnumi liik on „käsk". Selle sisu ei ole kuidagi defineeritud. Pakett edastatakse saajani ning saaja interpreteerib ja vajadusel täidab selles sisalduvat käsku.

Käsu näidis on sõne „reboot", mis paneb seda saanud moodulit taaskäivituma.

4.4. Seade

Tabloo moodulid on realiseeritud ESP32 mikrokontrollerite baasil. Seadmete elektrilised skeemid on toodud lisas 2.

Moodulite lähtekood C++ keeles on kättesaadav nutika bussitabloo repositooriumist [56] kataloogis /client/modules. Iga mooduli kood asub eraldi kataloogis ja kujutab endast PlatformIO [46] projekti. Osa koodist on korduvkasutatud mitme mooduli poolt. Sellised koodi osad asuvad kataloogi /client/modules/lib alamkataloogides ja edaspidi neid nimetatakse tabloo raamistikuks. Tabel 4.2 loetleb realiseeritud mooduleid. Arenduses kasutati ESP32 Arduino [44] ja FreeRTOS [60] raamistike ja erinevaid andmeside- või graafikateeke.

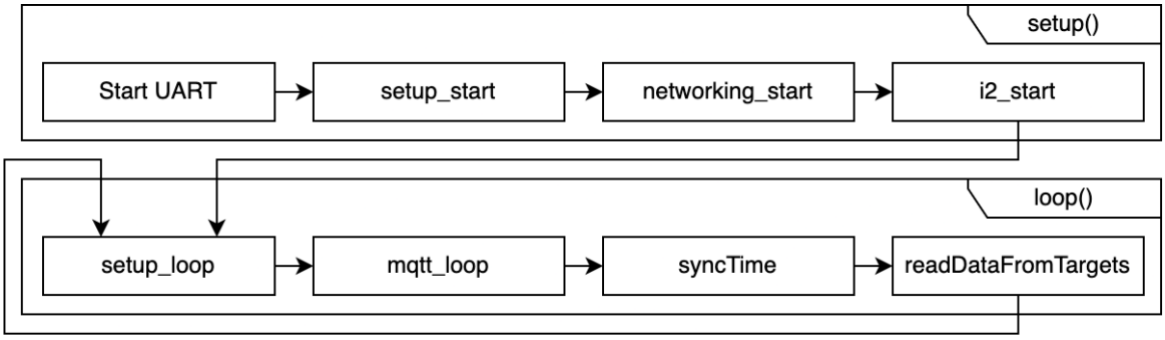
Tabel 4.2 Realiseeritud avatud bussitabloo moodulite nimekiri. Asukohad antud kataloogi /client/modules suhtes

Moodul	**Asukoht repositooriumis**
Peakontroller mobiilandmesidega	main_module_sim800l_mqtt
Peakontroller WiFi andmesidega	main_module_wifi_mqtt
LED maatriksdisplei kontroller	display_controller_hub75
E-paberi ekraaniga kontroller	display_eink_29
Näidisandur	sensor_dht11
Näidistäitur	actuator_blink

Peamoodul

Peamoodul vastutab andmevahetuse ja teiste moodulite juhtimise eest. See on realiseeritud mobiilse andmeside mooduliga SIM800L [61] arendusplaadil ESP32-WROVER-B [26]. Peamoodul suhtleb teiste moodulitega I^2C ja järjestikpordi kaudu. Peamoodul korraldab moodulite andmevahetuse serveriga ja serveri kaudu kõigi teiste süsteemi agentidega. Selleks kasutatakse mobiilsidet ja rakenduskihi protokollina MQTT-t. On olemas arenduse käigus kasutatud peakontrolleri modifikatsioon, mis põhineb ESP-WROOM-32 [25] mikrokontrolleril ja kasutab füüsilise võrgukihina WiFi-t.

Järgnevalt selgitatakse peakontrolleri töös kasutatud algoritme mobiilandmesidega mooduli näitel.



Joonis 4.8 Peamooduli tööalgoritmi plokkskeem.

Peamooduli programm on realiseeritud C++ keeles Arduino raamistikus [44] ja seda illustreerib joonis 4.8. Mikrokontrolleri käivitamise järel (meetodis setup) seadistatakse järjestikliides ning initsialiseeritakse seadistuste, võrgu ja I^2C moodulid.
Mikrokontrolleri tsüklis (meetod loop) toimub korduvate ülesannete käivitamine. Meetodid setup_loop ja mqtt_loop on vastavalt seadistuste mooduli ja MQTT moodulite meetodid, mis vastutavad seadistuste ja MQTT suhtlemise eest, syncTime iga

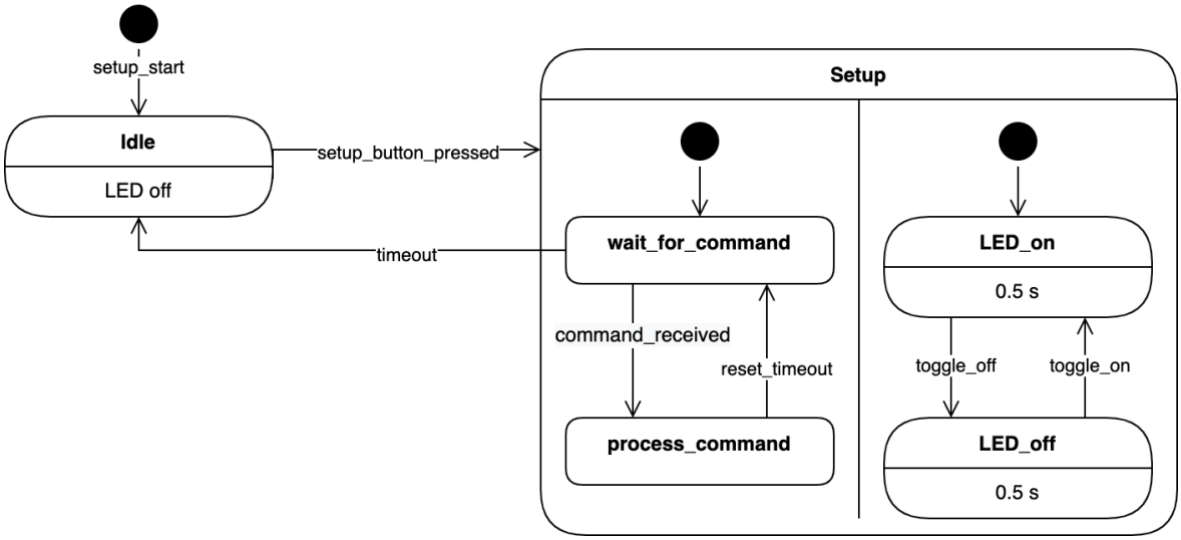
TIMER_SYNC_INTERVAL millisekundi järel alustab jooksva kellaaja pärimist. Meetod readDataFromTargets iga TARGETS_POLL_INTERVAL millisekundi möödumisel kontrollib, kas I^2C teenuse poolt on tulnud andmeid alamseadmetest ja kui midagi on saabunud, postitab neid MQTT vahendajasse.

