Ojala Henri

M2M-kommunikaatio langattomassa lähiverkossa

Opinnäytetyö
Kevät 2019
SeAMK Tekniikka
Automaatiotekniikan Tutkinto-ohjelma

SeAMK
SEINÄJOEN AMMATTIKORKEAKOULU
SEINÄJOKI UNIVERSITY OF APPLIED SCIENCES

SEINÄJOEN AMMATTIKORKEAKOULU

Opinnäytetyön tiivistelmä

Koulutusyksikkö: Tekniikan yksikkö

Tutkinto-ohjelma: Automaatiotekniikka

Suuntautumisvaihtoehto: Sähköautomaatio

Tekijä: Henri Ojala

Työn nimi: M2M-kommunikaatio langattomassa lähiverkossa

Ohjaaja: Marko Hietamäki

Vuosi: 2019 Sivumäärä: 37

Tämän opinnäytetyön tavoitteena oli selvittää mahdollinen toteutustapa langattomassa lähiverkossa käytävälle M2M (Machine-to-Machine) -kommunikaatiolle Epec Oy:n 6000-sarjan ohjausyksiköillä. Lisäksi tavoitteena oli toteuttaa yksinkertainen testisovellus, jolla toteutustavan toimivuus pystyttäisiin todentamaan.

Opinnäytetyössä tutkittiin neljän eri sovellustason ratkaisun soveltuvuutta Epec Oy:n 6000-sarjan ohjausyksiköiden väliseen M2M-kommunikointiin. Tutkitut toteutustavat olivat MQTT, CoAP, DDS sekä ZeroMQ. Opinnäytetyössä tehdyn vertailun perusteella käytettäväksi toteutustavaksi valikoitui ZeroMQ. Lisäksi opinnäytetyössä selvitettiin langattoman lähiverkon eri verkkotopologioita.

ZeroMQ-ohjelmistokirjaston avulla toteutettiin testisovellus, jossa kaksi Epec Oy:n 6000-sarjan laitetta kommunikoivat langattomassa lähiverkossa.

Tämän opinnäytetyön avulla, testisovellusta jatkokehittämällä, pystytään mahdollisesti toteuttamaan valmis ratkaisu Epec Oy:n 6000-sarjan ohjausyksiköiden väliseen M2M-kommunikaatioon langattomassa lähiverkossa.

Avainsanat: M2M, MQTT, CoAP, DDS, ZeroMQ, WLAN

SEINÄJOKI UNIVERSITY OF APPLIED SCIENCES

Thesis abstract

Faculty: School of Technology

Degree programme: Automation Engineering

Specialisation: Electric Automation

Author: Henri Ojala

Title of thesis: M2M Communication in a Wireless Local Area Network

Supervisor: Marko Hietamäki

Year: 2019 Number of pages:37

The purpose of this thesis was to find a solution for M2M communication between the control units of Epec 6000 series in a wireless local area network. The other goal was to implement a test application which would verify the functionality of the solution.

The thesis studied and compared four different application levels of M2M communication techniques: MQTT, CoAP, DDS and ZeroMQ. The thesis also studied wireless local area network topologies.

As a result, the comparison pointed out that the most suitable solution for the M2M communication was ZeroMQ. ZeroMQ library was used in the test application, where two units of Epec 6000 series communicated in a local area network. By improving the test application, which was implemented in this thesis, Epec can further develop a solution for M2M communication in wireless local area networks.

