TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Siiri Mangus 192844IADB

Koduautomaatika süsteemi oleku haldamine Node-RED näitel

Bakalaureusetöö

Juhendaja: Kristiina Hakk
PhD

Kaasjuhendaja: Mikk Mangus
BSc

Tallinn 2023

Autorideklaratsioon

Kinnitan, et olen koostanud antud lõputöö iseseisvalt ning seda ei ole kellegi teise poolt varem kaitsmisele esitatud. Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on töös viidatud.

Autor: Siiri Mangus

15.05.2023

Annotatsioon

Meie tehnoloogiaga põimitud maailmas ehitatakse targa kodu ja koduautomaatika süsteeme üha rohkem. Kuigi mõisteid tark kodu ja koduautomaatika kasutatakse sageli samas tähenduses, on need oma sisult mõnevõrra erinevad. Tark kodu tähendab kasutajapoolse sekkumisega töötavat süsteemi ning koduautomaatika on süsteem, mis suudab iseseisvalt toimida.

Käesolev bakalaureusetöö käsitleb erinevate targa kodu ja koduautomaatika platvormide võimekusi seadistada koduautomaatika reegleid. Osutub, et kommertsiaalsed platvormid, mis kasutaja jaoks on lihtsad seadistada, ei võimalda keerukamaid mitme tingimusega automaatika reegleid konfigureerida. Seda on võimalik teha kogukonna poolt hallatavatel automaatika platvormidel.

Selleks, et ehitada paljudest erinevatest lähtetingimustest sõltuvat automaatika reeglit, on tarvilik kasutada tingimuslausete konstruktsioone. Nende haldamine muutub iga tingimuse lisandumisega raskemaks ning veaohtlikumaks. Töö praktilises osas lahendatakse antud probleem automaatika platvormil Node-RED uue arhitektuurilise lähenemisega – süsteemi järgmist seisu ei tuletata üksikutest sisendsignaalidest vaid süsteemi olekust.

Lõputöö on kirjutatud eesti keeles ning sisaldab teksti 36 leheküljel, 6 peatükki, 24 joonist, 2 tabelit.

Abstract

State Management in Home Automation on the Example of Node-RED

In the world intertwined with technology, smart homes and home automation systems are being built increasingly. Although the terms smart home and home automation are often used interchangeably, they are somewhat different in their essence. A smart home refers to a system that operates with user intervention, while home automation is a system that can operate independently.

This thesis analyzes the capability of various smart home and home automation platforms to configure home automation rules. It turns out that commercial platforms, which are easy for users to configure, tend to not allow for more complex automation rules with multiple conditions to be configured. This can be done on community-managed automation platforms.

In order to build an automation rule that depends on many different starting conditions, it is necessary to use conditional clause constructions. Managing those becomes more difficult and error-prone with each additional condition. In the practical part of the work, this problem is solved with a new architectural approach on the Node-RED automation platform – the next state of the system is not derived from individual input signals, but from the system's state.

The thesis is in Estonian and contains 36 pages of text, 6 chapters, 24 figures, 2 tables.

Lühendite ja mõistete sõnastik

Amazon Alexa	Firma Amazon poolt hallatav pilvepõhine häälteenus
Android	Avatud lähtekoodiga operatsioonisüsteem mobiilsetele seadmetele
API	*Application Programming Interface*, rakendusliides, eri tarkvarakomponentide vaheline selgelt määratletud sidevahendite liides
Apple HomeKit	Firma Apple poolt arendatud targa kodu platvorm
Blockly	Kliendipoolne teek, mis loob plokkidel baseeruva visuaalse programmeerimiskeele
Control4	Kodude ja ettevõtete automaatika- ja võrgusüsteemide pakkuja
Crestron	Kodude ja ettevõtete automaatikasüsteemi pakkuja
DIY	*Do it yourself*, ise ehitatud
Docker	Tarkvara, mis teostab operatsioonisüsteemi tasemel virtualiseerimist, võimaldab ehitada, jooksutada ning hallata tarkvara konteinereid
Domoticz	Avatud lähtekoodiga tarkvara koduautomaatika jaoks
ECMAScript	Skriptikeelte standard
Google Assistant	Firma Google virtuaalne tehisintellektil baseeruv abitehnoloogia
Google Home	Firma Google poolt arendatud targa kodu platvorm
Home Assistant	Avatud lähtekoodiga tarkvara koduautomaatika jaoks
HTML	*Hyper Text Markup Language*, märgistuskeel veebilehtede loomiseks
iOS	Firma Apple poolt arendatud operatsioonisüsteem mobiilsetele Apple'i seadmetele
IP	*Internet Protocol*, peamine sideprotokoll, mida järgitakse andmepakettide saatmisel internetis kasutatavate võrguseadmete vahel
JavaScript	Skriptikeel
JQuery	JavaScripti teek, mis keskendub JavaScripti ja HTML-i vastastikusele mõjule
JSON	*JavaScript Object Notation*, andmevahetusformaat

MQTT	*Message Queuing Telemetry Transport*, masinside protokoll
Node.js	Avatud lähtekoodiga platvormist sõltumatu JavaScripti käitussüsteem
Node-RED	Voopõhine arendustööriist visuaalseks programmeerimiseks
Npm	*Node package manager*, Node.js jaoks loodud paketihaldur
OpenHAB	Avatud lähtekoodiga automatiseerimisplatvorm
Raspberry Pi	Väike, ühest trükkplaadist koosnev arvuti
React	JavaScripti teek kasutajaliideste arenduseks
Rules API	Samsung SmartThings API loomaks ning haldamaks automaatika reegleid SmartThings platvormil
Rules DSL	*Domain-Specific Language*, kõrge abstraktsioonitasemega programmeerimiskeel, mis on optimeeritud mingi konkreetse valdkonna jaoks, Rules DSL on arendatud spetsiaalselt OpenHAB platvormi jaoks
Samsung SmartThings	Firma Samsung poolt arendatud targa kodu platvorm
TCP/IP	*Transmission Control Protocol/Internet Protocol*, protokollistik sidumaks võrguseadmeid internetis
Thread	Juhtmevaba võrktopoloogiaga võrguprotokoll, mida kasutatakse sageli koduautomaatika süsteemides
Wi-Fi	Juhtmevaba võrguprotokoll, mida kasutatakse peamiselt kohtvõrgu loomiseks ning internetiga ühendumiseks
YAML	*YAML Ain't Markup Language*, inimloetav andmete serialiseerimise keel, mida kasutatakse peamiselt konfiguratsiooni failide kirjutamiseks
Z-Wave	Juhtmevaba võrktopoloogiaga võrguprotokoll, mida kasutatakse sageli koduautomaatika süsteemides
ZigBee	Juhtmevaba väikest energiakulu võimaldav võrktopoloogiaga võrguprotokoll, mida kasutatakse sageli koduautomaatika süsteemides

Sisukord

Jooniste loetelu

Tabelite loetelu

1 Sissejuhatus

Tarku ja automatiseeritud kodusid ehitatakse meie digitaliseeruvas maailmas üha rohkem. Tänu tehnoloogiale on võimalik tõsta kodu turvalisust, energiatõhusust ja mugavust muutes kodu kohandatavaks vastavalt inimeste vajadustele ja soovidele.

Tark kodu on kodu, mida saab juhtida tarkvaraliste lahenduste abil. Koduautomaatika on tehnoloogia, mis võimaldab kodu juhtida kasutaja sekkumiseta, st automatiseeritud seadmed suudavad toimida iseseisvalt.

Koduautomaatika süsteemid on üldjuhul üles ehitatud selliselt, et need reageerivad sündmuste toimumisele, näiteks anduritest tulenevate signaalide muutumisele. Kui aga sündmuseid/signaale, millele üks väljund (täitur) peab reageerima, on palju erinevaid, muutub oluliseks süsteemi oleku haldamine, sest automaatika reeglite seadistamise keerukus kasvab iga lisandunud signaaliga kiirelt. Paljude sisendsignaalide haldamine on aga miski, millele pööratakse koduautomaatika kontekstis vähe tähelepanu ja selle jaoks pole välja kujunenud parimaid praktikaid.

Käesoleva töö eesmärk on analüüsida, kas ja kuidas on lahendatud paljude sisendsignaalide haldus hetkel populaarsemates koduautomaatika süsteemides.

Töö teine eesmärk on arendada vabavaraline laiendus automaatika platvormile Node-RED, mis võimaldab hallata süsteemi olekut ning reageerida oleku muutustele, mitte sisendsignaalidele. Antud laiendus võimaldab konkreetse täituri olekut kontrollida ühes keskses kohas, mis omakorda aitab vähendada täituri mittesoovitud olekuid ja tagada, et ehitatud koduautomaatika töötab nii nagu eeldatud, ilma vigadeta.

Töö esimeses osas antakse ülevaade targast kodust ja koduautomaatikast, sõnastatakse projekti eesmärk. Teises osas käsitletakse hetkel populaarsemaid koduautomaatika süsteeme ning analüüsitakse nende võimekust tulla toime automaatika reeglitega, mis sõltuvad paljudest erinevatest sisendsignaalidest. Töö lõpuosas kirjeldatakse, kuidas arendati laiendus Node-RED süsteemile ning kuidas seda kasutada.

2 Probleemi taust ja projekti eesmärk

Targad kodud ja koduautomaatika muutuvad üha populaarsemaks, sest tehnoloogia arenguga tekib enam võimalusi oma elu lihtsustada automatiseerides tehnilisi nüansse oma elukeskkonnas. Globaalset targa kodu turgu hinnati 2022. aastal 79,16 miljardi dollari suuruseks. Eeldatakse, et 2030. aastaks on selle väärtus juba 537 miljardit dollarit. [1]

Suurimaks targa kodu turuliidriks on olnud Ameerika Ühendriigid. Aastal 2022 kasutati seal nutikaid koduseadmeid vähemalt üks kord kuus 57,4 miljonis majapidamises. Käesolevaks 2023. aastaks ennustatakse eelmise aastaga võrreldes 5% kasvu, s.o 60,4 miljonit majapidamist kasutab aktiivselt targa kodu võimalusi. See omakorda moodustab 46,5% kõikidest Ameerika Ühendriikide majapidamistest, mis tähendab, et peaaegu pooled majapidamised suhtlevad igakuiselt mingi targa kodu seadmega. [2]

Järgnevalt antakse ülevaade, millist kodu peetakse targaks, mis vahe on targal kodul ja koduautomaatikal, millised on targa kodu tüübid, ning milles seisneb oleku halduse probleem koduautomaatika süsteemides.

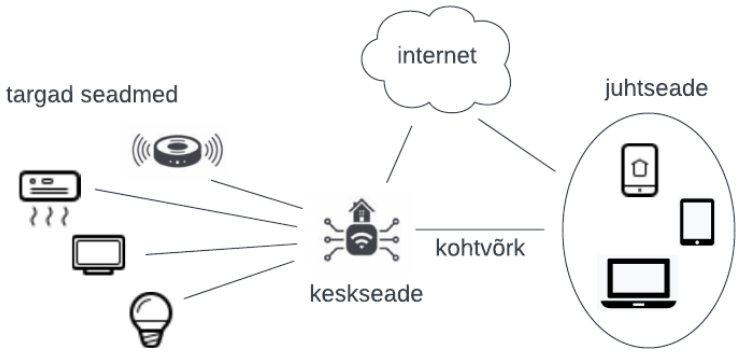
2.1 Tark kodu

Targa kodu mõistet on kujundatud alates 1990. aastast. Teadlane Satpathy nimetab targaks kodu eluaset, mis on piisavalt nutikas abistamaks elanikel elada mugavalt tehnoloogia toel. Lisaks, et targas kodus on mehaanilised ja digitaalsed seadmed omavahel ühendatud, moodustades selliselt ühise võrgustiku, mis võimaldab neil omavahel ning kasutajatega suhelda, luues seeläbi interaktiivse ruumi. [3]

Teadlased Alam ja Ali defineerivad tarka kodu kui rakendust, mis võimaldab automatiseerida või toetada kasutaja tegevusi mitmes aspektis, näiteks kodu eemalt juhtimises, koduautomaatikas või ümbritseva ruumi intelligentsuses. Üldistatult võib öelda, et kõik targa kodu kirjeldused seavad esikohale elanike igapäevaelu mugavdamise ning lihtsustamise tehnoloogia abil. [3]

On palju erinevaid tehnoloogiaid ja seadmeid, mille abil tark kodu üles seada. Nutikates kodudes võivad koduomanikud potentsiaalselt kontrollida kõike alates valgustusest ja temperatuurist kuni kodude turvasüsteemideni välja oma nutitelefonide, tahvelarvutite või muude seadmete abil. [4]

Kõige sagedasem targa kodu korraldus on kujutatud joonisel 1, kus targad seadmed ühenduvad keskseadmega (*hub*), mis omakorda on ühendatud kasutajapoolse juhtimissüsteemiga. Suhtlus keskseadme ja juhtseadme vahel võib toimuda ka läbi pilve.



Joonis 1. Targa kodu seadmete korraldus.

2.2 Tark kodu ja koduautomaatika

Targa kodu ja koduautomaatika mõisteid kasutatakse nende olemuse suure sarnasuse tõttu sageli samas tähenduses. Sellegipoolest on terminoloogiliselt täpsem neil vahet teha. [5]

Tark kodu on süsteem, mis vajab kasutaja sekkumist erinevate seadmete juhtimiseks ja seadistamiseks, olgu selleks külmkapid, lambid või kõlarid. Juhtimiseks kasutatakse pulti, juhtpaneeli või häälkäskluseid. Näiteks võimaldab targa kodu süsteem elektroonses juhtpaneelis nupule vajutusega aknakardinaid ette tõmmata või häälkäsklusega lambi kustu panna. [6], [7]

Koduautomaatika süsteemis olevad seadmed ei vaja kasutaja sekkumist selleks, et need tööle läheksid, st kõik on täielikult automatiseeritud. Näiteks lülituvad radiaatorid sõltuvalt vajadusest ise sisse ning välja või läheb muusika mängima juhul, kui toas on hämar. See tähendab, et automatiseeritud süsteemis töötavad seadmed automaatselt reageerides kodus olevale keskkonnale. [6], [7]

Ülimalt targa kodu üles seadmiseks võiks kombineerida manuaalse targa kodu juhtimise, kus vajalik, ning automatiseerida kõik, mis võimalik. Automatiseeritud süsteemide seadistamine on kahtlemata oluliselt keerukam, kuid selle tulemuseks on kodu, mis märkamatult mõistab oma elanikke ja loob neile mugavaima elukeskkonna. [7]

2.3 Targa kodu ja koduautomaatika süsteemide jaotus

Targa kodu ja koduautomaatika süsteemid jaotatakse kolme kategooriasse [8]:

- kommertsiaalsed (*commercial*);
- ühisvara, kogukonna poolt hallatud (*community*);
- ise ehitatud (*DIY* ehk *do it yourself*).