Seadistused

Peakontroller salvestab püsimällu peatuse identifikaatori ja võrguühenduse loomiseks vajalikud seadistused:

- Mobiilsidevõrgu ligipääsu parameetrid
  - SIM kaardi PIN
  - APN
  - Kasutajanimi
  - Salasõna
- Peatuse kood transpordiameti andmebaasist.

Seadistuste funktsionaalsust realiseerib tabloo raamistiku teek „TablooSetup.h", mis kasutab andmete salvestamiseks Arduino teeki „Preferences.h". Kasutajaliideseks kasutatakse arendusplaadi järjestikporti (UART0 või Serial), seda realiseerib tabloo raamistiku teek SerialInput.h ja BLE andmesidet, mida realiseerib tabloo raamistiku teek „TablooBLESetup.h" NimBLE [62] teegi abil.



Joonis 4.9 Seadistuste teegi olekudiagramm.

Selleks, et kaitsta peakontrollerit autoriseerimata ümberseadistamisest, seadet tuleb viia seadistuste muutmise olekusse vajutades riistvaralist nuppu, mida saab reaalse seadme puhul peita korpusesse ja kaitsta näiteks lukuga. Seadistamise olekut märgistab vilkuv tuluke. Seadistuste teeki tööd illustreerib olekudiagramm joonisel 4.9.

Internetiühendus

Andmevahetust realiseeritakse TinyGSM teegi [63] abil. Tabloo raamistiku teek „TablooGSM.h" põhineb TinyGSM näidistel. Mikrokontrolleri käivitamise järel käivitatakse modem ja seade ühendatakse andmesidevõrguga. MQTT toimimiseks vajalik pidev internetiühendus, selle tagamiseks aeg-ajalt kontrollitakse ühenduse olemasolu ja vajadusel ühendatakse uuesti.

Andmevahetus serveriga

Peakontroller suhtleb serveriga krüpteeritud MQTT protokolli vahendusel, mille eest vastutab tabloo raamistiku teek „TablooMQTT.h". Kasutatakse teegi PubSubClient [64].

MQTT teemad (ingl. *topics*) on järgmised:

- `Tabloo/stops/<peatuse_kood>/input/#` – sisendteema, MQTT puhul sümbol „#" tähendab seda, et klient saab kõigi selle alamteemade sõnumeid. Joonis 4.4 loetleb sisendteema alamteemad.
- `tabloo/stops/<peatuse_kood>/output` – väljundteema. Kõiki sõnumeid peakontroller postitab siia.

Tabloo poolt postitatavate sõnumite QoS on 0, see tähendab, et sõnumit edastatakse maksimaalselt ühe korra. See on kasutatud teegi piirang. Mõistlikum oleks kasutada QoS 2, mis tagaks täpselt ühekordse sõnumi edastamise.

Andmevahetus moodulite vahel

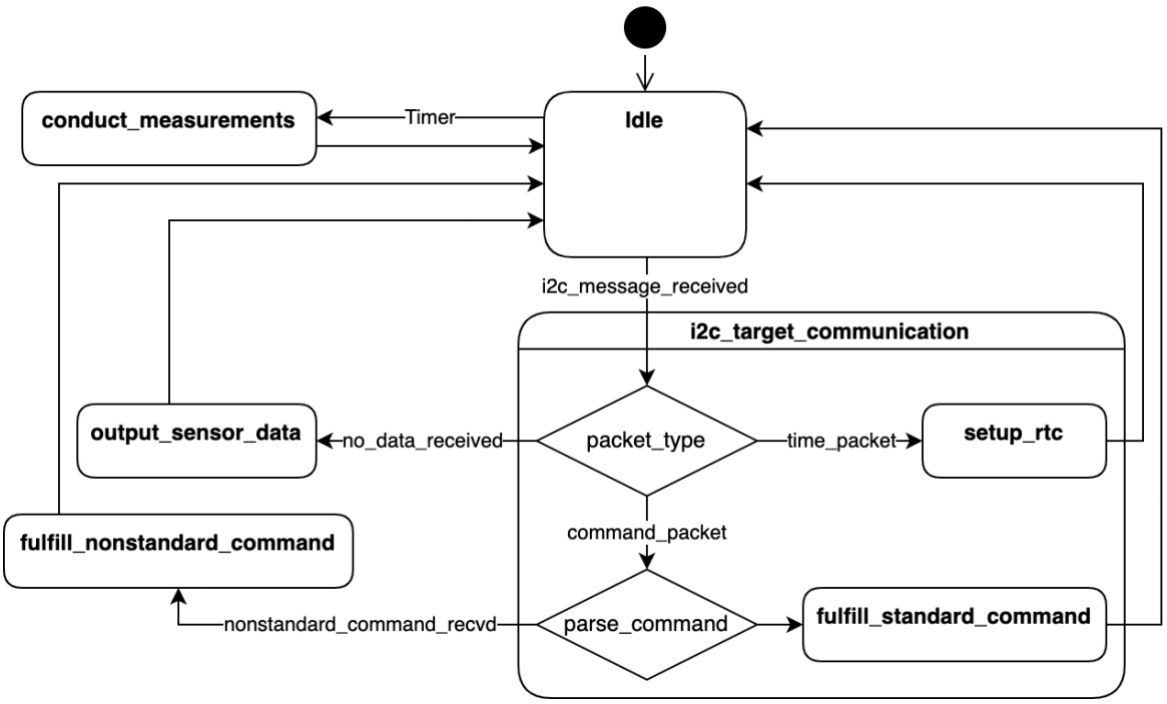
Mooduleid võib ühendada peamooduliga I^2C või UART liidese abil. Järjestikliidest UART kasutatakse erandina ekraanimooduli realisatsiooni riistvaraliste piirangute tõttu.

Moodulid peavad kasutama sama andmeedastuskiirust. UART puhul prototüüp kasutab kiirust 9600 bit/s, ESP32 I^2C siini vaikimisi kiirus on 100 kbit/s [24, lk 36].

Tarkvaraliselt andmevahetust realiseerivad tabloo raamistiku teegid „UARTTransport.h", „UARTIO.h" ja „TablooI2C.h".

Moodulid

Moodulid on realiseeritud I^2C alamseadmetena. Nad saavad sõnumeid peakontrollerilt, mis käitub I^2C ülemseadmena ning täidavad neid vastavalt sõnumi sisule. Anduri või täituri toimimist kirjeldab olekudiagramm joonisel 4.10.



Joonis 4.10 Anduri või täituri olekudiagramm.

Bussitabloo abstraktsioonitasemel moodulit identifitseerib number vahemikust [2, 127] („siht" Joonis 4.6), mis ühtib tema I^2C aadressiga. Aadresse 0 ja 1 renditavate moodulite jaoks ei kasutata. Ülejäänud aadresside kasutamine ei ole piiratud.