Keywords: M2M, MQTT, CoAP, DDS, ZeroMQ, WLAN

SISÄLTÖ

Kuvaluettelo

Käytetyt termit ja lyhenteet

M2M	M2M eli Machine-to-Machine tarkoittaa kahden tai useamman laitteen välistä kommunikointia, johon ihmisen ei tarvitse puuttua.
MQTT	MQTT eli Message Queuing Telemetry Transport on OASIS-järjestön standardoima julkaisija–tilaaja-mallin viestintäprotokolla.
CoAP	CoAP eli Constrained Application Protocol on HTTP-pohjainen viestintäprotokolla, joka mahdollistaa hyvin pienten laitteiden liittämisen web-sovelluksiin.
DDS	DDS eli Data Distribution Service on Object Management Groupin kehittämä viestintästandardi laitteiden väliseen kommunikointiin.
ZeroMQ	ZeroMQ on sovellusten väliseen kommunikointiin kehitetty avoimen lähdekoodin ohjelmistokirjasto.
Protobuf	Protobuf eli Protocol buffers on Googlen kehittämä menetelmä luokitellun tiedon muutamiseen binäärimuotoon ja binäärimuodosta takaisin.
Serialisointi	Tarkoittaa luokitellun tiedon muuttamista merkkijonoiksi.
Socket	Socket eli pistoke on rajapinta tiedon lähettämiseen ja vastaanottamiseen päätepisteiden välillä.
IoT	IoT eli Internet of Things tarkoittaa esineiden ja asioiden liittämistä Internet-verkkoon.
HTTP	HTTP eli Hypertext Transfer Protocol on protokolla, jota selaimet ja WWW-palvelimet käyttävät tiedonsiirtoon.

MANET	MANET eli Mobile Ad-Hoc Network on langaton lähiverkko, jossa ei ole yhtään tukiasemaa, vaan verkossa olevat laitteet itse muodostavat lähiverkon ja toimivat verkossa myös reitittiminä.
WLAN	WLAN eli Wireless Local Area Network tarkoittaa langatonta lähiverkkoyhteyttä.

1 JOHDANTO

1.1 Tausta

M2M-kommunikaatio on kasvava trendi, myös liikkuvien työkoneiden valmistajat voivat tuoda laitteisiinsa uusia ominaisuuksia M2M-kommunikaation avulla. Epec Oy:llä on olemassa ratkaisu GSM- ja 3G-verkossa toimivaan M2M-kommunikaatioon, mutta ohjausyksiköiden väliseen langattomalla lähiverkolla toimivaan kommunikaatioon ei ole vielä olemassa ratkaisua. Monet liikkuvat työkoneet toimivat olosuhteissa, joissa ei ole GSM- tai 3G-yhteyttä, vaan laitteiden täytyisi tällöin hyödyntää langatonta lähiverkkoa.

1.2 Tavoitteet

Tämän opinnäytetyön tavoitteena oli tutkia ja selvittää, millä tavalla Epec Oy:n 6000-sarjan laitteiden välinen M2M-kommunikaatio olisi mahdollista toteuttaa langattomassa lähiverkossa. Tavoitteena oli myös toteuttaa toimiva testisovellus, jossa kaksi 6000-sarjan ohjausyksikköä kommunikoivat langattomassa lähiverkossa. Testisovelluksen kommunikaatiossa käytettävä tieto oli rajattu laitteiden kuvitteelliseen prosessidataan, jota voi olla esimerkiksi kokonaislukuarvot. Tässä opinnäytetyössä ei huomioitu M2M-kommunikaation viestien salausta, sillä tämän opinnäytetyön käsittelemä M2M-kommunikaatio on tarkoitus toteuttaa suljetussa lähiverkossa.

1.3 Työn rakenne

Luvussa 2 tutustutaan langattoman lähiverkon erilaisiin verkkotopologioihin sekä esitellään neljä erilaista M2M-kommunikaation toteutusvaihtoehtoa. Luvussa 3 määritellään M2M-kommunikaation vaatimukset Epec Oy:n 6000-sarjan laitteiden välisen kommunikaation toteuttamiseksi langattomassa lähiverkossa, näiden vaatimusten pohjalta tehdään valinta käytettävästä toteutustavasta. Luvussa 4 esitellään Epec Oy:n valmistamat WLAN-verkkoa tukevat laitteet, sekä käydään läpi M2M-

kommunikaation testisovelluksen toteutus. Luku 5 sisältää yhteenvedon opinnäytetyöstä sekä arvion valitun toteutustavan toimivuudesta.