Kõige populaarsemad on kaks esimest süsteemi, kuna need pakuvad tarkvara pidevat tuge juhul, kui seadmed uuenevad. Samas, kui on huvi põhjalikumalt õppida koduautomaatika tehnoloogiaid, siis enda kokku pandud süsteem aitab sellele eesmärgile kõige paremini kaasa. [8]

Kommertsiaalsed süsteemid jagunevad omakorda kaheks [8]:

- Süsteemid, mida kasutaja saab ise paigaldada. Siin on enamjaolt tegemist pilvepõhiste teenuspakkujatega, mille kaudu saab seadmeid juhtida. Tuntuimad tootjad on Google Home, Amazon Alexa, Apple HomeKit ja Samsung SmartThings.
- Süsteemid, mida paigaldavad professionaalsed tehnikud ning mida saab osta ainult tunnustatud edasimüüjatelt. Sellised süsteemid on näiteks Control4 ja Crestron.

Kogukonna poolt hallatud tuntuimad süsteemid on Home Assistant, Domoticz ja openHAB. Ise ehitatud süsteemide jaoks kasutatakse enim automaatika platvormi Node-RED ning Raspberry Pi arvutit. [8]

2.4 Protokollid targa kodu süsteemides

Selleks, et tarka kodu juhtida ning automaatseid süsteeme luua, peavad seadmed nii omavahel kui ka juhtseadmega suhtlema. Kui seadmed on toodetud erinevate tootjate

poolt erinevat riistvara ning tarkvara kasutades, siis üldjuhul ei suuda nad omavahel ühendust luua. Siinkohal on tarvilikud ühilduvad protokollid.

Koduautomaatika kontekstis ei ole välja kujunenud üht kindlat suhtlusstandardit nagu seda on näiteks internetisuhtluse puhul, kus valitseb TCP/IP protokoll (*Transmission Control Protocol/Internet Protocol*). See tähendab, et targa kodu seadmed ühilduvad omavahel ainult juhul, kui need kasutavad suhtlemiseks sama protokolli, või kui süsteemis on keskseade, millega kõik seadmed suhtlevad, ning mis tegutseb tõlkija ja info vahendajana. [9] Populaarseimad protokollid targa kodu süsteemides on järgnevad:

- Wi-Fi – Juhtmevaba võrguprotokoll, mida kasutatakse peamiselt kohtvõrgu loomiseks ning internetiga ühendumiseks. Wi-Fi üheks plussiks on asjaolu, et see on enamikes kodudes olemas, mistõttu ei ole tarvilik targa kodu jaoks eraldi võrku seadistama hakata. Wi-Fi miinuseks on tõik, et rohkete võrgus olevate seadmete korral ei pruugi need eranditult kiirelt töötada. Lisaks tarbib suhtlus palju energiat, mistõttu peab patareide pealt töötavaid seadmeid sageli hooldama. [9]
- Z-Wave – Spetsiaalselt koduautomaatika jaoks disainitud juhtmevaba võrguprotokoll. Z-wave süsteem on võrktopoloogiaga andmesidevõrk (*mesh network*), mis tähendab, et kõik võrgus olevad seadmed saavad omavahel suhelda. Seega kogu võrgu ulatus ning ka usaldusväärsus kasvab iga lisatava seadmega. Juhul, kui seadmel puudub ühendus keskseadmega, siis saab ta ikkagi infot saata läbi teiste seadmete. [9]– [10]
- ZigBee – Spetsiaalselt koduautomaatika jaoks disainitud juhtmevaba võrguprotokoll. Selle protokolli suur eelis on väike energiakulu, mistõttu sobib see ideaalselt patareide pealt töötavate seadmete jaoks. ZigBee süsteem on võrktopoloogiaga, see tähendab, et seadmed võrgus võivad toimida releedena, kui otseühendus kahe seadme vahel puudub. [11]
- Thread – Suhteliselt uus (loodud 2014. aastal) juhtmevaba võrguprotokoll. Tegemist on võrktopoloogia süsteemiga, kuid teistest siiani vaadeldud protokollidest eristub Thread selle poolest, et see kasutab IP-l (*Internet Protocol*) baseeruvat võrku, võrdluseks Z-Wave ja ZigBee kasutavad eraldiseisvat võrku. Sõnumite saatmine Threadis on madala energiakuluga. [11], [12]
- MQTT – Akronüüm sõnadele *Message Queuing Telemetry Transport*. Tegemist on masinside (*machine-to-machine*) protokolliga, mis tavaliselt toimib üle TCP/IP võrgu. MQTT on lihtne ning kergekaaluline sõnumside protokoll, mida

kasutatakse madala ülekandekiiruse seadmete puhul. See töötab saatja/vastuvõtja (*publish/subscriber*) mudeli baasil. See tähendab, et on kliendid, kes infot saadavad, ning on kliendid, kes seda infot vastu võtavad. Samas saatjad ning vastuvõtjad ei ole üksteise olemasolust üldse teadlikud. Saadetud sõnumeid edastab vastuvõtjatele vahendaja (*broker*). [11], [10]

2.5 Süsteemi oleku haldamise probleem

Käesolevas töös keskendutakse koduautomaatikale mitte targale kodule, st vaadeldakse süsteeme, mis ei vaja kasutaja sekkumist selleks, et need tööle läheksid. Näiteks tulede või küttesüsteemi automaatne sisse-välja lülitus.

Koduautomaatika süsteemid on üldjuhul üles ehitatud selliselt, et need reageerivad sündmuste toimumisele, näiteks anduritest tulenevate signaalide muutumisele. Võib eeldada, et koduautomaatika toimib vigadeta juhul, kui väljund (täitur) peab reageerima vaid ühele sündmusele.

Kujutame ette olukorda, kus lambi lülitamise loogika sõltub kahest sündmusest – liikumisest ning valgustugevusest toas. Olgu meil aknaga tuba, kus soovime lambi sisse lülitada juhul, kui seal on pime ning liigutakse. See tähendab, kui toas on liikumine, aga valgust on piisavalt (aknast tuleva päevavalguse tõttu), siis ei tohiks lamp põlema minna. Pealtnäha lihtsalt paari tingimuslausega lahendatav tegevuste voog viib meid aga kohe loogikaveani. Nimelt, kui liigume pimedas toas, läheb lamp põlema. Nüüd aga on toas valgust piisavalt ning lamp läheb seetõttu kustu. Seejärel on toas taaskord pime ning lamp läheb uuesti põlema. Kokkuvõttes jääb lamp vilkuma.

Osutub, et kui sündmuseid/signaale, millele üks täitur peab reageerima, on palju, siis muutub oluliseks, mil moel süsteemi olekut hallatakse, selleks, et see toimiks ootuspäraselt ilma vigadeta. Paljude sisendsignaalide haldamine on aga miski, millele pööratakse koduautomaatika kontekstis vähe tähelepanu ja selle jaoks pole välja kujunenud parimaid praktikaid.

2.6 Projekti eesmärk

Käesoleva töö eesmärgiks on analüüsida hetkel populaarsemate koduautomaatika platvormide võimekust seadistada paljudest erinevatest sisendsignaalidest sõltuvaid automaatika reegleid. Lisaks uuritakse, kas nendes süsteemides on mingeid abivahendeid paljude sisendsignaalide haldamiseks, et automaatika reeglite seadistamine oleks võimalikult veakindel.

Töö praktilises osas arendatakse laiendus platvormile Node-RED, mis peab lihtsustama keerukate paljude sisendsignaalidega automaatika reeglite seadistamist. Laiendus peab võimaldama süsteemis kontrollida konkreetse täituri olekut ühes keskses kohas.

3 Erinevate koduautomaatika süsteemide analüüs

Käesolevas peatükis võetakse vaatluse alla mõned täna populaarsed targa kodu ja koduautomaatika süsteemid. Uuritakse, kas ja kuidas on nendes võimalik seadistada automaatika reegleid, mis sõltuvad mitmest sisendsignaalist. Kaarel Allemann väidab oma 2017. aastal kirjutatud koduautomaatika teemalises bakalaureusetöös, et „mida lihtsamaks on muudetud seadmete ühendamine süsteemi ning süsteemi haldamine tavakasutajale, seda vähem omab süsteem ka funktsionaalsust“ [13]. Hüpotees on, et nüüd, kuus aastat hiljem, on olukord sarnane.

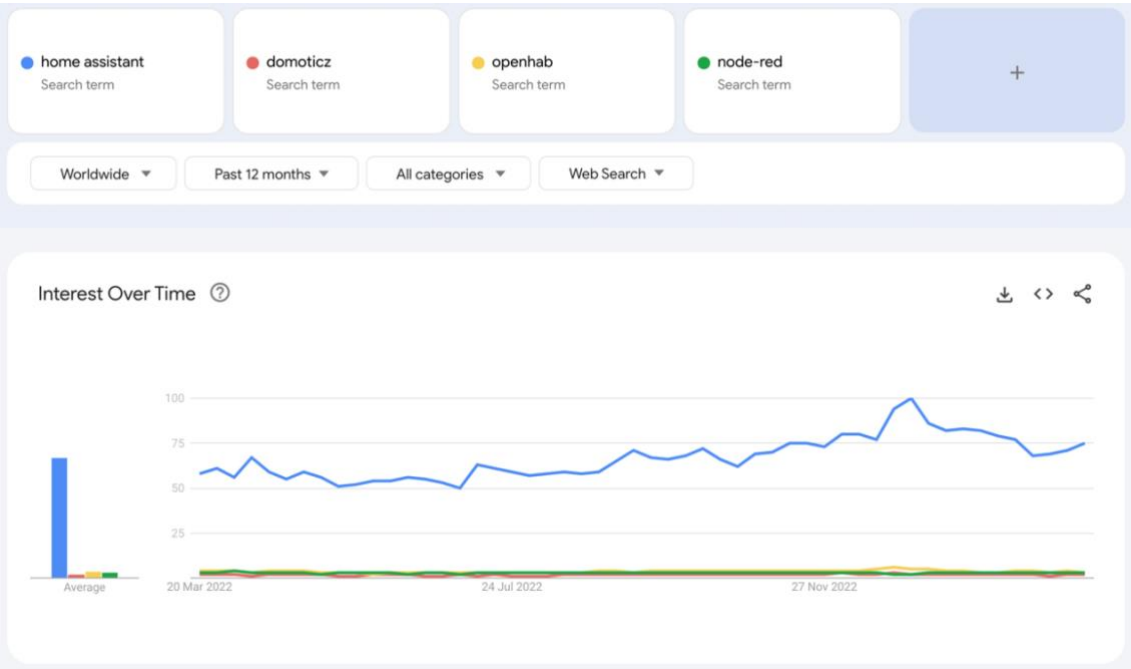
3.1 Koduautomaatika süsteemide valik analüüsiks

Koduautomaatika süsteemide analüüsiks valitakse nii kommertsiaalseid kui kogukonna poolt arendatud platvorme (vt ptk 2.3). Nende populaarsust hinnatakse Google'i trendide graafiku ning kasutajate arvustuste põhjal. Google'i trendid (*Google Trends*) on statistika, mis koondab teemade kaupa kasutajate huvi kolmelt platvormilt – Google'i otsing, Google'i uudised (*Google News*) ja videode jagamise veebiportaal YouTube [14].

Toetudes artiklitele [15] ja [16], mis reastavad parimaid targa kodu süsteeme, on koostatud Google'i trendide graafik, mis kirjeldab kommertsiaalsete süsteemide globaalset veebiotsingute mahtu viimase aasta jooksul. Antud graafik paikneb joonisel 2. Sellelt nähtub, et Google Home omab kindlat liidrikohta teiste süsteemide ees. Toetudes antud graafikule ning arvustustele, võetakse edasises põhjalikuma vaatluse alla Google Home, Amazon Alexa ja Samsung SmartThings. Viimane neist on Google'i trendide graafikul küll veidi vähem populaarsem kui Apple HomeKit, kuid SmartThings on arvustustes kindlal esikohal.

Joonis 2. Google'i trendide graafik kommertsiaalsete koduautomaatika süsteemide populaarsuse kohta [17].

Kogukonna poolt arendatud koduautomaatika süsteemide globaalset veebiotsingute mahtu viimase aasta kohta kirjeldab Google'i trendide graafik joonisel 3. Siin on selge liider Home Assistant. Edasises käsitletakse Home Assistant, openHAB ja Node-RED platvorme.



Joonis 3. Google'i trendide graafik kogukonna poolt arendatud koduautomaatika süsteemide populaarsuse kohta [18].

3.2 Koduautomaatika seadistamise simulatsiooni lähtetingimused

Koduautomaatika süsteemide analüüsiks ning võrdlemiseks püütakse kõikidel platvormidel seadistada üht ja sama etteantud stsenaariumi. Vaadeldakse olukordi, kus süsteemi järgmine olek sõltub ainult ühest tingimusest (liikumisest ruumis), ning samm-sammult üha enamatest sisenditest (lisaks liikumisele valgustugevusest, kellaajast või niiskusest ruumis).

Näidisstsenaariumis automatiseeritakse lambi põlemist vannitoas, kus on suur aken, millest tuleb sisse päevavalgus. Lamp saab põleda erinevate valgustugevustega. Olgu ettekirjutatud automaatika reeglid järgmised:

1. Lamp läheb põlema, kui toas on tuvastatud liikumine.
2. Lamp läheb kustu, kui toas pole viimased viisteist minutit liikumist tuvastatud.
3. Lamp läheb kustu, kui toas on valge.
4. Lamp läheb liikumise peale põlema juhul, kui toas on pime, st kui aknast ei tule piisavalt valgust.
5. Lamp läheb liikumise peale väiksema valgustugevusega põlema juhul, kui kellaaeg on vahemikus 23:00-08:00. Selline öine seadistus on mugav, sest siis ei ärrita ere valgus pimedas liikudes silmi.
6. Lamp põleb ja ei lähe kustu, kui toas pole viimased viisteist minutit liikumist olnud ja niiskustase on väga kõrge. Selline seadistus väldib olukorda, kus duši all käies võib lamp kustu minna, kuna liikumissensor ei tuvasta veeauru tõttu liikumist.