Kõigil sama tüübi moodulitel peab olema sama aadress sõltumata tabloost, mille koosseisu ta kuulub (samadest moodulitest erinevate tabloode külge ühendatud „virtuaalne võrgustik" joonisel 3.2 peab koosnema sama I^2C aadressiga moodulitest). See võib piirata süsteemis osalevate erinevate moodulitüüpide arvu ja välistab mitme sama tüüpi moodulite ühendamist sama peakontrolleriga, kuid sellest piirangust saab üle minna, võimaldades I^2C aadresside dünaamilist määramist seadmetele ning identifitseerides moodulit peatuse koodi ja I^2C aadressi paariga.

I^2C andmevahetuse lihtsustamiseks on loodud teek TablooI2CTarget.h [56], mis realiseerib andmevahetust peakontrolleriga ja vastutab standardsete tegevuste eest (nt seadistab seadme reaalaja kella, taaskäivitab kontrollerit jms)

Bussitabloo ei kirjuta ette seda, kuidas moodul hangib andurilt mõõtmistulemusi, säilitab neid andmesideseansside vahel, kuidas täitur reageerib käskudele jms.

Näidisanduri ja -täituri kood (vt tabel 4.2) on olemas repositooriumis [56].

Ekraanimoodul

Ekraanimoodul vastutab peatuse info kuvamise eest tehes seda autonoomselt ja vastavalt riistvara võimalustele ja piirangutele. Seda võib käsitleda täiturina bussitabloo abstraktsioonitasemel.

Riistvaraliste piirangute tõttu (ESP32 I^2S kasutamine RGB LED maatriksekraani juhtimiseks segab I^2C tööd [65]) andmevahetust tuli realiseerida UART järjestikliidese kaudu. Ekraanimoodul suhtleb ainult peakontrolleriga ning saab peakontrollerilt järgmisi sõnumeid:

- Jooksev kuupäev ja kellaaeg
- Peatuse info siis, kui see muutub
- Käsud eriinfo kuvamiseks

Kuvatav info

- Peatuse nimi
- Peatuvate liinide nimekiri
- Muud sõnumid

Järjestikpordi kaudu andmevahetust realiseerivad tabloo raamistiku teegid „UARTTransport.h" ja „UARTIO.h". Andmesidepakett sisaldab sõnumi tüüpi, pikkust ja kontrollsummat, mida arvutatakse CRC32 algoritmiga. Sõnumi kättesaamist kinnitab spetsiaalne pakett.

Saadud bussisõiduplaani kontroller salvestab püsimälusse, see võimaldab ekraanil toimida teatud ulatuses autonoomselt. Näiteks peakontrolleri rivist väljalangemise korral võib ekraanimoodul endiselt näidata antud peatust läbivate bussiliinide nimekirja. Kui varustada ekraanimoodul patareitoitel reaalajakellaga, võib selle autonoomsust veelgi tõsta – nii saab ekraanimoodul kuvada eeldatavaid busside väljumise aegu ka ilma internetiühenduseta.

Flash mälu korduva asjatu ülekirjutamise vältimiseks võib peamoodulist saabunud infot salvestada vaid siis, kui see erineb juba salvestatud infost. Võrdlemise lihtsustamiseks võib kasutada andmete räsimist (näiteks algoritmi MD5).

Arenduse käigus loodi kolm ekraanimoodulit: ühest RGB LED paneelist (64x32 pikslit), neljast RGB LED paneelist (kokku 128x64 pikslit) ja elektroonilise paberi ekraaniga. Ekraanimoodulite koodi asukohad repositooriumis [56] on toodud tabelis 4.2.

RGB LED ekraanipaneelide juhtimiseks kasutatakse teegi „ESP32-HUB75-MatrixPanel-I2S-DMA" [66]. Elektroonilisel paberil põhinev ekraan juhitakse teegi „GxEPD" abil [67].

4.5. Prototüübi katsetamine

Erinevaid taristu ja prototüübi funktsioone testiti arendamise ajal eraldi. Katsetati erinevate funktsioonide ja algoritmide tööd nii arvuti keskkonnal kui mikrokontrolleril.

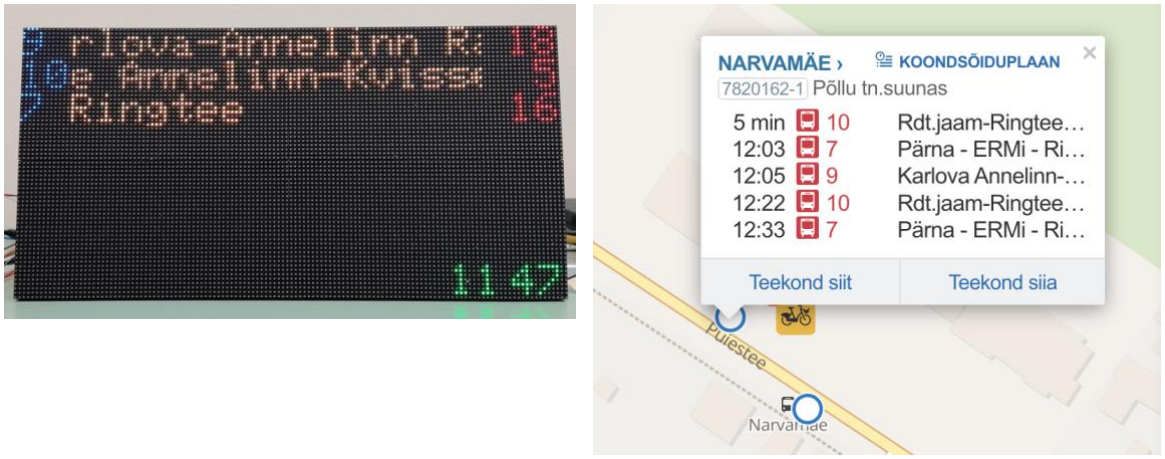
Prototüübi kõigi osade valmimise järel korraldati eksperimente, kus testiti kogu süsteemi toimimist. Järgnevalt kirjeldatakse nelja katse käiku, kus testiti kas valminud prototüüp täidab töö ülesandes loetletud eesmärke.

Ajad peatusest järgmiste busside väljumiseni

Tablood seadistati nii, et see näitaks infot peatuse kohta, mille ühistranspordi andmebaasi identifikaator on „7820162-1" (Taltechi Tartu kolledži lähedal asuv peatus „Narvamäe Põllu tänava suunas").