1.4 Epec Oy

Epec Oy on Seinäjoella toimiva, liikkuvien työkoneiden ohjausjärjestelmiin erikoistunut ratkaisutoimittaja. Tuotevalikoimaan kuuluu sulautettuja ohjausyksiköitä, sulautettuja näyttöjä sekä IoT-laitteita. Laitteet ovat IP-luokitukseltaan ja tärinän kestoltaan suunniteltu kestämään kovia olosuhteita. Yrityksen palveluihin kuuluvat lisäksi ohjausjärjestelmien ohjelmistosuunnittelu ja -testaus, käyttöönotto, asiakastuki ja asiakaskoulutus. Epec tarjoaa myös ohjelmistotuotteita ohjausjärjestelmän konfigurointiin, diagnostiikkaan sekä ylläpitoon. (Epec [Viitattu 13.2.2019].)

2 M2M-KOMMUNIKAATIO JA LANGATON LÄHIVERKKO

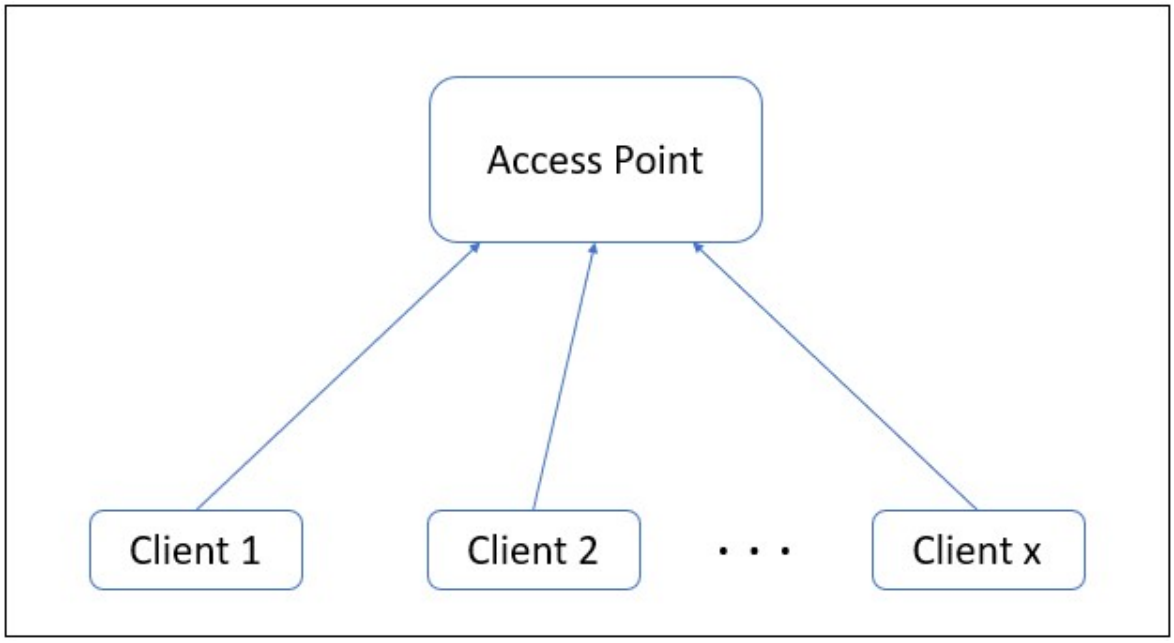
M2M (Machine-to-Machine) -kommunikaatio on kahden tai useamman laitteen välistä viestintää, johon ihmisen ei tarvitse välttämättä osallistua. M2M-kommunikaatioon pohjautuvien laitteiden avulla voidaan automatisoida haluttujen koneiden keskeiset viestintä- ja päätöksentekoprosessit. (ETSI 2013.)

Tässä luvussa tutustutaan M2M-kommunikaation sovellustason toteutustapoihin sekä langattoman lähiverkon verkkotopologioihin.