Siinkohal peab täheldama, et kui lambi lülitamiseks on vajalik seadistada kõik reeglid, siis peab automaatika süsteem suutma lahendada vastuolulisi reegleid 1 ja 5 ning 1 ja 3. See tähendab, et peab olema võimalik koostada *if-else* konstruktsioone. Reeglite 1 ja 5 vastuolu seisneb olukorras, kus toas tuvastati öisel kellaajal liikumine, mistõttu lamp läks vastavalt reeglile 5 õrnema valgustugevusega põlema, kuid seejärel rakendub reegel 1, mis paneb toas lambi täisvalgusel põlema. Reeglite 1 ja 3 vastuolulisust kirjeldab järgmine olukord: kui päevasel ajal toas liikuda, läheb tuli vastavalt reeglile 1 põlema, kuid reegel 3 kirjutab selle kohe üle, sest on valge ja lamp läheb kustu. Seejärel läheb tuli jälle liikumise peale põlema ning valguse peale kustu jne, lamp vilgub.

Analüüs viiakse läbi süsteemide dokumentatsiooni, kasutajate arvustuste ja õppevideote põhjal. Google Home süsteem on töö autoril enda nutitelefonis seadistatud, mistõttu seda testitakse osaliselt ka praktiliselt.

3.3 Kommertsiaalsete süsteemide ülevaade ja analüüs

Osutub, et kommertsiaalsetel süsteemidel on oma ülesehituselt palju sarnasusi. Esiteks on nende kõigi ülesseadmiseks vaja vastava tootja mobiilirakendust ning kasutajakontot, seejuures SmartThings puhul sobib Samsungi konto asemel ka Google'i konto. Platvormidega ühilduvate tarkade seadmete hulk kasvab iga aastaga. Nimekiri algab lambipirnidest ja termostaatidest ning jätkub valvekaamerate, televiisorite, suitsuandurite, ukselukkude, kõlarite ja palju muuga. [19]– [20]

Teiseks ühisjooneks kommertsiaalsetel platvormidel on süsteemi juhtimine. Seda on võimalik teha nii mobiilirakenduse kui ka häälkäskluste kaudu. Amazon ja Google on eraldiseisvalt arendanud kaks virtuaalselt tehisintellektil baseeruvat abitehnoloogiat, vastavalt Amazon Alexa ning Google Assistant. Alexa aktiveeritakse häälkäsklusega, Google Assistanti saab lisaks häälkäsklusele aktiveerida kirjaliku juhisega. [19], [20]

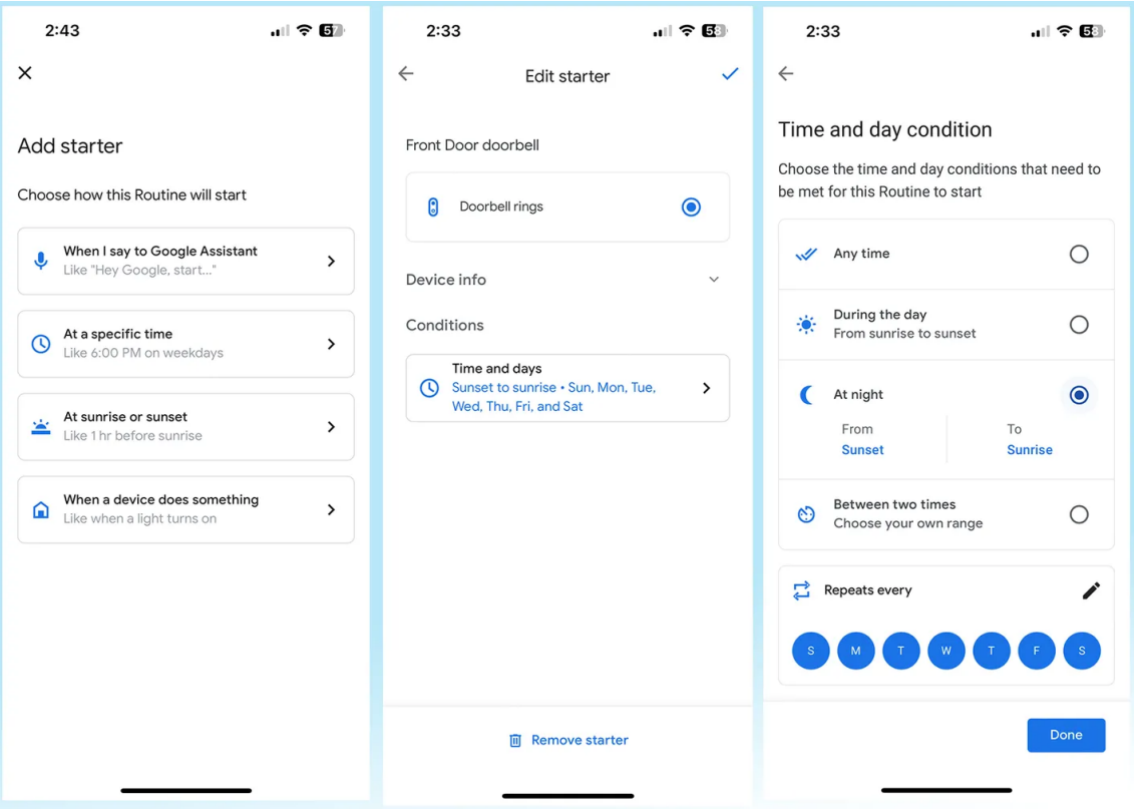
Vimati nimetatud abiteenuseid on võimalik kasutada ka väljapool Amazoni ja Google'i ökosüsteemi. Kujundlikult võib öelda, et tegemist on ajudega, mis saavad olla mistahes ühilduvas kehas. Näiteks Samsungil ei ole eraldiseisvalt arendatud kõneteenust, kuid SmartThings süsteemi on võimalik nii Alexa kui ka Google Assistant ühendada. [21]

Kõigis kolmes vaatluse all olevas süsteemis on lisaks targa kodu juhtimisele võimalik seadistada koduautomaatikat, st automatiseerida tegevusi selliselt, et need ei vaja kasutajapoolset sekkumist. Antud süsteemides nimetatakse automatiseeritud tegevuste jada rutiinideks (*routines*). Rutiin on mitmete tegevuste (*actions*) kogum, mis käivitatakse korraga ühe käsu (*starter*) abil. Rutiini saab käivitada kas häälkäsklusega või automaatselt mingi sündmuse toimumisel. [22]– [23]

3.3.1 Koduautomaatika Google Home platvormil

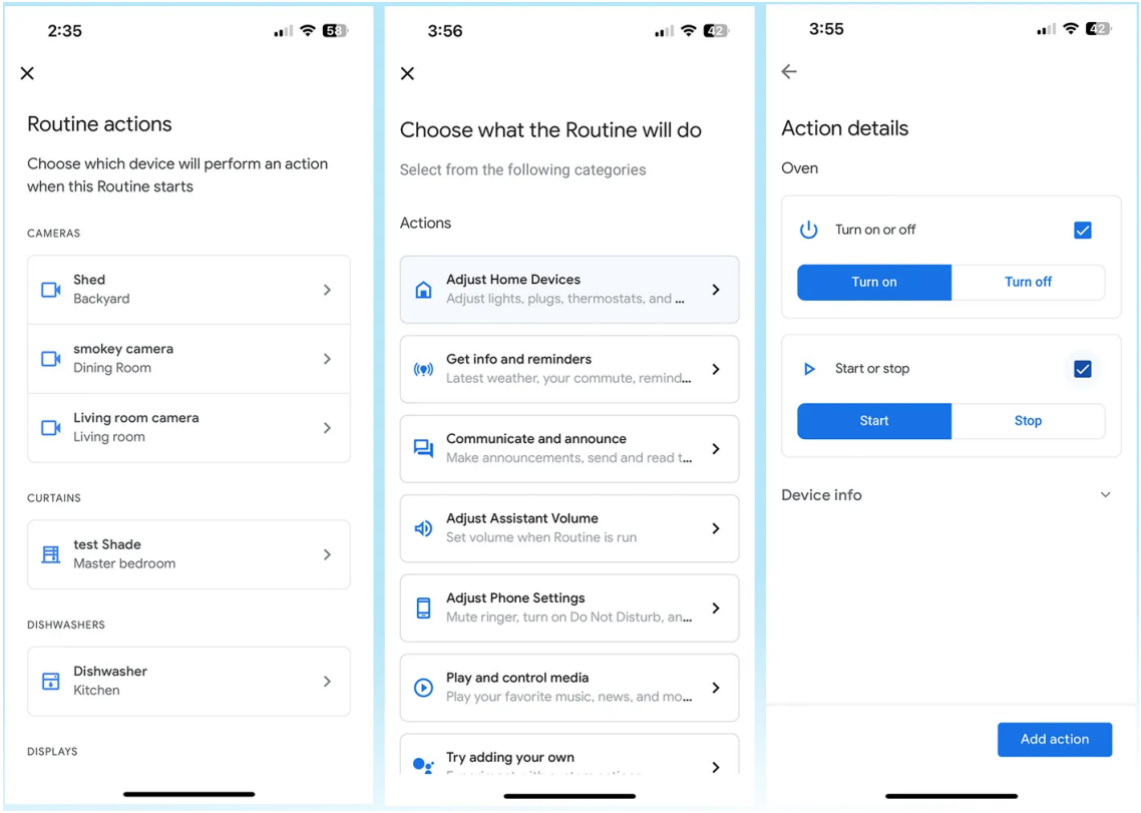
Järgnevalt uuritakse Google Home koduautomaatika seadistamise võimalusi. Joonisel 4 on näidatud sammud, kuidas toimub rutiini käivitamistingimuse seadistamine Google Home rakenduses. Rutiini algatajaks võib olla tark seade, häälkäsklus, kindel kellaaeg,

päikesetõus või -loojang. Seejuures targa seadme kui rutiini algataja tugi lisandus platvormile alles 2022. aasta lõpus. [22], [24]



Joonis 4. Rutiini käivitamise seadistamine Google Home rakenduses [22].

Pärast rutiini algataja valimist saab lisada rutiini tegevused, mida illustreerib joonis 5.



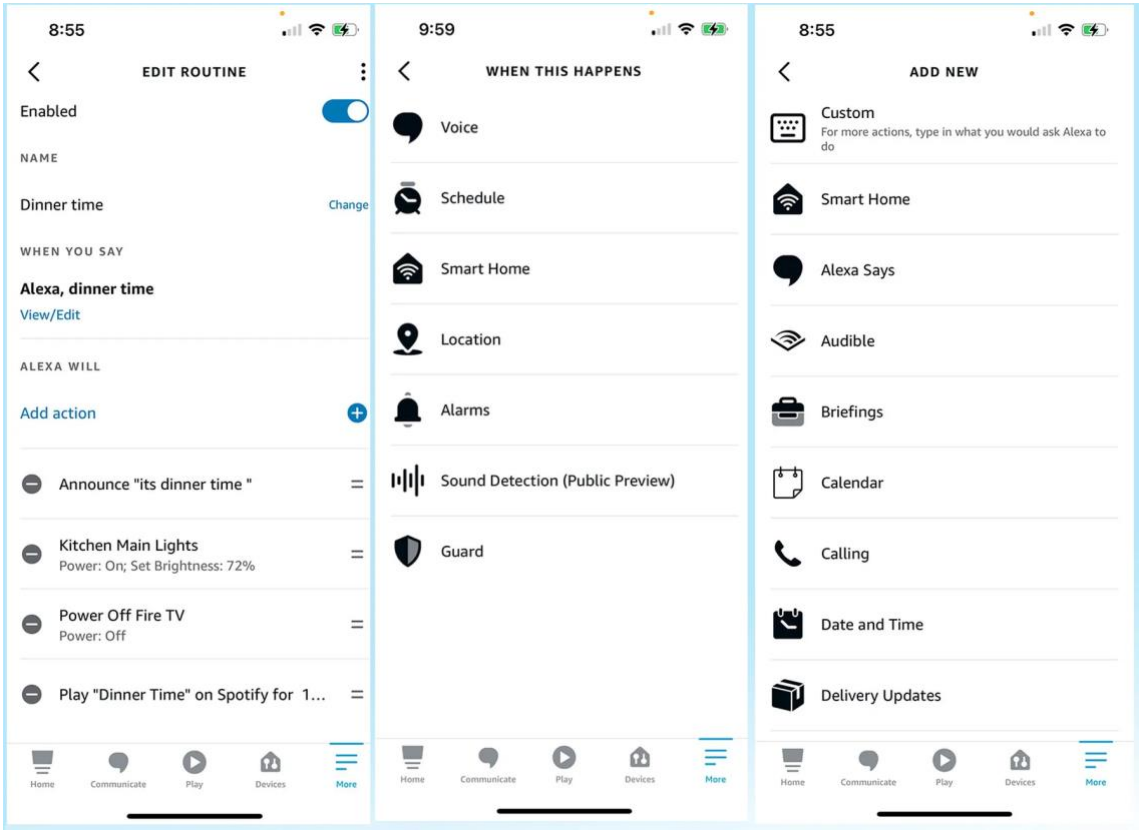
Joonis 5. Rutiini tegevuse seadistamine Google Home rakenduses [22].

Antud infost lähtudes võib öelda, et peatükis 3.2 kirjeldatud näidisreeglite iga üksik tingimus on siin platvormil seadistatav, st rutiinides on võimalik kasutada liikumis-, valgus- ja niiskusanduri infot, kellaaega ning lambi olekut. Toetudes Google'i dokumentatsioonile [25] ning joonisele 4, on näha, et rutiini algatajaks on võimalik määrata targa seadme olek või targa seadme olek koos kellaajaga, kuid mitte mitme targa seadme olek korraga (liikumis-, valgus- ja niiskusandur). Seega ei ole võimalik konfigureerida näidisreegleid 4 ja 6.

Reeglite 1 ja 5 vastuolulisust on võimalik lahendada, kui reeglit 1 täiendada päevase kellaaja lisatingimusega. Reeglite 1 ja 3 vastuolu aga ei ole võimalik lahendada, kuna süsteem ei võimalda rutiini algatajaks määrata kahe targa seadme, liikumis- ja valgusanduri olekut, et neid *and* konstruktsiooniga ühendada. Samuti ei võimalda süsteem *if-else* konstruktsiooni, et tingimusi üksteise järel kontrollida.