Andmete õigsust kontrolliti võrreldes tabloo ekraanil kuvatud infot Transpordiameti keskkonna peatus.ee infoga sama peatuse kohta. [68]

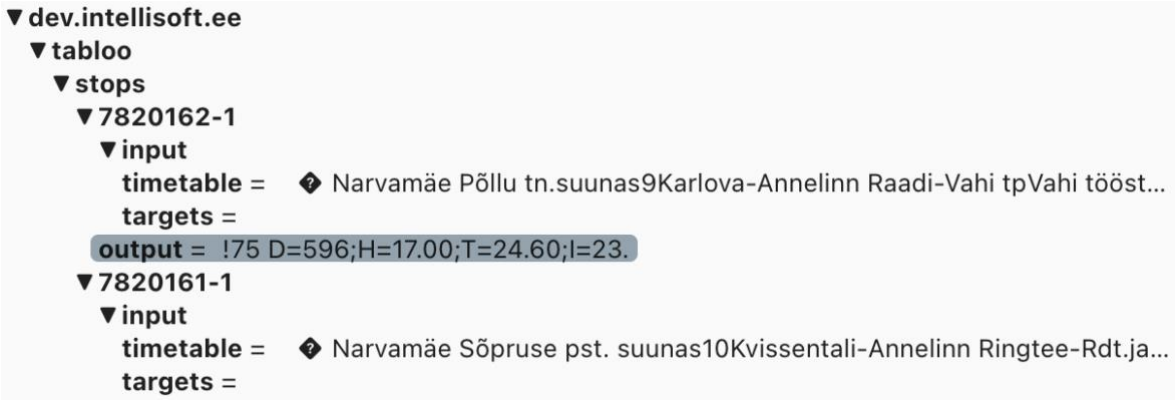
Väljundit võrreldi erinevatel aegadel tööpäevadel ja nädalavahetustel. Tabloo prototüübi esitatud busside ooteajad ei erinenud. Joonis 4.11 illustreerib ühte katsetest.



Joonis 4.11 Tabloo prototüübi (foto vasakul) ja peatus.ee veebikeskkonna (kuvatõmmis paremal) väljundi võrdlus.

Anduri näidismooduli katsetamine

Eeltoodud peatuse infot kuvatava tabloo külge ühendati prototüüpmoodul DHT temperaturi- ja niiskuseanduriga. Prototüübi käivitamise järel see hakkas postitama MQTT vahendajale anduri poolt tulevad andmed.



Joonis 4.12 Andurite väljund postitatakse MQTT vahendajasse vastava tabloo alamteemasse „output" , kust seda võtab vastu mooduli omaniku pistikprogramm. Joonisel on kuvatõmmis tarkvarast MQTT Explorer.

Täituri näidis juhtmooduli katsetamine

Eeltoodud peatuse infot kuvatava tabloo külge ühendati täituri juhtmooduli prototüüp. Mooduli identifitseeriv aadress on "5".

Serveri käsurealt käivitati joonisel 4.13 kujutatud käsk, kus „7820162-1" on tabloot identifitseeriv peatuse kood, „5" on mooduli aadress ühe tabloo koosseisus ja „blink" on käsk, mida peakontroller edastab täituri juhtmoodulile.

Käsu käivitamise järel juhtmooduli prototüübil vilkus LED indikaator.

Joonis 4.13 Täituri juhtmoodulile käsu saatmine terminalist

Sama tulemuse andis REST API-le saadetud tabelis 4.3 esitatud päring.

Tabel 4.3 Täituri juhtmooduli käsu saatmine omaniku poolt REST API kaudu. Klient – curl.

```
curl --location --request POST
'https://dev.intellisoft.ee/tablooapi/modulemessage/7820162-1/5' \
--header 'Authorization: Bearer testapikey' \
--header 'Content-Type: text/plain' \
--data-raw '{"type": "3","message": "blink"}'
```

Käskude edastamine

Analoogselt eelnevale katsele võib saata ka muid käske, näiteks „reboot" käsu saatmine sihile 1 taaskäivitab ekraani kontrolleri. Ekraani kontrollerile (siht 1) võib edastada ka tekste, mida kontroller kuvab ekraanile. Katsetamiseks saadeti tekst „test tekst". Katse tulemusi illustreerib joonis 4.14.

Joonis 4.14 Saadetud teksti kuvamine tabloo ekraanile

5. ARUTELU JA JÄRELDUSED

Töö eesmärgid said täidetud:

- Valmis nutika bussitabloo prototüüp, mis:
    - võib kuvada vabalt valitud Eestis asuva bussipeatuse busside väljumise aegu transpordiameti poolt avaldatud sõiduplaani alusel
    - Peab andmevahetust andurite ja täituritega
- Valmisid nutika bussitabloo moodulite prototüübid (anduri ja täituri prototüübid)

Valmis nutika bussitabloo töötamist toetava taristu prototüüp.

Töö käigus autor sai küberfüüsikalise süsteemi arendamise kogemuse, sai tunda, kui keerukas ja kompleksne võib see olla ning sai katsetada võtteid, mille abil võib süsteemi keerukust ohjata.

5.1. Järgmised sammud prototüübi arenduses

Autor kavatseb jätkata avatud nutika bussitabloo arendusega ja kunagi soovib jõuda tegelikult kasutatava tabloode võrgustikuni. Arusaadav, et reaalselt kasutatava seadmeni on veel pikk tee minna. Reaalsus kindlasti toob korrektiive süsteemi ülesehitusse, aga juba praegu võib välja tuua erinevaid puudujääke, mida tuleb edasises tootearenduses kõrvaldada:

- **Seadme ilmastiku- ja vandaalikindluse tagamine.** Tegelikult kasutatav seade peab toimima üsna laias temperatuuride vahemikus, olema vee ja tolmukindel ning suutma vastu pidada vandalismile.
- **Seadme ergonoomika. Kasutajakogemus. Kuvatava info parem kujundus.** Linnakeskkonnas kasutatav seade peab sulanduma keskkonda, olema visuaalselt meeldiv. Bussitabloo poolt edastatav info peab olema vastuvõetav ja arusaadav väga erineva taustaga inimestele. Palju abi võib siin tulla olemasolevate lahenduste analüüsist.
- **Seadme töökindluse parendamine.** Renditavad moodulid peavad järgima teatud kokkuleppeid (nt voolutarbe osas) ja neid kokkuleppeid tuleb defineerida. Peamoodul peab jälgima, kas moodulid neid kokkuleppeid täidavad. Peamoodulil peavad olema vahendid rikkiläinud või lubamatult käituva mooduli taaskäivitamiseks või väljalülitamiseks nii andmeside- kui ka vooluvõrgust.
    - Tuleb hinnata võimalik moodulite energiavajadus ning vastavalt sellele valida sobiv toiteallikas
    - Moodulite voolutarvet tuleks mõõta (näiteks Halli efekti kasutavate anduritega)