2.1 Langattoman lähiverkon verkkotopologiat

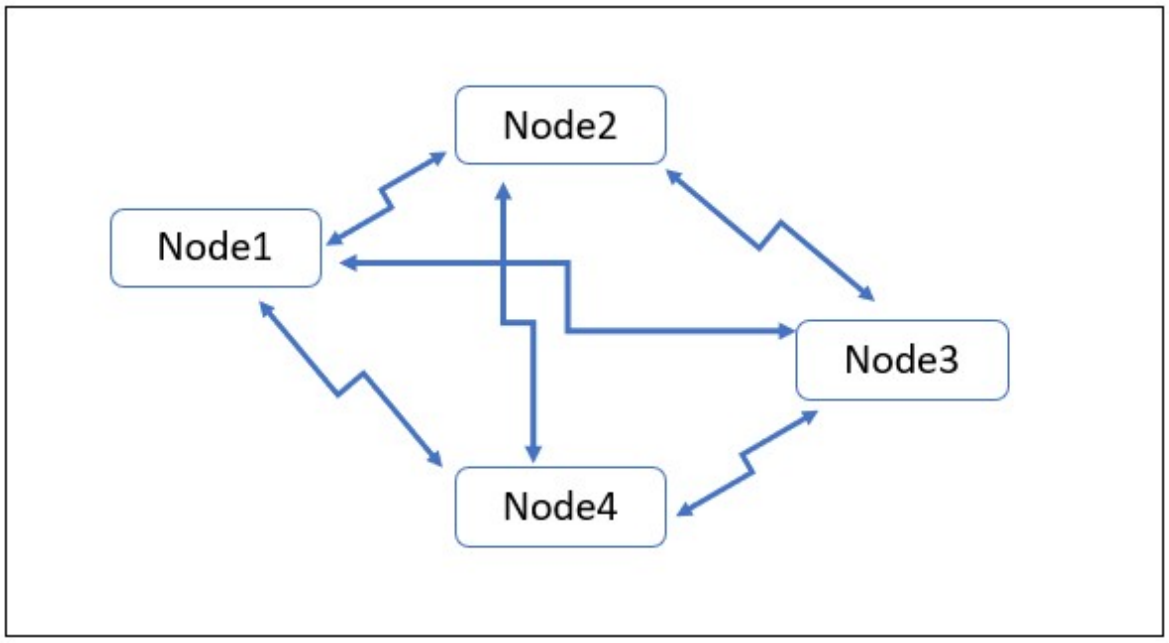
Puhuttaessa langattomasta lähiverkosta, tarkoitetaan sillä yleensä IEEE 802.11-standardia. Standardille käytetään myös markkinointitermiä Wi-Fi. IEEE 802.11, tai toisin sanoen Wi-Fi, on joukko standardeja, jotka IEEE-järjestö on kehittänyt langattomaan viestintään 2,5 gHz:n, 3,6 gHz:n ja 5 gHz:n taajuusalueille. Standardi määrittelee kaksi erilaista tapaa muodostaa langaton lähiverkko, infrastruktuuriverkko sekä Ad-Hoc-verkko. (Wei, Rykowski & Dixit 2013, 109-110.)

Infrastruktuuriverkossa laitteet yhdistyvät ja kommunikoivat toisilleen tukiaseman kautta, kun taas Ad-Hoc-verkossa laitteet yhdistyvät suoraan toisiinsa. Alkuperäinen Ad-Hoc-verkko ei sisältänyt tukea multi-hop-toiminnolle, joka mahdollistaa toistensa kantamattomissa olevien laitteiden kommunikoinnin niiden välissä olevan tai olevien laitteiden kautta. M2M-kommunikaatioon perustuvien ratkaisujen yleistyttyä, multi-hop-toimintoa tukeville verkoille nähtiin olevan tarvetta. Tuki multi-hop-ominaisuudelle tuli 802.11s-standardin myötä. Multi-hop-toimintoa tukevia Ad-Hoc-verkkoja kutsutaan MANET-verkoiksi. (Wei ym. 2013, 109-110.)