3.3.2 Koduautomaatika Amazon Alexa platvormil

Alexa platvormil on rutiini algatajate valik mõnevõrra laiem Google Home poolt pakutavast. Siin süsteemis võib rutiini käivitada häälkäsklus, alarm, etteantud kellaaeg, targa seadme kindel olek, geograafiline asukoht, mingi heli (näiteks beebi nutt, koera haukumine) või Alexa valve (*Alexa guard*) aktiveerumine, s.o suitsu- ja vingugaasi anduri või klaasi purunemise hääle tuvastamine. Rutiini tegevuste valik on väga lai, näiteks tarkade seadmete oleku muutmine, muusika või audioraamatu mängimine jne. Joonisel 6 on kuvatõmmised Alexa rakendusest rutiini seadistamise kohta. Vasakpoolsel pildil on rutiini üldinfo seadistamine, keskmisel pildil on rutiini käivitamisvõimalused ning parempoolsel pildil on valik rutiini võimalikest tegevustest. [26], [27]

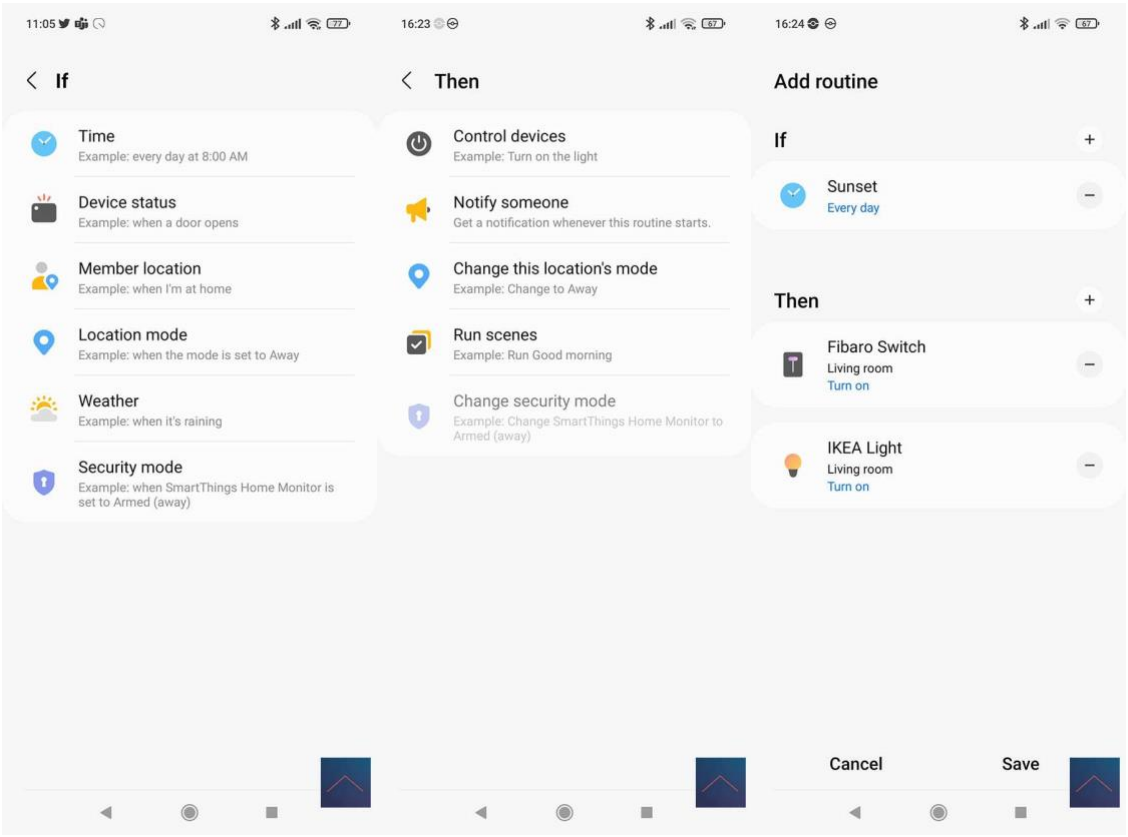


Joonis 6. Rutiini seadistamine Alexa rakenduses [27].

Toetudes joonisele 6 ning allikale [28], osutub, et näidisreeglite konfigureerimise võimalused ja kitsaskohad on siin samad kui Google Home platvormil: reegli iga üksik tingimus on seadistatav; rutiini algatajaks on võimalik määrata targa seadme olek või targa seadme olek koos kellaajaga, kuid mitte mitme targa seadme olek korraga; ei ole võimalik konfigureerida näidisreegleid 4 ja 6; reeglite 1 ja 5 vastuolu on lahendatav, reeglite 1 ja 3 oma mitte (vt lk 24).

3.3.3 Koduautomaatika Samsung SmartThings platvormil

Sammud rutiini seadistamiseks SmartThings rakenduses on näidatud joonisel 7. Seal kasutatakse visuaalselt tingimuslikku *if-this-then-that* loogikat. Esimesel pildil on näha tingimused, mis saavad olla rutiini algatajaks: kellaaeg, targa seadme olek, elaniku asukoht, asukoha profiil, ilm, turvalisuse profiil. Teisel pildil on loetletud rutiini tegevuste võimalused ning viimane pilt kirjeldab lõppreeglit.



Joonis 7. Rutiini seadistamine SmartThings rakenduses [23].

Eelmiste platvormidega võrreldes on SmartThings kõige võimekam, kuna siin on võimalik rutiini algatajaks määrata mitu tingimust, seejuures kõik võivad olla seotud mingi targa seadme olekuga. Seega on võimalik seadistada reegleid 4 ja 6, mis on seotud nii liikumis-, valgus- kui niiskusanduriga. Samas reeglite 1 ja 5 ning 1 ja 3 vastuolu on siingi aktuaalne. [23]

Automaatika reegleid, mida rakenduses seadistada ei saa, on võimalik ise kirjutada Rules API (*Application Programming Interface*) abil. Tegemist on SmartThings API-ga, mis võimaldab reegleid JSON (*JavaScript Object Notation*) formaadis kirjutada ning need SmartThings platvormile salvestada. Salvestatud reeglid ei ole aga hetkel SmartThings

mobiilirakenduses nähtavad, nende kasutamiseks on vaja kolmanda osapoole integratsiooni SmartThings ning Rules API vahel. Kui automaatika reeglid sõltuvad paljudest sisendsignaalidest, siis muutub nende haldamine tülikaks, kuna JSON formaadis kirja pandud loogika läheb pikaks ja raskesti loetavaks. Taoline mitme tingimuslausega reegel on näitena toodud lisas 2. [29]

3.3.4 Kommertsiaalsete süsteemide analüüsi kokkuvõte

Tabel 1 annab ülevaate, millised etteantud näidisreeglid on süsteemides seadistatavad.

Tabel 1. Kommertsiaalsete koduautomaatika süsteemide võimekus erinevate stsenaariumite korral.

	Google Home	**Amazon Alexa**	**Samsung SmartThings**
1. toas liikumine → lamp põlema	+	+	+
2. toas pole viimased viisteist minutit liikumist olnud → lamp kustu	+	+	+
3. toas on valge → lamp kustu	+	+	+
4. toas on liikumine ja pime → lamp põlema	–	–	+
5. toas on liikumine ja kell on vahemikus 23:00–08:00 → lamp põlema väikse valgustugevusega	+	+	+
6. lamp põleb, viimased 15 minutit pole liikumist, niiskustase on kõrge → lamp jääb põlema	–	–	+

Siinkohal tuleb tähele panna, et reegleid on võimalik üksikult seadistada, kuid kui need kõik kombineerida lambi lülitamist juhtima, siis on automaatika vigane. Näiteks on reeglid 1 ja 3 vastuolulised: kui keegi liigub toas, siis tuli hakkab vilkuma, sest reegel 1 reageerib liikumisele ja lülitab lambi sisse, ning reegel 3 reageerib valgusele ja lülitab selle välja. Potentsiaalselt on kogu loogika kirjeldatav SmartThings platvormil Rules API abil, kuid vastav konfiguratsioonifail tuleks pikk, raskesti loetav ning puudub mehhanism, mis lihtsustaks paljude sisendsignaalide haldust automaatika reeglites.

Kokkuvõtvalt võib öelda, et koduautomaatika valdkonnas on kommertsiaalsetel süsteemidel arenguruumi. Need on väga võimekad juhtima tarka kodu, st kasutajapoolse sekkumisega muudavad need kasutaja igapäevaelu lihtsamaks ja mugavamaks.

Koduautomaatika osa, st kasutajapoolse sekkumiseta pool, on aga limiteeritud võimekusega. Süsteemides ei ole praktiliselt võimalik seadistada keerukamaid, mitme lähtetingimusega automaatika reegleid. Selles valdkonnas on SmartThings kõige võimekam. Üldiselt saab reegleid seadistada *if-this-then-that* ja *if-this-and/or that-then-that* loogikalausetega, kuid ei ole võimalik kasutada *if-else* konstruktsiooni ning välistavaid tingimusi *while*, *unless*. Lisaks on kommertsiaalsetel platvormidel puuduseks piiratud lisatavate reeglite hulk jäädes vahemikku 200–250. [30], [31]

3.4 Kogukonna poolt arendatud süsteemide ülevaade ja analüüs

Kogukonna poolt arendatud koduautomaatika platvorm on avatud lähtekoodiga tarkvara, mis toimib tarkade seadmete keskse juhtimissüsteemina. Seda on võimalik installeerida erinevate operatsioonisüsteemide peale. Üldiselt soovitatakse koduautomaatika jaoks kasutada eraldiseisvat riistvara, näiteks Raspberry Pi arvutit, kuna see on soodne ning töötab madala energiakuluga. [32]

Kommertsiaalsete süsteemidega võrreldes on kogukonna poolt arendatud süsteemid lõppkasutaja jaoks tehniliselt väljakutsuvamad. Nende installeerimine ning konfigureerimine nõuab mõningaid teadmisi riistvarast, võrkudest, programmeerimisest. Samas on need spetsialiseerunud justnimelt koduautomaatikale mitte targale kodule, mis tähendab, et need peaksid võimaldama luua keerukaid automaatika reegleid. [32]

Antud platvormid on võimelised ühenduma erinevate tootjate seadmete ja süsteemidega, seejuures ka siin eelnevalt käsitletud kommertsiaalsete süsteemidega. Arvestades kogukonna jätkuvat arendustööd, lisandub platvormidele aina enam laiendusi, mis võimaldavad neid uute süsteemidega integreerida. Platvormidel on olemas ka graafilised kasutajaliidesed, mille kaudu on võimalik süsteemi seadistada, juhtida ning jälgida. [33]–[34]

3.4.1 Süsteem Home Assistant

Antud süsteemil on olemas mobiilirakendus nii Android kui iOS operatsioonisüsteemi jaoks, mille abil on võimalik juhtida arvutisse installeeritud Home Assistanti instantsi [35].

Automaatika reeglid (*automations*) Home Assistant platvormil koosnevad kolmest osast [36]:

1. triger (*trigger*) – kirjeldab sündmuse, mis käivitab automaatika reegli kontrolli, näiteks „kui Jaak jõuab koju“;
2. tingimus (*condition*) – valikuline lisatingimus limiteerimaks automaatika reegli käivitumist, näiteks „päike on loojunud“;
3. tegevus (*action*) – tegevus, mis käivitub, kui kõik eeltingimused on täidetud, näiteks „pane tuled elutoas põlema“.

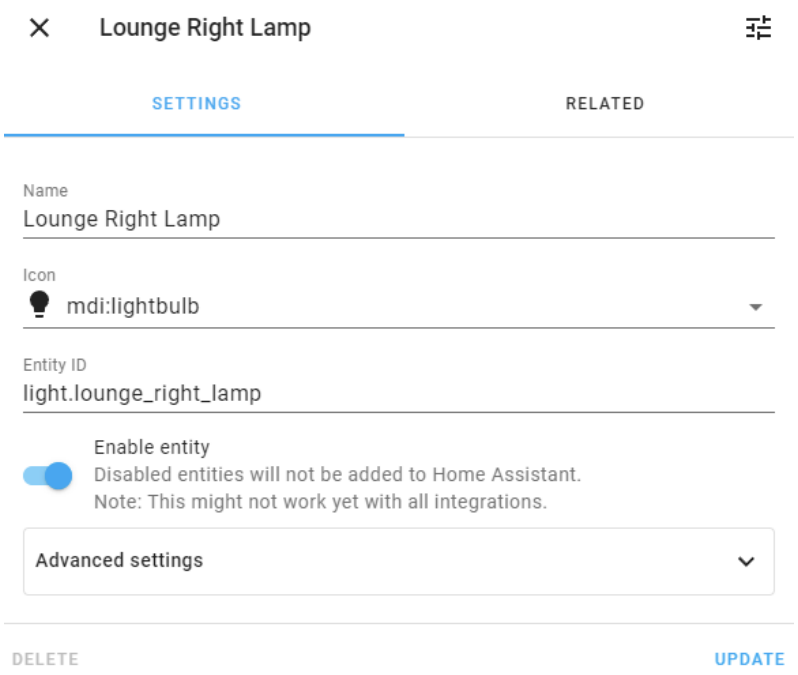
Trigerite ja tingimuste erinevus on see, et trigerid käsitlevad tegevusi, ning tingimused süsteemi olekut. Vastav kontseptsioon on näiteks „pane tuli põlema“ vs „tuli põleb“. [36]

Home Assistant platvorm hoiab kogu süsteemi olekut ühes kohas, mida nimetatakse olekumasinaks (*state machine*). Seal on info olemite hetkeseisu kohta, mida teised olemid, mallid ja graafiline kasutajaliides saavad küsida ja kasutada. Olem on üks osa automaatika süsteemist, see võib olla lamp, lüliti, inimene, isegi päike. Olemi oleku muutuseid saab kasutada automaatika trigeritena ning olemi hetkeseisu saab kasutada tingimustes. [36]

Automaatika reegleid on võimalik platvormil seadistada nii graafilise kasutajaliidese kaudu kui ka konfiguratsiooni failides, mis on kirjutatud YAML (*YAML Ain't Markup Language*) formaadis [37]. Näiteks on joonisel 8 kirjutatud YAML formaadis ühe lambi konfiguratsioon, mille seadistus graafilises kasutajaliideses on kujutatud joonisel 9.

```yaml
homeassistant:
  name: Home
  unit_system: metric
customize:
  light.lounge_right_lamp:
    friendly_name: Lounge Right Lamp
    icon: mdi:lightbulb
```

Joonis 8. Seadme info YAML formaadis Home Assistant platvormil.



Joonis 9. Seadme info Home Assistant graafilises kasutajaliideses [38].

Lähtudes Home Assistant dokumentatsioonis olevast infost, et automaatika reeglitel võib olla mitu trigerit ja tingimust, ning uurides, milliseid tingimusi tehniliselt üldse kirja saab panna, võib väita, et kõik süsteemi analüüsiks etteantud näidisreeglid on siin platvormil teostatavad [39]. See tähendab, et on võimalik defineerida järgmisi tingimusi: liikumine toas, lambi põlemine, viimased viisteist minutit pole liikumist olnud, toas on pime, kellaaja vahemik, lambi valgustugevus, niiskustase. Samuti on võimalik stsenaariume korraga seadistada, sest YAML formaat toetab kõikvõimalikke loogikakonstruktsioone. Seega peaks olema võimalik lahendada reeglite 1 ja 5 ning 1 ja 3 vastuolud.