- **Moodulite riistvara põhjendatud valik.** Prototüübi loomiseks kasutati ESP32 mikrokontrollerit. Selle omadused võimaldasid kasutada seda universaalse kontrollerina kõigi moodulite jaoks, mis tegi arenduse palju lihtsamaks. Aga alati selle kasutamine ei ole põhjendatud. Pakutud süsteemi arhitektuur võimaldab erinevate mikrokontrollerite kasutamist.
    - Mõnikord piisab vähem võimsast mikrokontrollerist (soodsam, energiaefektiivsem)
    - Mõnikord tuleb võtta kasutusele spetsiifiliste funktsioonidega riistvara (näiteks 4 RGB LED moodulist koosneva ekraaniga ESP32 töötab oma võimete piiril, teatud andurimoodulite puhul on mõistlik teostada keerulisi arvutusi kohapeal jne)
- **Tarkvara turvalisemaks ja efektiivsemaks muutmine.** Prototüübi arenduse käigus toimus järk-järguline tarkvara parendamine. Võimalusel võeti kasutusele FreeRTOS funktsioone, tarkvara disainis mõeldi turvalisusele. Edaspidi peab seda jätkama.
    - PubSubClient ei pruugi olla parim MQTT klient. See ei toeta kõiki QoS tasemeid
- **Laiendatavuse ja autonoomsuse tõstmine.** Loodud süsteemi pudelikaelteks võib saada I^2C piiratud aadressiruum. Järjestikporti kasutamist ekraanimooduli ühendamiseks võib pidada ajutiseks lahenduseks. Tasub veel katsetada teiste kaasaegsemate ESP32 modifikatsioonidega ja võib-olla saab sellest probleemist üle. Võib mõelda ka teise mikrokontrolleri kasutamise peale selleks, et kõik süsteemi komponendid saaksid kasutada sama siini andmevahetuseks.
- **Seadmete energiaefektiivsuse tõstmine.** Mikrokontroller ESP32 ei pruugi olla kõige efektiivsem energiakasutuse poolest ning võib olla põhjendatud selle asendamine mõne muu mikrokontrolleri vastu teatud moodulites. Enne aga tuleb analüüsida, kas ESP32 töörežiimide häälestus lahendab energia probleeme. Näiteks tavarežiimis mikrokontrollerit läbiv vool on kuni 260 mA, modemite väljalülitamine alandab voolu 3 – 20 mA-ni. On aga olemas režiimid, kus ESP32 voolutarve võib olla vaid 0.8 mA või isegi 10 µA. Madalal energiakulul on aga oma hind, milleks on osade funktsioonide väljalülitamine või madalam arvutusvõimsus. Igat otsust tuleb eraldi kaaluda analüüsides energiarežiimi muutusest saadud kasu ja kahju.
- **Parem ühilduvus.** Prototüüp saab andmeid GTFS formaadis ühe info tarnija käest ja katsete käigus vaadeldud tulemused vastasid tegelikkusele.
    - Tuleb analüüsida, kas saadud andmeid interpreteeritakse alati õigesti, kas kõigil erijuhtudel bussitabloo saab kuvada õigeid andmeid.

    - Teistes riikides võivad andmed olla vormindatud/struktureeritud teisiti. Tuleb kasuks katsetada mujalt tulnud andmetega
    - GTFS ei ole ainuke formaat, mida kasutatakse ühistranspordi andmete edastamiseks. Tasub mõelda serveri migreerimisele GTFS-keskest üldisemale andmete arhitektuurile.
- Lisamoodulite lülitamine. Voolutarve mõõtmine Halli anduritega.

5.2. Idee üldistamine

Lõputöö üks kandvaid ideid on bussipeatuste kasutamine mitmeks otstarbeks, aga sama ideed võib rakendada ükskõik millistele targa linna seadmetele. Prototüübi arenduse käigus jõuti punkti, kus bussitabloo peamoodul täidab abstraktseid, bussiliiklusega otseselt mitte seotud ülesandeid nagu andmevahetuse tagamine ja reaalse kellaaja hankimine teiste moodulite jaoks. Sama ideed võib rakendada valgusfooridele, tänavavalgustuse kontrolleritele, tarkadele prügikastidele [69], valvesüsteemidele jne. Loogilise järgmise sammuna autor näeb targa linna IoT seadmete klastreid, kus toimub ressursside (andmeside, energia, arvutusvõimsus) jagamine nende raiskamise vältimiseks ja dubleerimise vältimiseks.

Tehniliselt see edasiarendus taandab seni süsteemi tuumaks olnud GTFS andmestiku üheks mitmest võimalikust andmekogudest ja kogu bussiliiklust puudutava tegevuse üheks pistiksüsteemiks üldise serveri koosseisus.

KOKKUVÕTE

Bussitabloo on oluline ja reisijatele vajalik targa linna komponent. Lisaks oma otsesele eesmärgile – bussisõiduplaani kasutajasõbralikule esitamisele, nutikat bussitablood võib kasutada ümbritseva keskkonna info kogumiseks või keskkonna mõjutamiseks. Selleks tabloole võib ühendada andureid või täiturite juhtmooduleid.

Iga targa linna IoT võrgustiku loomise juures tuleb lahendada muu hulgas kahte tüüpprobleemi: seadmete energiavarustus ja andmeside. Käesolevas töös uuriti võimalust, kuidas üks IoT võrgustik – nutikate bussitabloode võrk võib pakkuda energiavarustuse ja andmeside teenust tulevaste IoT seadmete võrgustikute jaoks, näiteks lihtsustades sensorvõrkude loomist, võimaldades juhtida valgustust, pakkuda lisateenuseid reisijatele jm.

Töö eesmärgiks oli luua laiendatava funktsionaalsusega nutika bussitabloo prototüüp koos toetava taristuga ja lisatavate moodulite näidistega. Prototüüp pidi kuvama Transpordiameti avaandmetest hangitud bussisõiduplaani vabalt valitud peatuse jaoks, võimaldama tablooga lisamoodulite ühendamist, moodulite kaugjuhtimist ning moodulitelt tuleva info (mõõdised) edastamist moodulite omanikele.

Eesmärgi saavutamiseks analüüsiti küberfüüsikalist süsteemi, tuvastades selle peamisi komponente, kaardistades nendevahelist suhtlust ning eraldades süsteemi alamsüsteemideks ja abstraktsioonitasemeteks. Analüüsiti võimalikke riistvaralisi ja tarkvaralisi komponente ja andmeside tehnoloogiaid prototüübi ehitamiseks. Töötati välja protokollid süsteemi erinevatel tasemetel osalevate agentide suhtlemiseks.

Seadmete prototüübis kasutati ESP32 mikrokontrollereid, mobiilandmesidet ja MQTT protokolli andmevahetuseks. Pandi kokku tabloode toimimist toetav prototüüptaristu. Realiseeriti Transpordiandmete avaandmete import ja tabloode jaoks sobivasse formaati teisendamine.

Arenduse käigus loodi kaks tabloo peakontrolleri prototüüpi, kolm ekraanimooduli prototüüpi, üks anduri ja üks täituri juhtmooduli prototüüp.

Loodi tarkvaraline teek moodulite arenduse lihtsustamiseks ning teegid serveris asuvate pistikmoodulite loomiseks, mis võimaldavad andurite ja täiturite omanikel oma seadmeid kontrollida.

Katsetati bussitabloo prototüübi toimimist: bussisõiduplaanide kuvamist erinevatel aegadel, anduri info edastamist serverisse ja selle serverist väljavõtmise võimalust, moodulite kaugjuhtimist.

Arendus toimus iteratiivselt. Analüüsi, füüsilise prototüübi arenduse, tarkvara arenduse ja katsetamise faasis selgusid uued asjaolud, mis kutsusid kaasa muudatusi loodavas süsteemis.