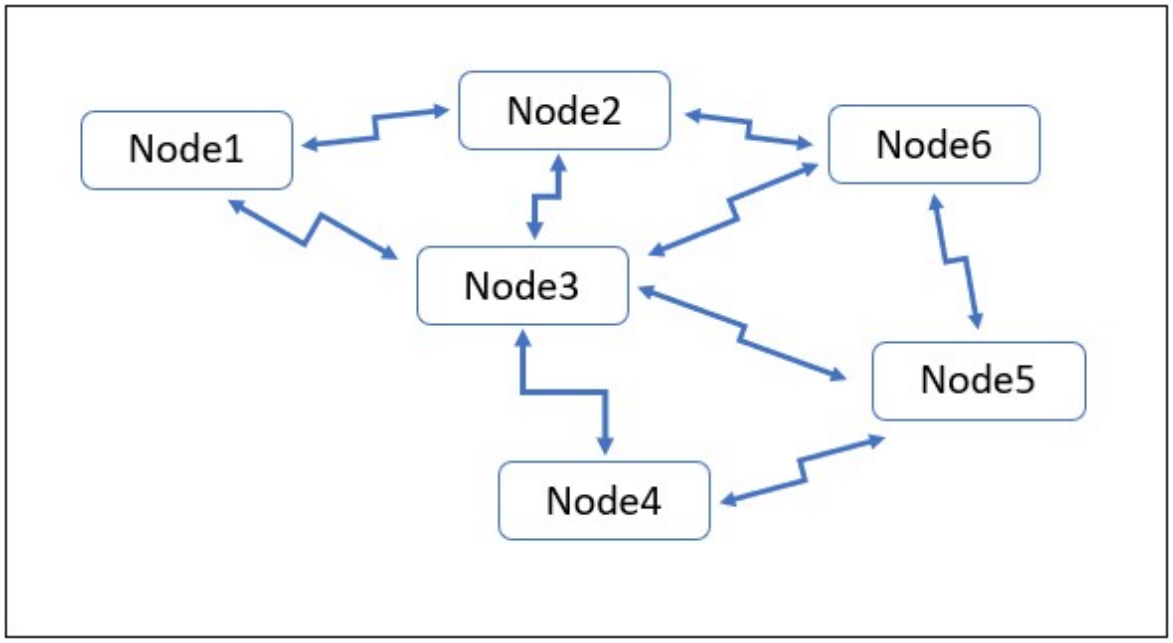
Kuva 1. Infrastruktuuriin perustuva verkkotopologia
(Wei ym. 2013, 109)

Kuvassa 1 on esitetty infrastruktuuriin perustuva WLAN-verkkotopologia. Tässä verkkotopologiassa asiakkaat (Clients) yhdistyvät verkon välityksellä tukiasemaan (Access Point). Tukiaseman välityksellä ne voivat kommunikoida toisilleen tai yhdistyä ulkoisiin verkkoihin, kuten internetiin.



Kuva 2. Ad-Hoc-verkkotopologia
(Wei ym. 2013, 109)

Kuvassa 2. on esitetty Ad-Hoc-verkkotopologia, jossa verkossa olevat laitteet tai toisin sanottuna verkon solmupisteet (Nodes) yhdistyvät suoraan toisiinsa.



Kuva 3. MANET verkkotopologia
(Wei ym. 2013, 114)

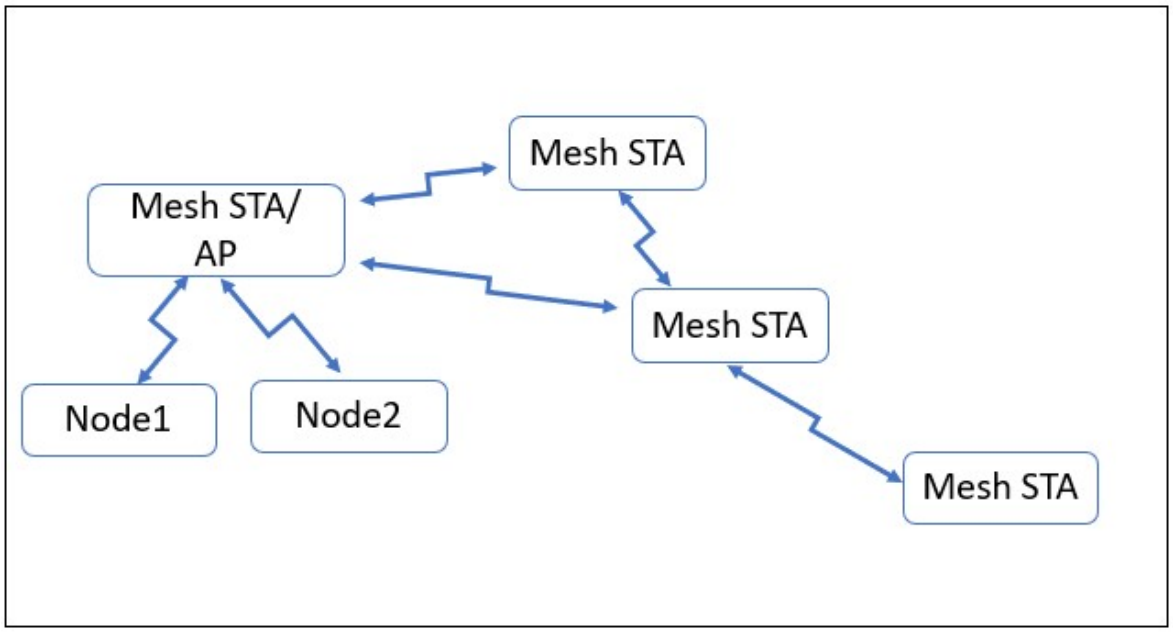
Kuvassa 3 on esitetty MANET-verkkotopologia, jossa verkon solmupisteet voivat yhdistyä toisiinsa, sekä reitittää verkkoa myös eteenpäin. Tässä kuvassa esimerkiksi Node1 ja Node5 pystyvät kommunikoimaan toisilleen Node3-solmupisteen välityksellä.