YAML formaat hoiab suhteliselt selget ja loetavat struktuuri keerukamate reeglite puhul. Näiteks on joonisel 10 automaatika reegel kahe sisendsignaaliga: Paulus on kodus ja temperatuur on alla 20 kraadi. Teisalt võib konfiguratsiooni loogika olla oluliselt pikem nagu on näidis lisas 3, kus seadistatakse magamistoa lambilülitit.

```
- alias: „Check if Paulus is home AND temperature is below 20“
  condition:
    - condition: state
      entity_id: „device_tracker.paulus“
      state: „home“
    - condition: numeric_state
      entity_id: „sensor.temperature“
      below: 20
```

Joonis 10. Tingimuste defineerimine automaatika reeglis [40].

Kokkuvõtvalt on Home Assistant võimekas automaatika platvorm, kus paljude sisendsignaalide haldamiseks tuleb need YAML formaadis korrektselt defineerida. Võimalikes pikkades failides võib korra loomisel abi olla skriptidest. Need on seadistuse plokid, mida saab konfiguratsioonides käivitada ja taaskasutada. [41]

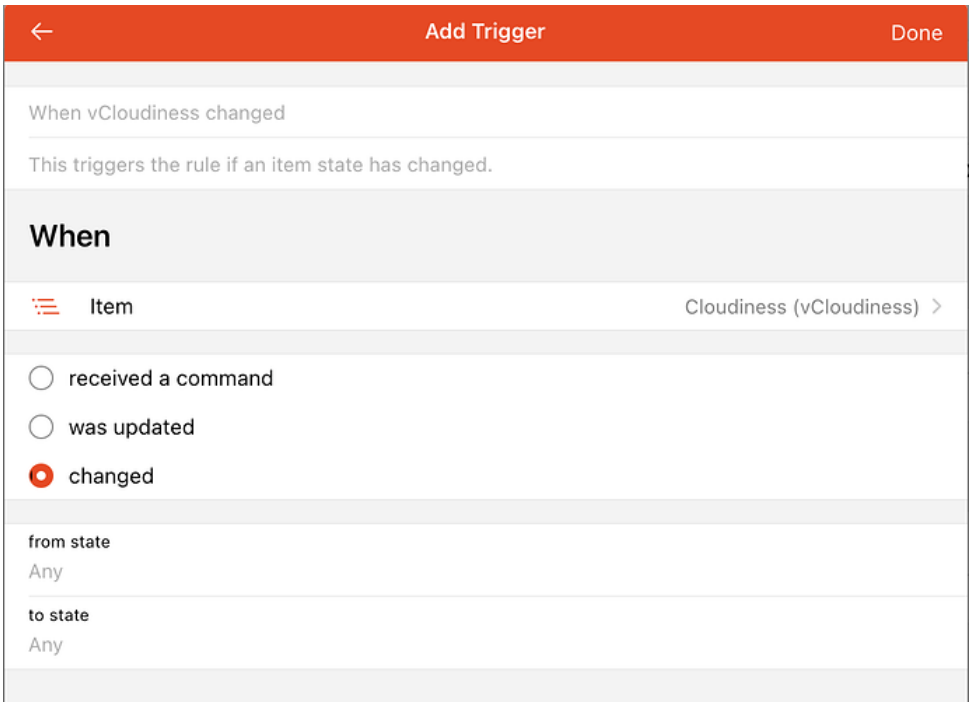
3.4.2 Süsteem openHAB

OpenHAB on akronüüm sõnadest *Open Home Automation Bus*. Antud süsteemil on olemas openHAB mobiilirakendus nii Android kui ka iOS operatsioonisüsteemi jaoks, mille abil on võimalik juhtida arvutisse installeeritud openHAB instantsi. [35]

Dokumentatsiooni uurides jääb esimesena silma süsteemis kasutatavate mõistete rohkus. See tuleneb asjaolust, et openHAB teeb vahet füüsilisel, seadmetega seotud kihil, ning virtuaalsel, platvormiga seotud kihil. Süsteemi lisatud seadmetega töötamisel eristatakse kolme mõistet:

- Asjad (*Things*) – Olemid, mida saab füüsiliselt süsteemi ühendada. Need võivad olla nii seadmed, võrguteenused kui mistahes muud informatsiooni ja funktsionaalsuse allikad. [42]
- Kanalid (*Channels*) – Ühendused Asjade ning Elementide vahel. Kanal pärineb Asjast ning see kirjeldab funktsionaalsust, mis Asjal on. Kui Asi on füüsiline olem, siis Kanal on Asja konkreetne funktsionaalsus. Näiteks elektripirn on Asi ning värv ja värvi temperatuur on selle kaks Kanalit. [42]
- Elemendid (*Items*) – Kirjeldavad funktsionaalsust, mida süsteemis kasutatakse. Need hoiavad infot seadmete kohta, mis on nendeni jõudnud Kanalite kaudu. Elementidel on olek. Hetkel on süsteemis saadaval kaksteist Elemendi tüüpi, sinna kuuluvad näiteks värv (*Color*), lüliti (*Switch*), number (*Number*). [43]

Automaatika konfiguratsioone nimetatakse siin reegliteks (*rules*) ning need koosnevad samadest osadest, mis Home Assistant platvormil: trigerid, tingimused ja tegevused (vt ptk 3.4.1). Graafilises kasutajaliideses on võimalik seadistada ainult *if-this-then-that* loogikat. Kõik graafiliselt lisatud reeglid salvestatakse süsteemi YAML formaadis. Joonisel 11 on kujutatud ühe reegli trigeri seadistamine, mis jälgib ilmateenusest tulenevat infot pilvisuse kohta. Selle reegli täielik konfiguratsioon YAML formaadis on toodud joonisel 12. [44]



Joonis 11. Automaatika reegli trigeri seadistamine openHAB platvormil [44].

Joonis 12. Automaatika reegel YAML formaadis openHAB platvormil [44].

OpenHAB toetab paljusid programmeerimiskeeli, milles on võimalik automaatika reegleid failidesse kirjutada. Vaikimisi on platvormiga kaasas Blockly, Rules DSL (*Domain-Specific Language*) ja ECMAScript tugi, teiste keelte tugi on saadaval laiendite näol. Joonisel 13 on näitena toodud ECMAScript 2021 keeles kirjutatud tingimus, et kellaaeg on vahemikus päikeseloojang kuni 23:00. [45]

```
var sunset = time.ZonedDateTime.parse(items.getItem('Sunset').state);
var endTime = time.ZonedDateTime.now()
              .withHour(23).withMinute(0).withSecond(0).withNano(0);
now.isAfter(sunset) && now.isBefore(endTime);
```

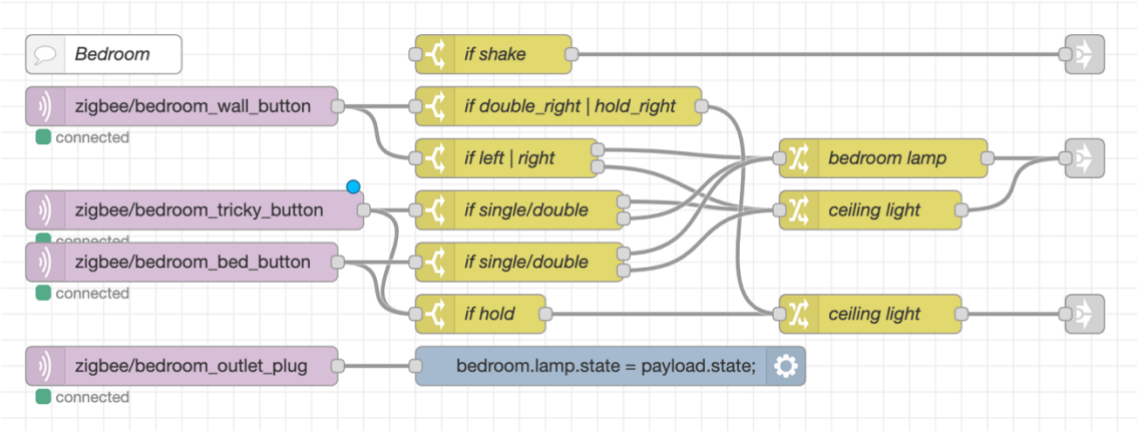
Joonis 13. Kellaaja tingimus ECMAScript keeles openHAB platvormil [45].

Lähtudes openHAB dokumentatsioonis olevast infost, võib väita, et kõik süsteemi analüüsiks etteantud näidisreeglid on siin platvormil teostatavad. Automaatika reeglitel võib olla mitu trigerit ja tingimust ning on võimalik defineerida järgmisi seadistusi: liikumine toas, lambi põlemine, viimased viisteist minutit pole liikumist olnud, toas on pime, kellaaja vahemik, lambi valgustugevus, niiskustaseme tõus. Samuti on võimalik koodis lahendada reeglite 1 ja 5 ning 1 ja 3 vastuolu. Mingeid erilisi võtteid paljude sisendsignaalide halduseks platvorm ei paku, neid peab haldama tingimuslausetega vastavalt valitud programmeerimiskeele võimalustele.

3.4.3 Süsteem Node-RED

Node-RED on avatud lähtekoodiga graafiline arendustööriist, mis võimaldab omavahel ühendada riistvara, API-sid ning võrguteenuseid. Algselt arendas IBM antud platvormi firmasiseseks kasutuseks, kuid 2016. aastal muudeti see vabavaraks. Node-RED platvormi saab kasutada mistahes automaatika liinides, tegemist ei ole spetsiaalselt koduautomaatika jaoks arendatud tööriistaga. [46], [47]

Node-RED brauseripõhises tööriistas on võimalik luua vooge (*flow*) tegevuste automatiseerimiseks. Voog ehitatakse üles sõlmede (*node*) abil. Hetkel on saadaval üle 4000 sõlme ning neid arendatakse kogukonna poolt pidevalt juurde. Iga sõlm võib olla teiste sõlmedega ühendatud ning igaüks täidab mingit konkreetset ülesannet – kogub, töötleb või saadab andmeid. Sõlmed võimaldavad täita väga keerukaid ülesandeid, näiteks saata andmeid kasutades MQTT protokolli, või avaldada andmeid veebiteenustes nagu Dropbox ja Google Drive. [46], [47] Joonisel 14 on näide Node-RED brauseripõhisest tööriistast, milles on sõlmede abil ehitatud erinevad vood.



Joonis 14. Node-RED platvormil koostatud voog tegevuste automatiseerimiseks.

Node-REDi käitusaeg (*runtime*) on ehitatud Node.js süsteemi peale, mis on sündmusjuhitava arhitektuuriga JavaScripti käitussüsteem [46].

Node-REDis ehitatud rakenduse voo paneb käima mingi sündmus, mis tuleb üldjuhul väljastpoolt rakendust. Selleks võib olla näiteks riistvara oleku muutus või kindla etteantud aja möödumine. Antud sündmusest tulenev info saadetakse mööda sõlmesid edasi sõnumi (*message*) kujul. [47]

Kuigi Node-RED platvorm ei ole spetsiaalselt koduautomaatika jaoks arendatud, on selles olemas vajalikud komponendid koduseadmete juhtimiseks. Kogukonna poolt on arendatud mitmeid koduautomaatika spetsiifilisi laiendusi. Näiteks on võimalik ühendada süsteemi eelnevalt käsitletud targa kodu platvormid Google Home, Alexa, SmartThings, juhtida Roborock robottolmuimejat, Sonose kõlareid jpm. [34]

Kõik etteantud näidisreeglid on Node-RED platvormil teostatavad. Tegemist on väga võimeka süsteemiga, milles on võimalik lahendada reeglite 1 ja 3 ning 1 ja 5 vastuolud. Seejuures on aga oluliseks miinuseks asjaolu, et paljude sisendsignaalide korral lähevad vood väga keeruliseks: kui sõlmi on voos palju, siis nendevaheliste seoste rägastiku haldamine nõuab palju tähelepanu ning on veaohtlik.

3.4.4 Kogukonna poolt arendatud platvormide analüüsi kokkuvõte

Kogukonna poolt arendatud koduautomaatika süsteemides on võimalik koostada keerukaid mitmetest erinevatest sisendsignaalidest sõltuvaid automaatika reegleid. Tabel 2 kirjeldab, et kõik näidisreeglid on antud platvormidel teostatavad.

Tabel 2. Kogukonna arendatud koduautomaatika süsteemide võimekus erinevate stsenaariumite korral.

	Home Assistant	**OpenHAB**	**Node-RED**
1. toas liikumine → lamp põlema	+	+	+
2. toas pole viimased viisteist minutit liikumist olnud → lamp kustu	+	+	+
3. toas on valge → lamp kustu	+	+	+
4. toas on liikumine ja pime → lamp põlema	+	+	+
5. toas on liikumine ja kell on vahemikus 23:00–08:00 → lamp põlema väikse valgustugevusega	+	+	+
6. lamp põleb, viimased 15 minutit pole liikumist, niiskustase on kõrge → lamp jääb põlema	+	+	+

Küll aga ei järeldu süsteemide dokumentatsioonidest, et paljude sisendsignaalide haldus oleks olulisel määral lihtsustatud. Rohkete sündmuste baasil koostatud automaatika reeglite puhul tuleb kasutada keerukate konstruktsioonidega tingimuslauseid, jagada sisendinfo loogiliselt ära trigerite ning tingimuste vahel. Iga sisendsignaali lisandumisega muutuvad reeglid keerukamaks, seetõttu ka raskemini loetavaks ning hallatavaks.

4 Süsteemi oleku haldamise sõlme arendus Node-RED platvormile

Koduautomaatika platvormide ülevaatest ja analüüsist peatükis 3 nähtus, et paljude sisendsignaalide haldus tuleb automaatika reeglites lahendada tingimuslausete abil. Need aga muutuvad tingimuste lisandumisel üsna kiirelt mahukaks ning keerukaks, mis omakorda muudavad automaatika vood veaohtlikuks.

Antud peatükk käsitleb paljude sisendsignaalide haldamise teemat Node-RED platvormil. Praktilise töö eesmärk on arendada sõlm, mis aitab lihtsustada süsteemi järgmise oleku otsustamise loogikat.

4.1 Lahenduse arhitektuuriline idee

Lahenduse arhitektuuriline idee pärineb eesrakenduse teegist React. Selle teegi abil ehitatud kasutajaliides toimib järgnevalt:

- kasutajaliidese kuvamine sõltub mingitest parameetritest, näiteks kasutajaliideses kuvatav nupp on roheline, kui sellele pole vajutatud, ja hall, kui sellele on vajutatud;
- kasutajaliides värskendab oma vaadet automaatselt, kui on toimunud mingi muudatus andmetes, mida antud vaate jaoks vaja on, näiteks info kasutajapoolse nupuvajutuse kohta.