Autor on rahul töö tulemustega. Tulemused vastavad seatud eesmärkidele ja prototüübi katseid võib seni pidada edukateks. Autor kavatseb arendata tabloo prototüübi ja selle võrgulahendusi edasi eesmärgiga luua funktsioneeriv toode.

SUMMARY

Bus stop information display is an important Smart City component that is welcomed by passengers. In addition to its direct purpose - user-friendly bus schedule presentation, it can be used to collect environmental data or to influence processes in a physical environment. Sensors or actuators could be connected to the bus stop information display to achieve this goal.

Among other task-specific issues, two problems are common for many Smart City network development projects: energy supply and data transfer. Current research explores how one of the networks of IoT devices - a network of bus stop information displays - could provide energy and networking as a service for future networks of IoT devices. Possible applications may be the simplified creation of sensor networks, enabled smart street lighting management, and various additional services for passengers.

The objective of this thesis was to create a prototype of a bus stop information display with extendable functionality along with supporting infrastructure and additional sample modules. The prototype should present the bus schedule according to data published by the Estonian Transport Authority for the stop chosen. The prototype should support the installation and remote management of compatible extension modules. It should be possible to control extension modules distantly and to exchange data (measurements) between extension modules and module owners.

The designed system was handled as a cyberphysical system. The agent-oriented analysis and design methodology were used. The main components of the system were detected, and interactions between different agents were mapped. Different abstraction levels were introduced. Several hardware and software components and network technologies were studied. Protocols were proposed for achieving interactions between agents on different abstraction levels of the system.

ESP32 microcontroller and mobile networking were chosen for the prototype. MQTT was chosen as an application-level protocol. Software infrastructure was created to support prototypes.

The following prototype devices were created: two bus stop information display main controllers, three display module prototypes, one sample sensor module, and one sample actuator control module.

Software frameworks were created to simplify the development of server-side module-owner-specific plugins and firmware for new extension modules.

Several tests of prototype devices and infrastructure were conducted: bus schedule display for different times and different stops, sensor data transfer to the server, and remote control of modules.

An iterative methodology has been used while developing the prototype and supporting infrastructure.

The author is satisfied with the outcome of this thesis - all goals were achieved, and tests of the prototype went successfully. The author plans to continue the development of a bus stop information display prototype to create a fully functional product.

KASUTATUD KIRJENDUSE LOETELU

[1] Cisco, „cisco ioe-value-index-faq.pdf". Vaadatud: 18. aprill 2022. [Online]. Available at: https://www.cisco.com/c/dam/en_us/about/business-insights/docs/ioe-value-index-faq.pdf

[2] J. M. F. Rodrigues, M. Martins, N. Sousa, ja M. Rosa, „IoE Accessible Bus Stop: an initial concept", *Proceedings of the 8th International Conference on Software Development and Technologies for Enhancing Accessibility and Fighting Info-exclusion*, Thessaloniki Greece, juuni 2018, lk 137–143. doi: 10.1145/3218585.3218659.

[3] M. Kubina, D. Šulyová, ja J. Vodák, „Comparison of Smart City Standards, Implementation and Cluster Models of Cities in North America and Europe", *Sustainability*, kd 13, nr 6, lk 3120, märts 2021, doi: 10.3390/su13063120.

[4] K. Tärnov, „Targa linna tippkeskus, TalTech", *Targa linna tippkeskus*. https://taltech.ee/targa-linna-tippkeskus (vaadatud 19. aprill 2022).

[5] H. Ahvenniemi, A. Huovila, I. Pinto-Seppä, ja M. Airaksinen, „What are the differences between sustainable and smart cities?", *Cities*, kd 60, lk 234–245, veebr 2017, doi: 10.1016/j.cities.2016.09.009.

[6] G. Heinmaa, „Peatus on enamat kui prügikast ja paviljon", *Sirp*, 23. juuli 2021. Vaadatud: 19. aprill 2022. [Online]. Available at: https://sirp.ee/s1-artiklid/arhitektuur/peatus-on-enamat-kui-prugikast-ja-paviljon/

[7] K. Grišakov, „Tallinna ühistranspordipeatuste kasutajakogemus Mustamäe linnaosa näitel", lk 96.

[8] J. Järve *et al.*, „Transpordi ja tehiskeskkonna ligipääsetavuse analüüs. Lõppraport", Ligipääsetavuse foorum, Tallinn, 2020. Vaadatud: 19. aprill 2022. [Online]. Available at: https://www.sm.ee/sites/default/files/transpordi_ja_tehiskeskkonna_analyys.pdf

[9] J. Yiu ja A. Frame, „Cortex-M Processors and the Internet of Things (IoT)", lk 14, 2013.

[10] K. Soundarapandian ja A. Kumar, „Challenges of wireless sensor networks and issues associated with time synchronization", märts 2015.

[11] S.-H. Lin, Y.-H. Kuo, E. H.-K. Wu, H.-S. Lin, E. Jou, ja M.-H. Jin, „Smart Rural E-Bus System Using Global Radio (GloRa)", *Sens. Mater.*, kd 33, nr 7, lk 2345, juuli 2021, doi: 10.18494/SAM.2021.3288.

[12] India. *A Smart Bus Stop, Smart Street Lights, Smart Metering and more from Ericsson. Digit.in*, (23. mai 2016). Vaadatud: 19. aprill 2022. [Online Video]. Available at: https://www.youtube.com/watch?v=gB7HoLHDwcs

[13] „Compact LED Bus Shelter Display", *Compact LED Bus Shelter Display*. https://trueform.com/products/compact-led-bus-shelter-display/ (vaadatud 26. aprill 2022).

[14] T. Boshita, H. Suzuki, ja Y. Matsumoto, „Smart Bus Stop using LoRaWAN and e-Paper", *2019 IEEE 8th Global Conference on Consumer Electronics (GCCE)*, Osaka, Japan, okt 2019, lk 278–282. doi: 10.1109/GCCE46687.2019.9015448.

[15] T. Boshita, H. Suzuki, ja Y. Matsumoto, „IoT-based Bus Location System Using LoRaWAN", *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, Maui, HI, nov 2018, lk 933–938. doi: 10.1109/ITSC.2018.8569920.

[16] B. Foubert ja N. Mitton, „Long-Range Wireless Radio Technologies: A Survey", *Future Internet*, kd 12, nr 1, lk 13, jaan 2020, doi: 10.3390/fi12010013.

[17] „Ühistranspordi infosüsteem", *Transpordiamet*, 6. mai 2021. https://transpordiamet.ee/liikuvus-ja-transpordikorraldus/uhistransport/uhistranspordi-infosusteem (vaadatud 23. aprill 2022).

[18] M. Wooldridge, „The Gaia Methodology for Agent-Oriented Analysis and Design", lk 28.

[19] X. Lu, P. Wang, D. Niyato, D. I. Kim, ja Z. Han, „Wireless Charging Technologies: Fundamentals, Standards, and Network Applications", *IEEE Commun. Surv. Tutor.*, kd 18, nr 2, lk 1413–1452, 2016, doi: 10.1109/COMST.2015.2499783.