Analoogiliselt võib mõelda koduautomaatikast – see on üks eesrakendus, mille komponendid on lambid, radiaatorid, kõlarid jne, ning mille kuvamine sõltub andmetest, näiteks liikumisest, valgusest, kellaajast. Vaade võiks end automaatselt värskendada juhul, kui andmetes on toimunud muudatus, vastupidiselt sellele, et iga vaates olev komponent peaks iga kuvamiseks vajamineva info üle ise järge pidama. Viimane stsenaarium on analoogiline eesrakenduse teegi jQuery loogikaga.

4.2 Funktsionaalsed nõuded

Selleks, et koduautomaatika seadmed oleksid võimelised oma olekut vastavalt uuele arhitektuurile toetudes muutma, peab uus arendatav Node-RED laiendus võimaldama jälgida automaatika süsteemi olekut ning vajadusel reageerida selle muutustele. Nende eesmärkide saavutamiseks on tarvilikud järgmised funktsionaalsed nõuded laiendusele:

1. Süsteem peab oskama lugeda infot Node-RED globaalsest kontekstist.
2. Süsteem peab oskama aru saada, kas mingi väärtus kontekstis muutus.
3. Kui mingi väärtus, mida automaatika voos kasutatakse, kontekstis muutus, peab süsteem oma oleku uuesti arvutama.

4.3 Arenduskeskkond

Arenduse jaoks on tarvilik Node-RED tarkvara, mis installeeriti kohalikku arvutisse Dockeri konteinerisse. Joonisel 15 on toodud Dockeri konteineri konfiguratsiooni faili *docker-compose.yml* sisu, mida on võimalik käivitada käsurealt käsuga *docker compose up*. Seejärel on Node-RED graafiline kasutajaliides saadaval aadressil http://127.0.0.1:1880. Arendusprojekt kannab nime *node-red-contrib-context-hook*.

```
services:
  thesis-node-red:
    container_name: thesis-node-red
    image: nodered/node-red:latest
    restart: always
    environment:
      - TZ=Europe/Tallinn
    ports:
      - "1880:1880"
    volumes:
      - ./data:/data:rw
      - ./node-red-contrib-context-hook:/node-red-contrib-context-hook
```

Joonis 15. Dockeri konfiguratsiooni fail Node-RED sõlmede arenduseks.

Node-RED platvorm dikteerib programmeerimiskeele valiku. Kuna selle käitusaeg on ehitatud Node.js süsteemi peale, mis on JavaScripti käitussüsteem, siis toimub arendus JavaScript keeles.

4.4 Lahenduse tehniline kirjeldus

Peatükis 4.1 kirjeldati, kuidas lahenduse arhitektuuriline idee pärineb eesrakenduse teegist React. Täpsemalt on seal kasutusel funktsioon nimega *useState*, mis võimaldab komponendi piires hoida infot komponendi oleku (*state*) kohta. See tähendab, kui komponendi kuvamine sõltub mingist muutuja väärtusest, siis seda väärtust jälgitakse, ning kui see muutub, siis komponendi kuva värskendatakse vastavalt.

Node-RED süsteemis kasutatakse info salvestamiseks konteksti (*context*), millel võib olla kolm erinevat ligipääsupiirangut [48]:

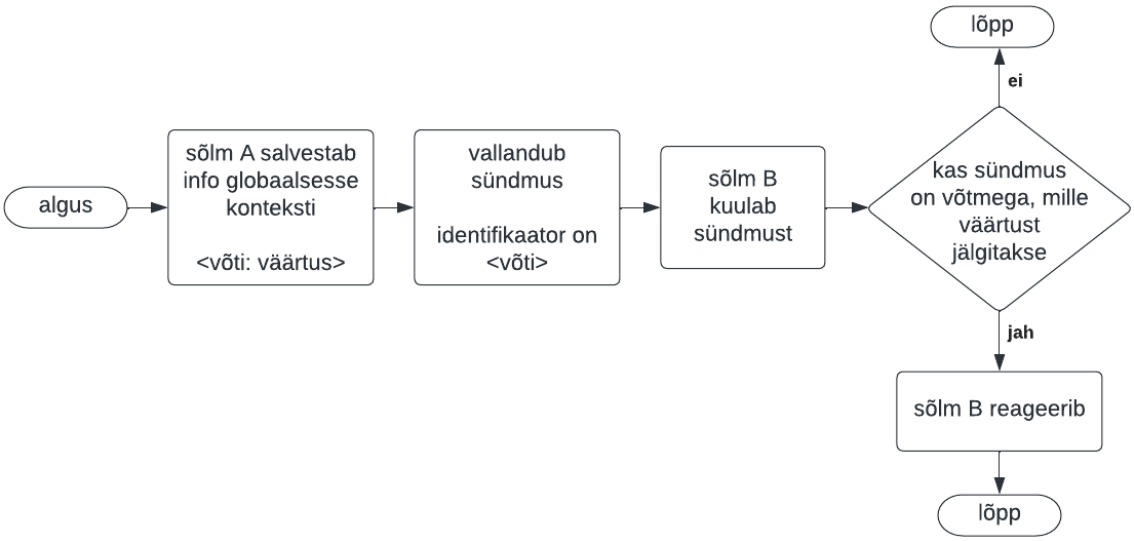
- info on nähtav ainult selle sõlme sees, kus see salvestati;
- info on nähtav kõikidele samasse voogu kuuluvatele või samas redigeerimisaknas paiknevatele sõlmedele;
- info on nähtav kõikidele sõlmedele.

Kontekstis on info sõnastiku kujul. Loodud lahenduses kasutatakse kõige laiema ligipääsuga konteksti, mida nimetatakse globaalseks kontekstiks.

Tegevuste jada selleks, et automaatika voos oleks võimalik seadmete ehk täiturite olekute jaoks olulisi väärtusi salvestada ning nende muutustele reageerida, on järgmine:

1. sõlm A kirjutab info globaalsesse konteksti võti-väärtus kujul;
2. see vallandab süsteemis sündmuse, mille identifikaatoriks on salvestatud võti;
3. sõlm B kuulab sündmuseid;
4. kui sündmus on sõlme B jaoks huvipakkuva võtmega, siis sõlm reageerib, vastasel juhul ei tee midagi.

Kirjeldatud tegevuste voog on kujutatud joonisel 16.



Joonis 16. Lahenduse voodiagramm.

4.5 Sõlmede arendus

Sõlm Node-RED platvormil koosneb kahest failist [49]:

1. JavaScript fail, mis kirjeldab sõlme funktsionaalsust;
2. HTML (*Hyper Text Markup Language*) fail, mis defineerib sõlme omadused, selle redigeerimise dialoogiakna kujunduse ning kasutajale kuvatava abiteksti.

Arenduseks loodi projekt nimega *node-red-contrib-context-hook*, mis on avalikult kättesaadav aadressil https://github.com/siirimangus/node-red-contrib-context-hook. Nimi järgib tava, et kogukonna poolt arendatud laiendused Node-RED platvormile algavad enamjaolt sõnadega *node-red-contrib*.

Arendatud laiendus sisaldab kolme sõlme:

1. Sõlm nimega *set-global-state* – Sõlm, mis esiteks salvestab info globaalsesse konteksti ning seejärel vallandab süsteemi sündmuse, mille identifikaatoriks on salvestatud info võti. Sündmused on Node.js arhitektuuri alustalad. Node.js-i sisseehitatud moodul *events* sisaldab spetsiaalset objekti (*emitter*), mis vallandab sündmuseid, mille peale funktsioonid (*listeners*) reageerivad [50]. Joonisel 17 on kirjeldatud sõlme põhitegevused.

```
const EventEmitter = require('events');
global.set(property, value);
new EventEmitter.emit(property, { previousValue, value });
```

Joonis 17. Sõlme *set-global-state* põhiline funktsionaalsus – info salvestamine ja sündmuse vallandamine.

2. Sõlm nimega *subscribe-state* – Tegemist on abisõlmega juhuks, kui on tarvilik saada teavitusi süsteemi oleku muutuste kohta. Sõlm kuulab talle huvipakkuvaid sündmuseid, mida sõlm *set-global-state* on vallandanud, ning saadab info edasi järgmisse sõlme. Kirjeldatud funktsionaalsust illustreerib joonis 18.

```
const EventEmitter = require('events');
new EventEmitter.on(property, ({ previousValue, value }) => {
  node.send({
    property,
    previousValue,
    value,
    payload: value
  });
});
```

Joonis 18. Sõlme *subscribe-state* funktsionaalsus – sündmuste kuulamine ja info edastamine.

3. Sõlm nimega *state-hook* – Sõlm, mis kasutab kontekstis olevat infot otsustamaks süsteemi järgmise oleku üle, seejuures, kui mingi jälgitav väärtus kontekstis muutub, käivitatakse kogu järgmise oleku arvutuskäik automaatselt uuesti. Kirjeldatud funktsionaalsuse defineerib funktsioon *useGlobal* (sarnaselt Reactis oleva funktsiooniga *useState*), mis on kujutatud joonisel 19.

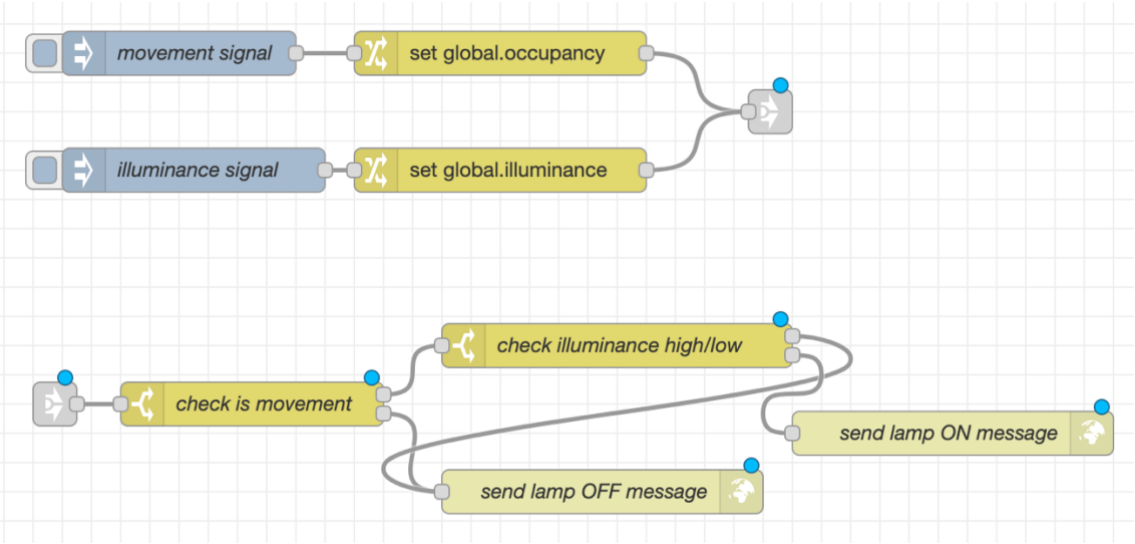
```
const EventEmitter = require('events');

const useGlobal = (property, defaultValue = null) => {
  if (!watchedValues.includes(property)) {
    watchedValues.push(property);
    new EventEmitter.on(property, () => {
      runCode();
    });
  }

  let stateValue = global.get(property);
  if (undefined === stateValue || null === stateValue) {
    stateValue = defaultValue;
  }

  return stateValue;
};
```

Joonis 19. Sõlmes *state-hook* defineeritud funktsioon *useGlobal*.

4.6 Oleku halduse sõlme kasutamine

Eelmise alapeatüki lõpus kirja pandud kolmest sõlmest kõige keerukama ning väärtuslikuma sisuga on sõlm nimega *state-hook*. See on sõlm, mis võimaldab automaatika vooge oluliselt lihtsustada. Selle asemel, et voog üles ehitada paljude tingimuskonstruktsioonide sõlmedega, piisab ühest sõlmest, mis jälgib süsteemi olekut ning vajadusel reageerib muutustele. See tähendab, et süsteemi järgmine olek otsustatakse ühes keskses kohas, graafiliselt ei ole tarvilik ehitada paljudest sõlmedest koosnevat raskesti hallatavat rägastikku.

Olgu näitena vaatluse all peatükis 3.2 kirja pandud reegel number kolm, st toas läheb tuli põlema, kui seal liigutakse ja on pime. Vaatleme kõigepealt, kuidas seadistada seda automaatika voogu ilma uut laiendust kasutamata. Esmapilgul korrekstsena tunduv lahendus on kujutatud joonisel 20.



Joonis 20. Vigane voog otsustamaks lambi põlemise oleku üle liikumise ja valguse infole toetudes.

Osutub aga, et voog on vigane. Tuleb täheldada, et signaalid võivad süsteemi tulla ajalise erinevusega. Siin on võimalik järgmine mittesoovitud stsenaarium:

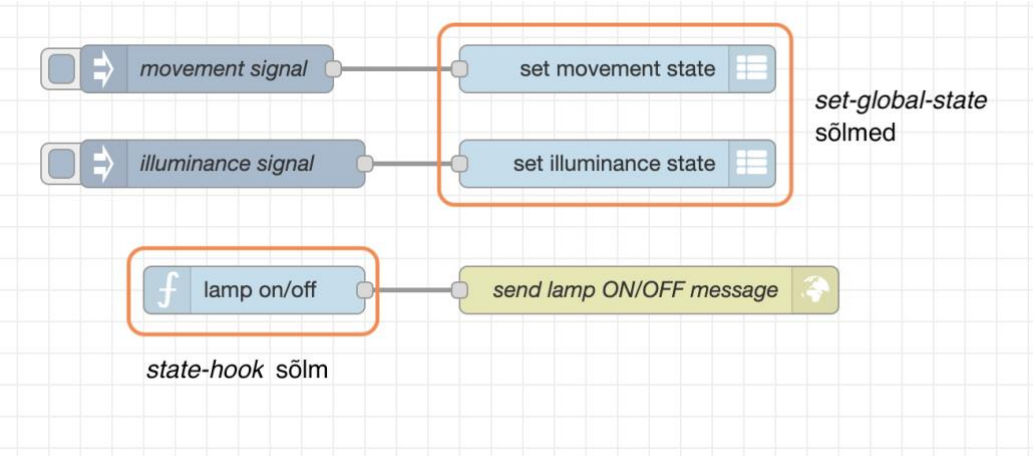
1. on liikumine – lamp läheb põlema;
2. on pime – lamp läheb põlema, st lamp jääb põlema;
3. lambi põlemise tõttu on toas valge – lamp läheb kustu;
4. on pime – lamp läheb põlema jne.

See tähendab, et lamp hakkab toas vilkuma.

Järgnevalt seadistatakse sama automaatika voog uue laienduse abil. Selleks tuleb:

1. liikumise ja valguse kohta sissetulev info salvestada *set-global-state* sõlme abil globaalsesse konteksti;
2. *state-hook* sõlmes kuulata muutusi kontekstis ning otsustada süsteemi järgmise oleku üle.

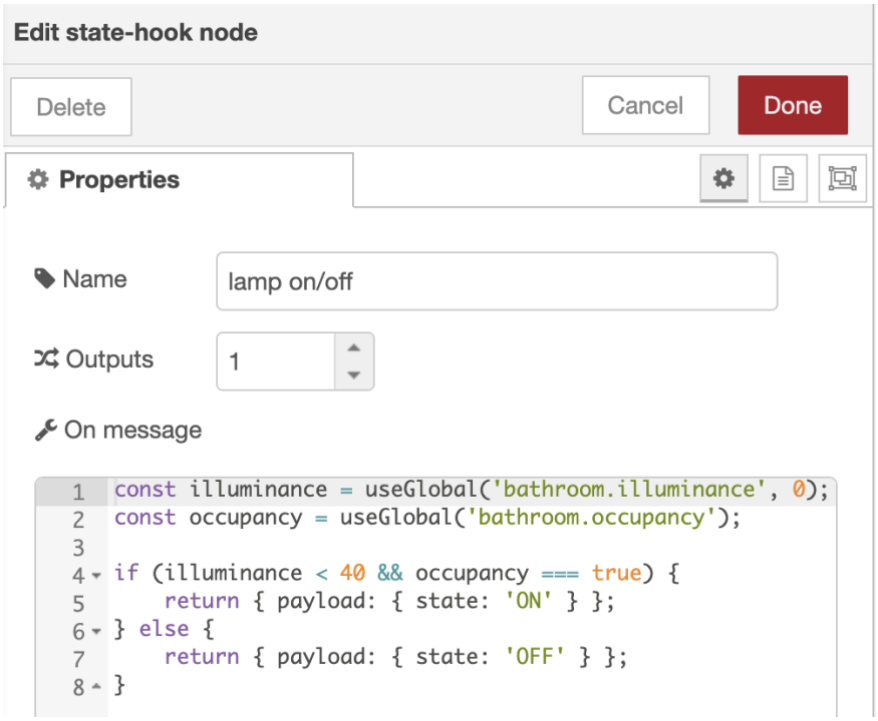
Joonisel 21 on selle voo graafiline esitus.



Joonis 21. Voog kasutades sõlmi uuest arendatud laiendusest.

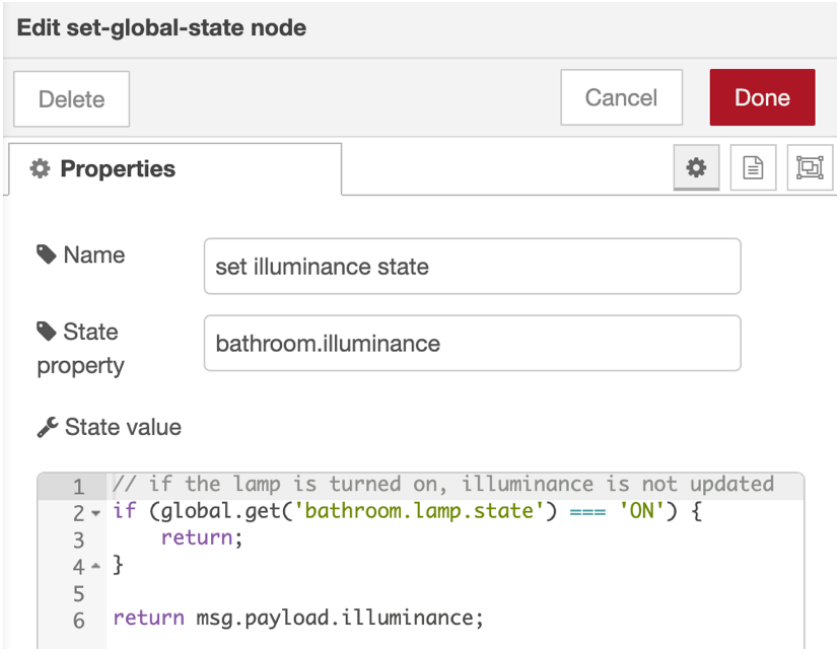
Joonisel 22 on *state-hook* sõlme konfiguratsioon, milles on kogu loogika lambi põlemise kohta. Muutusi globaalses kontekstis jälgib funktsioon *useGlobal*. Kui kumbki väärtustest *bathroom.illuminance* või *bathroom.occupancy* globaalses kontekstis muutub, arvutatakse kogu kirja pandud funktsiooni sisu uuesti. Kui kummaski väärtuses muutusi ei toimu, siis funktsioon ei käivitu ning süsteem säilitab oma viimase oleku.

Võrreldes jooniseid 20 ning 21, on ilmne, et uue laiendusega voog joonisel 21 on lihtsam ning selgem. Seal on kogu tingimuslausete loogika koondunud ühte sõlme, kus seda on koodis kergem hallata võrrelduna graafiliselt mitme sõlme vahel seoste loomisega.



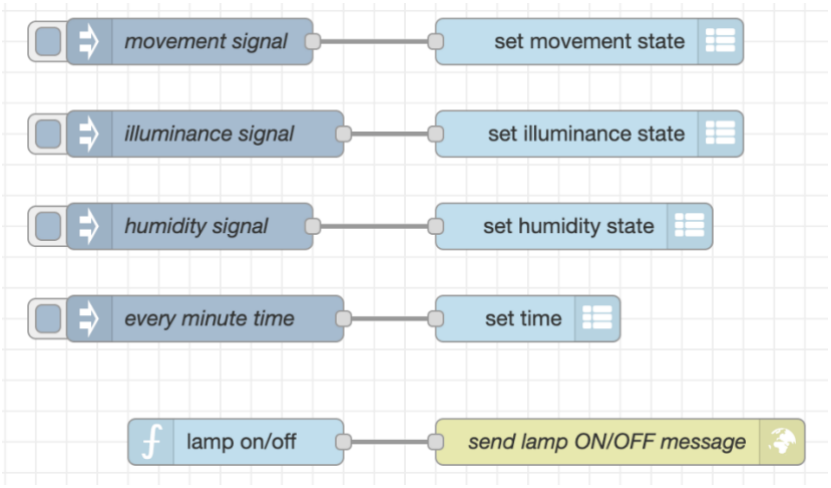
Joonis 22. Lambi oleku seadistamine globaalsele kontekstile toetudes.

Selleks, et joonisel 22 kirja pandud funktsioon ei põhjustaks varasemalt kirjeldatud ebasoovitud lambi vilkumist, tuleb valguse oleku uuendamisel selle olukorraga arvestada. Piisab, kui valguse olekut uuendatakse ainult juhul, kui lamp ei põle. Seega, joonisel 21 oleva sõlme *set illuminance state* loogika peaks olema selline nagu on kirjeldatud joonisel 23.



Joonis 23. Valgusanduri signaali uuendamine globaalses kontekstis.

Lisades lambi põlemise voogu näidisreeglitest veel kellaaja ning niiskustaseme tingimuse, muutub joonisel 20 kujutatud visuaal oluliselt keerukamaks. Samal ajal uut lahendust kasutades lisandub joonisele 21 kaks signaali, mis salvestatakse globaalsesse konteksti, kuid lambi põlemise otsustamise koht sõlmi juurde ei too. See uus voog on kujutatud joonisel 24.



Joonis 24. Voog nelja sisendsignaaliga kasutades sõlmi uuest arendatud laiendusest.

Lambi põlemise loogikat kirjeldav funktsioon joonisel 22 täieneb samuti paari tingimuslause võrra. Kõigepealt tuleb hakata lisaks kuulama muutuseid kahes globaalse konteksti väärtuses funktsioonide *useGlobal(time)* ja *useGlobal(bathroom.humidity)* abil. Seejärel täieneb kood tingimuslausetega, mis nende väärtustega arvestavad.

Kokkuvõtvalt võib öelda, et arendatud laiendus Node-RED platvormile aitab lihtsustada paljude sisendsignaalide haldamist. Ilma laienduseta on tarvilik graafiliselt ehitada pikki, visuaalselt keerukaid ning seetõttu raskesti hallatavaid ja veaohtlikke automaatika vooge. Laienduse abiga muutuvad vood visuaalselt selgemaks ning kogu loogika süsteemi oleku üle otsustamiseks on koondunud ühte kohta.

5 Tulemused ja edasiarenduse võimalused

Töö tulemusena valmis laiendus automaatika platvormile Node-RED, mis täidab kõiki püstitatud nõudeid ja võimaldab süsteemi olekut hallata ühes keskses kohas. Selline lähenemine omakorda lihtsustab oluliselt automaatika voogude ülesehitust ning muudab need töökindlamaks. Lisaks võrreldi täna populaarseid targa kodu ja koduautomaatika süsteeme eesmärgiga mõista, kuidas on neis lahendatud paljude erinevate sisendsignaalide haldus. Läbiviidud analüüs võib osutuda kasulikuks ka teistele koduautomaatika huvilistele, kes soovivad ülevaadet erinevate platvormide võimekusest saada hakkama keerukamate automaatika reeglite seadistamisega.

Arendatud laiendus süsteemi oleku haldamiseks Node-RED platvormil on avalikustatud, st see on kättesaadav kõikidele Node-RED platvormi kasutajatele. Laienduse avalikustamiseks Node-RED voogude hoidlas on vajalik:

1. Avalikustada projekt võrgus olevas JavaScripti pakettide registris *npm* (*node package manager*). Arendatud laiendus on nähtav aadressil https://www.npmjs.com/package/@siirimangus/node-red-contrib-context-hook. Node-RED paketid peavad eeskirja kohaselt olema seotud kasutaja või organisatsiooni nimega, seetõttu on paketi nimele lisatud kasutaja @siirimangus.
2. Lisada laiendus Node-RED voogude hoidlasse. Arendatud laiendus on nähtav aadressil https://flows.nodered.org/node/@siirimangus/node-red-contrib-context-hook.

Laiendust on 14.05.2023 seisuga alla laetud 306 korda. Lisaks on see kasutusel töö autori koduautomaatika süsteemis. Selle abil koostatud automaatika vood toimivad hästi ning on lihtsustanud olulisel määral nende visuaalset esitust Node-RED graafilises kasutajaliideses.

Üheks projekti edasiarenduse võimaluseks on koodi testidega kaetuse suurendamine, kuna hetkel on kirjutatud ainult mõned üksustestid. Lisaks, kuna tegemist on avaliku paketiga, on edasises tarvilik jälgida, milline on kasutajate tagasiside ning kas teavitatakse vigadest.

Arendatud laienduse arhitektuuriline idee – süsteemi oleku jälgimine ning sellele automaatselt reageerimine – võiks olla filosoofia, mida võib-olla õnnestub rakendada ka teistel koduautomaatika platvormidel. Koduautomaatika süsteeme võiks käsitleda kui eesrakendusi, mille komponentideks on targad seadmed (lambid, radiaatorid, kõlarid jne) ning nende olek sõltub mingitest parameetritest (ruumis on valge, ruumis on liikumine, on päikesetõus jne). Eesmärk oleks ehitada mehhanism, mis võimaldaks parameetrite muutuste peale eesrakenduses automaatselt kohandusi sisse viia.

6 Kokkuvõte

Käesoleva töö üks eesmärk oli analüüsida hetkel populaarsemate koduautomaatika süsteemide võimekust seadistada paljudest erinevatest sisendsignaalidest sõltuvaid automaatika reegleid.

Osutub, et kommertsiaalsed süsteemid on head targa kodu juhtijad, st kasutajapoolse sekkumisega on need võimelised kasutaja elu mugavdama ja lihtsustama. Automaatika, st kasutajapoolse sekkumiseta poole peal on süsteemidel aga arenguruumi. Täna ei võimalda need konfigureerida kuitahes keerukaid loogikakonstruktsioone alustades näiteks pelgalt kahest lähtetingimusest „kui on tuvastatud liikumine ja on pime“.

Kogukonna poolt hallatud süsteemid on aga suunitletud justnimelt koduautomaatikale. See tähendab, et nendel platvormidel on võimalik seadistada kõikvõimalike loogikakonstruktsioonidega automaatika reegleid. Erinevate sisendsignaalide lisandumisega lähevad aga reeglid pikaks, raskesti loetavaks ja hallatavaks, veaohtlikuks.

Töö teine eesmärk oli tõhustada paljude erinevate sisendsignaalide haldust automaatika platvormil Node-RED. Töö tulemusena arendati laiendus, mis võimaldab automaatika vooge graafilises kasutajaliideses visuaalselt lihtsustada. Täituri järgmise oleku arvutus on laienduse abil koondatud ühte kesksesse kohta, mis aitab vähendada täituri mittesoovituid olekuid. Arendatud laiendus on avalikustatud, sellele on ligipääs kõikidel Node-RED platvormi kasutajatel.

Laienduse arendamise idee pärineb eesrakenduse teegist React. See võib olla filosoofia, mida üldiselt koduautomaatikale rakendada – koduautomaatika on eesrakendus, milles komponentide uuenemine toimub automaatselt vastavalt süsteemis toimunud oleku muutustele. See tähendab, et komponendid ise ei pea muutuste kohta järge pidama.

Kasutatud kirjandus

[1] „Smart Home Market Size, Share & Trends Analysis Report By Product, By Protocol (Wireless Protocols, Wired Protocols), By Application (New Construction, Retrofit), By Region, And Segment Forecasts, 2022 - 2030," [Võrgumaterjal]. Available: https://www.grandviewresearch.com/industry-analysis/smart-homes-industry. [Kasutatud 03 03 2023].

[2] „US Smart Home Statistics(2018–2025)," [Võrgumaterjal]. Available: https://www.oberlo.com/statistics/smart-home-statistics. [Kasutatud 22 02 2023].

[3] S. C. E. L. Gabriele Lobaccaro, „A Review of Systems and Technologies for Smart Homes and Smart Grids," 07 05 2016. [Võrgumaterjal]. Available: https://www.mdpi.com/1996-1073/9/5/348. [Kasutatud 03 03 2023].

[4] A. Hayes, „Smart Home: Definition, How They Work, Pros and Cons," 14 09 2022. [Võrgumaterjal]. Available: https://www.investopedia.com/terms/s/smart-home.asp. [Kasutatud 23 02 2023].

[5] S. Roger, „What is difference between smart home & home automation 2020," 25 02 2020. [Võrgumaterjal]. Available: https://medium.com/@samiyajutt.wbm/what-is-a-smart-home-ab59459649f6. [Kasutatud 03 03 2023].