[20] I. Roos, A. Brandt, ja P. Pikner, „Ühistranspordiregistri avaandmete spetsifikatsioon". Maanteeamet, 3. märts 2020. Vaadatud: 6. september 2022. [Online]. Available at: https://www.transpordiamet.ee/media/1339/download

[21] „What is LoRaWAN® Specification", *LoRa Alliance*. https://lora-alliance.org/about-lorawan/ (vaadatud 26. september 2022).

[22] „What are the differences between 2G, 3G, 4G LTE, and 5G networks?", *rantcell.com*. https://rantcell.com/comparison-of-2g-3g-4g-5g.html (vaadatud 27. september 2022).

[23] J. M. Khurpade, D. Rao, ja Parth. D. Sanghavi, „A Survey on IOT and 5G Network", *2018 International Conference on Smart City and Emerging Technology (ICSCET)*, jaan 2018, lk 1–3. doi: 10.1109/ICSCET.2018.8537340.

[24] „ESP32 Series Datasheet Version 3.9". Espressif Systems, 2022. Vaadatud: 5. september 2022. [Online]. Available at: https://www.espressif.com/en/support/download/documents

[25] „ESP32-WROOM-32 Datasheet Version 3.3". Espressif Systems, 2022. Vaadatud: 5. september 2022. [Online]. Available at: https://www.espressif.com/en/support/download/documents

[26] „ESP32-WROVER-B & ESP32-WROVER-IB Datasheet Version 1.8“. Espressif Systems, 2022. Vaadatud: 5. september 2022. [Online]. Available at: https://www.espressif.com/en/support/download/documents

[27] R. P. Ltd, „Raspberry Pi 4 Model B specifications“, *Raspberry Pi*. https://www.raspberrypi.com/products/raspberry-pi-4-model-b/ (vaadatud 21. detsember 2022).

[28] „Arduino Nano Every“, *Arduino Official Store*. https://store.arduino.cc/products/arduino-nano-every (vaadatud 21. detsember 2022).

[29] „Battery solution for Nano Every Project - Nano Family / Nano Every“, *Arduino Forum*, 6. aprill 2022. https://forum.arduino.cc/t/battery-solution-for-nano-every-project/977829 (vaadatud 21. detsember 2022).

[30] „RP2040 Thing Plus Hookup Guide - SparkFun Learn“. https://learn.sparkfun.com/tutorials/rp2040-thing-plus-hookup-guide/hardware-overview (vaadatud 30. detsember 2022).

[31] „Raspberry Pi 4 Model B | The Pi Hut“. https://thepihut.com/products/raspberry-pi-4-model-b?src=raspberrypi&variant=31994565689406 (vaadatud 21. detsember 2022).

[32] A. Voronova, „Did You Know That The Raspberry Pi 4 Has More SPI, I2C, UART Ports?“, *Hackaday*, 1. veebruar 2022. https://hackaday.com/2022/02/01/did-you-know-that-the-raspberry-pi-4-has-more-spi-i2c-uart-ports/ (vaadatud 21. detsember 2022).

[33] „Power Consumption Benchmarks | Raspberry Pi Dramble“. https://www.pidramble.com/wiki/benchmarks/power-consumption (vaadatud 21. detsember 2022).

[34] „E-paper bus stop passenger information displays“, *Papercast*. https://www.papercast.com/E-paper-display-applications/bus-stops/ (vaadatud 6. september 2022).

[35] „Smart Bus Station“, *Marvel Technology*. https://www.outdoor-lcddisplay.com/product/smart-bus-station/ (vaadatud 6. september 2022).

[36] „Display 64x32 Dot Matrix Smd2121 Led Module Indoor Screen Full Color Video Wall 192x96mm Message Board - Led Displays - AliExpress“, *aliexpress.com*. //www.aliexpress.com/item/32728985432.html?src=ibdm_d03p0558e02r02&sk=&aff_platform=&aff_trace_key=&af=&cv=&cn=&dp= (vaadatud 11. november 2022).

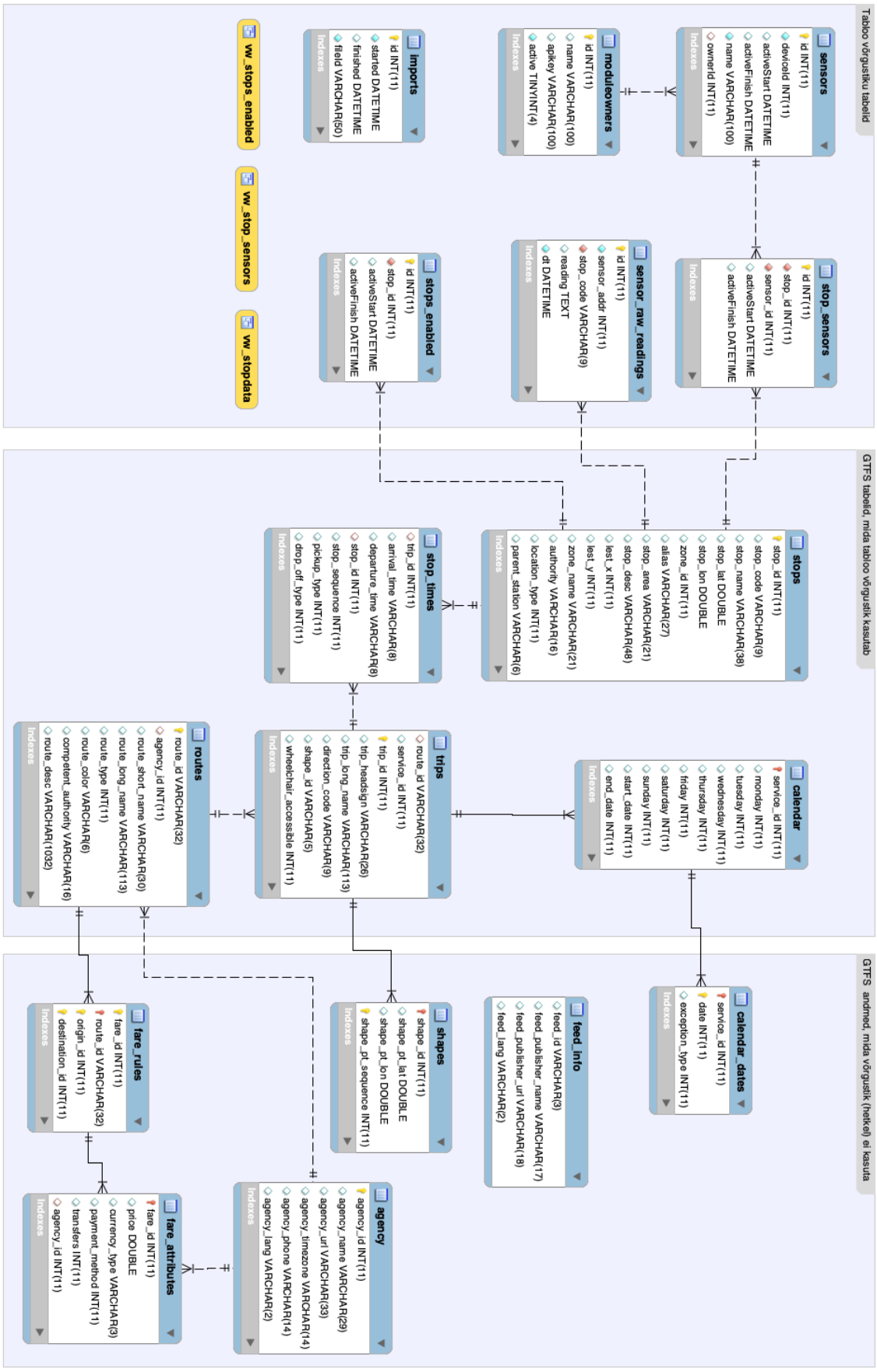
[37] „14.11€ |2.9" 2.9 Inch Epaper Module E-paper E-ink Eink Display Screen Spi Support For Arduino Uno Stm32 Raspberry Pi Esp32 - Integrated Circuits - AliExpress“, *aliexpress.com*.