[6] „Smart Home Vs. Connected Home Vs. Home Automation," [Võrgumaterjal]. Available: https://www.cannyelectrics.com.au/smart-home-vs-connected-home-vs-home-automation/. [Kasutatud 03 03 2023].

[7] „What's The Difference Between Smart Home Automation And Home Control?," 24 01 2020. [Võrgumaterjal]. Available: https://integratedlifestylesinc.com/blog/what-s-the-difference-between-smart-home-automation-and-home-control. [Kasutatud 03 03 2023].

[8] S. Cope, „Smart Home Automation Systems," 5 11 2022. [Võrgumaterjal]. Available: https://stevessmarthomeguide.com/smart-home-automation-systems/. [Kasutatud 03 03 2023].

[9] C. Habas, „What Are the Different Operating Standards for Home Automation Tech?," SafeWise, 01 11 2022. [Võrgumaterjal]. Available: https://www.safewise.com/faq/home-automation/home-automation-operating-standards/. [Kasutatud 12 03 2023].

[10] D. Galvin, „What Is A Smart Home Hub & Do You Need One?," 04 01 2023. [Võrgumaterjal]. Available: https://apacheiot.org/home-automation/what-is-a-smart-home-hub-do-you-need-one/#simplified-overview. [Kasutatud 12 03 2023].

[11] D. Galvin, „Guide To Home Automation Protocols," 4 01 2023. [Võrgumaterjal]. Available: https://apacheiot.org/home-automation/guide-to-home-automation-protocols/#what-is-a-communication-protocol. [Kasutatud 12 03 2023].

[12] M. Woodall, „How Home Automation Protocols Work," 11 02 2022. [Võrgumaterjal]. Available: https://www.reviews.org/home-security/home-automation-languages/. [Kasutatud 12 03 2023].

[13] K. Allemann, „Koduautomaatika tarkvara arendamine teenusepõhisest ärimudelist tulenevaid nõudeid järgides,“ Taltech Digikogu, 2017.

[14] „Google Trends: See what’s trending across Google Search, Google News and YouTube.,“ [Võrgumaterjal]. Available: https://newsinitiative.withgoogle.com/resources/lessons/google-trends-see-whats-trending-across-google-search-google-news-and-youtube/. [Kasutatud 30 03 2023].

[15] M. Diaz, „The 6 best home automation systems: Put your home on auto-pilot,“ ZDNET, 16 08 2022. [Võrgumaterjal]. Available: https://www.zdnet.com/home-and-office/smart-home/best-home-automation-system/. [Kasutatud 18 03 2023].

[16] Justin, „5 Best Home Automation System Options (Reviewed 2023!),“ 16 03 2023. [Võrgumaterjal]. Available: https://justjooz.com/best-home-automation-systems/. [Kasutatud 18 03 2023].

[17] „Google Trends,“ [Võrgumaterjal]. Available: https://trends.google.com/trends/explore?q=google%20home,amazon%20alexa,samsung%20smarthings,apple%20homekit,ezlo&hl=en-GB. [Kasutatud 18 03 2023].

[18] „Google Trends,“ [Võrgumaterjal]. Available: https://trends.google.com/trends/explore?q=home%20assistant,domoticz,openhab,node-red&hl=en-GB. [Kasutatud 18 03 2023].

[19] E. Lawrence, „What is Google Home?,“ 14 03 2023. [Võrgumaterjal]. Available: https://www.digitaltrends.com/home/google-home-defined/. [Kasutatud 27 03 2023].

[20] „Alexa Smart Home,“ [Võrgumaterjal]. Available: https://www.amazon.com/alexa-smart-home/b?ie=UTF8&node=21442899011&ref=pe_alxhub_aucc_en_us_IC_HP_1_HUB_SMA. [Kasutatud 28 03 2023].

[21] N. Laan, „Aeotec Smart Home Hub - Smartthings,“ 11 02 2022. [Võrgumaterjal]. Available: https://www.thesmarthomeblog.com/post/review-aeotec-smart-home-hub-smartthings/google-home-alexa/. [Kasutatud 22 04 2023].

[22] J. Pattison, „How to set up Google Home Household Routines,“ 28 10 2022. [Võrgumaterjal]. Available: https://www.theverge.com/23428772/google-home-household-routines-how-to-set-up. [Kasutatud 27 03 2023].

[23] N. Laan, „Aeotec Smart Home Hub - Smartthings,“ 11 02 2022. [Võrgumaterjal]. Available: https://www.thesmarthomeblog.com/post/review-aeotec-smart-home-hub-smartthings/automations-scenes-rules/. [Kasutatud 30 03 2023].

[24] A. I. Shaik, „Google is rolling out device triggers for Assistant routines,“ 21 10 2022. [Võrgumaterjal]. Available: https://www.sammobile.com/2022/10/20/device-triggers-assistant-routines/. [Kasutatud 22 04 2023].

[25] „Set up and manage routines,“ [Võrgumaterjal]. Available: https://support.google.com/googlenest/answer/7029585?hl=en-CA&co=GENIE.Platform%3DAndroid#zippy=%2Ccreate-a-household-routine. [Kasutatud 13 04 2023].

[26] „Alexa Routines,“ [Võrgumaterjal]. Available: https://www.amazon.com/alexa-routines/b?ie=UTF8&node=21442922011&ref_alxhb_smhm_text_creat_cstm_rtns. [Kasutatud 29 03 2023].

[27] J. Pattison, „How to create an Alexa Routine,“ 19 08 2022. [Võrgumaterjal]. Available: https://www.theverge.com/23312859/amazon-alexa-echo-routine-smart-speaker-how-to. [Kasutatud 29 03 2023].

[28] Y. Griffith, „How To Only Run Alexa Routines At Night (Or After Sunset),“ 27 10 2022. [Võrgumaterjal]. Available: https://www.smarthomepoint.com/alexa-routines-only-at-night/. [Kasutatud 13 04 2023].

[29] „Rules,“ [Võrgumaterjal]. Available: https://developer.smartthings.com/docs/automations/rules. [Kasutatud 13 04 2023].

[30] „ Maximum Number of Automation Routines?,“ 11 2021. [Võrgumaterjal]. Available: https://community.smartthings.com/t/maximum-number-of-automation-routines/235260/6. [Kasutatud 31 03 2023].

[31] „Set Up an Alexa Routine,“ [Võrgumaterjal]. Available: https://www.amazon.com/gp/help/customer/display.html?nodeId=G2PYLKJN3XVZ55EQ. [Kasutatud 31 03 2023].

[32] „Installation,“ [Võrgumaterjal]. Available: https://www.home-assistant.io/installation/. [Kasutatud 01 04 2023].

[33] „Integrations,“ [Võrgumaterjal]. Available: https://www.home-assistant.io/integrations/. [Kasutatud 22 04 2023].

[34] „Node-RED library,“ [Võrgumaterjal]. Available: https://flows.nodered.org/. [Kasutatud 22 04 2023].

[35] „Mobile App,“ [Võrgumaterjal]. Available: https://www.home-assistant.io/integrations/mobile_app/. [Kasutatud 01 04 2023].

[36] „Understanding Automations,“ [Võrgumaterjal]. Available: https://www.home-assistant.io/docs/automation/basics/. [Kasutatud 01 04 2023].

[37] „Automation YAML,“ [Võrgumaterjal]. Available: https://www.home-assistant.io/docs/automation/yaml/. [Kasutatud 22 04 2023].

[38] „Customizing entities,“ [Võrgumaterjal]. Available: https://www.home-assistant.io/docs/configuration/customizing-devices/. [Kasutatud 01 04 2023].

[39] „Automation Trigger,“ [Võrgumaterjal]. Available: https://www.home-assistant.io/docs/automation/trigger/. [Kasutatud 22 04 2023].

[40] „Script Syntax,“ [Võrgumaterjal]. Available: https://www.home-assistant.io/docs/scripts/. [Kasutatud 02 04 2023].

[41] „Script Syntax,“ [Võrgumaterjal]. Available: https://www.home-assistant.io/docs/scripts/. [Kasutatud 22 04 2023].

[42] „Things,“ [Võrgumaterjal]. Available: https://www.openhab.org/docs/concepts/things.html. [Kasutatud 03 04 2023].

[43] „Items,“ [Võrgumaterjal]. Available: https://www.openhab.org/docs/concepts/items.html. [Kasutatud 03 04 2023].

[44] „Basic Rules and Rule Templates,“ [Võrgumaterjal]. Available: https://www.openhab.org/docs/tutorial/rules_basic.html. [Kasutatud 15 04 2023].

[45] „Advanced Rules,“ [Võrgumaterjal]. Available: https://www.openhab.org/docs/tutorial/rules_advanced.html#but-only-if-conditions. [Kasutatud 15 04 2023].

[46] „Node-RED. Low-code programming for event-driven applications,“ [Võrgumaterjal]. Available: https://nodered.org/. [Kasutatud 19 03 2023].

[47] K. Bourke, „What is Node-RED,“ 16 12 2022. [Võrgumaterjal]. Available: https://realpars.com/node-red/. [Kasutatud 19 03 2023].

[48] „Node context,“ [Võrgumaterjal]. Available: https://nodered.org/docs/creating-nodes/context. [Kasutatud 08 04 2023].

[49] „Creating your first node,“ [Võrgumaterjal]. Available: https://nodered.org/docs/creating-nodes/first-node. [Kasutatud 04 04 2023].

[50] „Events,“ [Võrgumaterjal]. Available: https://nodejs.org/api/events.html#events. [Kasutatud 23 04 2023].

[51] „DubhAd/Home-AssistantConfig,“ [Võrgumaterjal]. Available: https://github.com/DubhAd/Home-AssistantConfig/blob/live/automation/master_bedroom/master_bedroom_button_pushed.yaml. [Kasutatud 14 04 2023].

[52] „A simpler way to create your smart home,“ [Võrgumaterjal]. Available: https://www.amazon.com/b?node=37490568011&ref_pe_alxhub_aucc_en_us_SMA_BT_2_HUBI_37490568011. [Kasutatud 28 03 2023].

[53] „Add-on Reference,“ [Võrgumaterjal]. Available: https://www.openhab.org/addons/. [Kasutatud 22 04 2023].

Lisa 1 – Lihtlitsents lõputöö reprodutseerimiseks ja lõputöö üldsusele kättesaadavaks tegemiseks[1]

Mina, Siiri Mangus

1. Annan Tallinna Tehnikaülikoolile tasuta loa (lihtlitsentsi) enda loodud teose „Koduautomaatika süsteemi oleku haldamine Node-RED näitel“, mille juhendajad on Kristiina Hakk ja Mikk Mangus
   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.

15.05.2023

1 Lihtlitsents ei kehti juurdepääsupiirangu kehtivuse ajal vastavalt üliõpilase taotlusele lõputööle juurdepääsupiirangu kehtestamiseks, mis on allkirjastatud teaduskonna dekaani poolt, välja arvatud ülikooli õigus lõputööd reprodutseerida üksnes säilitamise eesmärgil. Kui lõputöö on loonud kaks või enam isikut oma ühise loomingulise tegevusega ning lõputöö kaas- või ühisautor(id) ei ole andnud lõputööd kaitsvale üliõpilasele kindlaksmääratud tähtajaks nõusolekut lõputöö reprodutseerimiseks ja avalikustamiseks vastavalt lihtlitsentsi punktidele 1.1. ja 1.2, siis lihtlitsents nimetatud tähtaja jooksul ei kehti.

Lisa 2 – Automaatika reegel SmartThings platvormil, mida on võimalik salvestada Rules API kaudu [29]

```
{
    "name": "Sample for precondition",
    "actions": [
        {
            "if": {
                "equals": {
                    "left": {
                        "string": "pushed"
                    },
                    "right": {
                        "device": {
                            "devices": ["button-device-id"],
                            "component": "main",
                            "capability":"button",
                            "attribute":"button",
                            "trigger":"Always"
                        }
                    }
                },
                "then": [
                    {
                        "if": {
                            "equals": {
                                "left": {
                                    "device": {
                                        "devices": [
                                            "switch1-device-id"
                                        ],
                                        "component": "main",
                                        "capability": "switch",
                                        "attribute": "switch",
                                        "trigger":"Never"
                                    }
                                },
                                "right": {
                                    "string": "on"
                                }
                            },"then": [
                            {
                                "command": {
                                    "devices": [
                                        "switch2-device-id"
```

```
                                ],
                                "commands": [
                                    {
                                        "component": "main",
                                        "capability": "switch",
                                        "command": "on",
                                        "arguments": []
                                    }
                                ]
                            }
                        }],"else": [{
                            "command": {
                                "devices": [
                                    "switch2-device-id"
                                ],
                                "commands": [
                                    {
                                        "component": "main",
                                        "capability": "switch",
                                        "command": "off",
                                        "arguments": []
                                    }
                                ]
                            }
                        }]
                    }
                }
            ]
        }
    }
]
}
```

Lisa 3 – Automaatika reegel Home Assistant platvormil [51]

```
id: 'master_bedroom_button_pushed'
initial_state: 'on'
alias: 'Master bedroom button pushed'
trigger:
  - platform: state
    entity_id: sensor.master_bedroom_button_action
    to:
    - 'single'
    - 'double'
action:
  - choose:
    # Single press - turn the lights off if they're on, unless it's in motion
    # mode where we turn it on. Turn them on otherwise.
    - conditions:
      - condition: template
        value_template: "{{ trigger.to_state.state == 'single' }}"
      sequence:
      - choose:
        - conditions:
          - condition: state
            entity_id: input_select.master_bedroom
            state:
            - 'On'
            - 'Bedtime'
            - 'Motion'
            - 'Wake'
          sequence:
          - service: input_select.select_option
            data:
              entity_id: input_select.master_bedroom
              option: 'Off'
        - conditions:
          - condition: state
            entity_id: input_select.master_bedroom
            state:
            - 'Off'
            - 'Motion'
            - 'Auto'
          sequence:
          - service: input_select.select_option
            data:
              entity_id: input_select.master_bedroom
              option: 'On'
```

```
    default:
    - service: input_select.select_option
      data:
        entity_id: input_select.master_bedroom
        option: 'On'
# Double press, On if currently bedtime, else Bedtime
- conditions:
  - condition: template
    value_template: "{{ trigger.to_state.state == 'double' }}"
  sequence:
  - choose:
    - conditions:
      - condition: state
        entity_id: input_select.master_bedroom
        state:
        - 'Bedtime'
      sequence:
      - service: input_select.select_option
        data:
          entity_id: input_select.master_bedroom
          option: 'On'
    default:
    - service: input_select.select_option
      data:
        entity_id: input_select.master_bedroom
        option: 'Bedtime'
default:
- service: input_select.select_option
  data:
    entity_id: input_select.master_bedroom
    option: 'Auto'
```