//www.aliexpress.com/item/1005003648220347.html?src=ibdm_d03p0558e02r02&sk=&aff_platform=&aff_trace_key=&af=&cv=&cn=&dp= (vaadatud 28. november 2022).
[38] „Enterprise Open Source and Linux", *Ubuntu*. https://ubuntu.com/ (vaadatud 28. november 2022).
[39] „Welcome! - The Apache HTTP Server Project". https://httpd.apache.org/ (vaadatud 28. november 2022).
[40] „MySQL :: MySQL Community Edition". https://www.mysql.com/products/community/ (vaadatud 28. november 2022).
[41] „Eclipse Mosquitto", *Eclipse Mosquitto*, 8. jaanuar 2018. https://mosquitto.org/ (vaadatud 28. november 2022).
[42] „PHP: Hypertext Preprocessor". https://www.php.net/index.php (vaadatud 28. november 2022).
[43] „crontab". https://pubs.opengroup.org/onlinepubs/9699919799/utilities/crontab.html (vaadatud 28. november 2022).
[44] „Arduino core for the ESP32, ESP32-S2, ESP32-S3 and ESP32-C3". Espressif Systems, 18. oktoober 2022. Vaadatud: 18. oktoober 2022. [Online]. Available at: https://github.com/espressif/arduino-esp32
[45] „Visual Studio Code. Code editing. Redefined." https://code.visualstudio.com/ (vaadatud 6. september 2022).
[46] „Professional collaborative platform for embedded development", *platformio.org*. https://platformio.org/ (vaadatud 6. september 2022).
[47] „Git". https://git-scm.com/ (vaadatud 6. september 2022).
[48] „Build software better, together", *GitHub*. https://github.com (vaadatud 21. detsember 2022).
[49] „MySQL :: MySQL Workbench". https://www.mysql.com/products/workbench/ (vaadatud 21. detsember 2022).
[50] „Diagram Software and Flowchart Maker". https://www.diagrams.net/ (vaadatud 21. detsember 2022).
[51] T. Nordquist, „MQTT Explorer", *MQTT Explorer*. http://mqtt-explorer.com/ (vaadatud 2. jaanuar 2023).
[52] I. Craggs, „MQTT Vs. HTTP: Understanding the Differences", *HiveMQ*, 16. mai 2022. https://www.hivemq.com/blog/mqtt-vs-http-protocols-in-iot-iiot/ (vaadatud 3. september 2022).
[53] C. Wang, „HTTP vs. MQTT: A tale of two IoT protocols", *Google Cloud Blog*, 26. november 2018. https://cloud.google.com/blog/products/iot-devices/http-vs-mqtt-a-tale-of-two-iot-protocols (vaadatud 2. oktoober 2022).

[54] „FreeRTOS - User Guide“, *2022*, lk 490.
[55] Gnusmas, „Как получить актуальное время от GSM модуля, даже если оператор его не дает“, *Сообщество EasyElectronics.ru*, 29. oktoober 2018. http://we.easyelectronics.ru/Soft/kak-poluchit-aktualnoe-vremya-ot-gsm-modulya-dazhe-esli-operator-ego-ne-daet.html (vaadatud 2. oktoober 2022).
[56] I. Tihhanovski, „Avatud bussitabloo repositoorium“. 9. detsember 2021. Vaadatud: 11. oktoober 2022. [Online]. Available at: https://github.com/tihhanovski/tabloo
[57] I. Tihhanovski, „Avatud bussitabloo veebiserver“. https://dev.intellisoft.ee/tabloo/
[58] „Tark Tartu“. https://tarktartu.telia.ee/et/ (vaadatud 21. detsember 2022).
[59] „UTC – Coordinated Universal Time“. https://www.timeanddate.com/time/aboututc.html (vaadatud 28. november 2022).
[60] „API Reference - ESP32 - — ESP-IDF Programming Guide latest documentation“. https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-reference/index.html (vaadatud 18. oktoober 2022).
[61] „SIM800“, *SimCom Wireless Solutions*. https://www.simcom.com/product/SIM800.html (vaadatud 23. november 2022).
[62] h2zero, „NimBLE-Arduino“. 20. detsember 2022. Vaadatud: 21. detsember 2022. [Online]. Available at: https://github.com/h2zero/NimBLE-Arduino
[63] „TinyGSM/HttpsClient.ino at master · vshymanskyy/TinyGSM“, *GitHub*. https://github.com/vshymanskyy/TinyGSM (vaadatud 18. oktoober 2022).
[64] N. O’Leary, „Arduino Client for MQTT“. 18. oktoober 2022. Vaadatud: 18. oktoober 2022. [Online]. Available at: https://github.com/knolleary/pubsubclient
[65] „Trouble with i2c and i2s running simultaneously - ESP32 Forum“. https://esp32.com/viewtopic.php?t=2902 (vaadatud 11. oktoober 2022).
[66] mrfaptastic, „HUB75 RGB LED matrix library utilizing ESP32 DMA Engine“. 21. detsember 2022. Vaadatud: 22. detsember 2022. [Online]. Available at: https://github.com/mrfaptastic/ESP32-HUB75-MatrixPanel-DMA
[67] ZinggJM, „GxEPD“. 18. detsember 2022. Vaadatud: 22. detsember 2022. [Online]. Available at: https://github.com/ZinggJM/GxEPD
[68] „Digitransit“, *Digitransit*. https://web.peatus.ee/ (vaadatud 2. jaanuar 2023).
[69] https://www.studiox.bg, „Leuven is experimenting with smart trash bins | TheMayor.EU“. https://www.themayor.eu/en/a/view/leuven-is-experimenting-with-smart-trash-bins-8437 (vaadatud 11. november 2022).

LISAD

Lisa 1. Andmebaasi struktuur



Lisa 2. Seadme elektriline skeem