Esimerkiksi autojen väliseen langattomaan tiedonsiirtoon kehitetään MANET-verkkojen tapaisia ratkaisuja. Autojen välisiä verkkoja kutsutaan VANET-verkoiksi (Vehicular Ad-Hoc networks). Autojen välisissä verkoissa autojen tulee pystyä kommunikoimaan sekä muiden autojen että tien varressa olevan infrastruktuurin kanssa. (Abbasi 2018, 2-5.)

2.1.1 IEEE 802.11s mesh -standardi

Wireless Mesh -verkoksi kutsutaan verkkoa, jossa IEEE 802.11 -standardia tukevat laitteet muodostavat Mesh-topologian. Mesh-topologiassa laitteet voivat olla yhteydessä joko suoraan toisiinsa tai verkossa olevien toisten laitteiden välityksellä. Lan-

gattomia Mesh-verkkoja on kahdenlaisia. On verkkoja, joissa reitittimet muodostavat Mesh-topologian ja päätelaitteet yhdistyvät reitittimiin, tai toisessa tapauksessa päätelaitteet muodostavat itsenäisen MANET-verkon (Mobile Ad-Hoc Network), jossa ei ole tukiasemia, vaan laitteet itsessään reitittävät verkkoa eteenpäin. Mesh-verkko voi myös olla topologialtaan sellainen, jossa yhdistyy molemmat tapaukset. Tällainen verkkotopologia on esitetty kuvassa 4. (Wei ym. 2013, 109-111.)



Kuva 4. Mesh verkkotopologia
(Wei ym. 2013, 118)

Kuva 4 esittää Mesh-verkkotopologiaa, jossa ”Mesh STA” -solmut ovat joko Mesh-standardia tukevia reitittimiä tai Mesh-standardia tukevia pääteasemia. ”Mesh STA/AP” on reititin, joka mahdollistaa perinteisten infrastruktuuriverkkoa tukevien solmujen yhdistymisen Mesh-verkkoon. Verkossa voi olla myös yksi tai useampi portti, jonka kautta verkossa olevat laitteet voivat olla yhteydessä ulkoisiin verkkoihin.

Mesh-verkkoja on aikaisemmin rakennettu hyödyntäen IEEE 802.11b/g/n -standardeja. Näissä verkoissa on kuitenkin ollut ongelmia suorituskyvyn ja ruuhkautumisen kanssa. IEEE-järjestö kehitti uuden 802.11s Mesh -verkkostandardin vastaamaan tähän ongelmaan. Standardi 802.11s määrittelee esimerkiksi, kuinka laitteet löytävät toisensa monimutkaisessa verkossa ja mitä reittiä ne keskustelevat muiden verkossa olevien laitteiden kanssa, huomioiden verkon kuormituksen eri solmupisteissä. (Wei ym. 2013, 116-117)

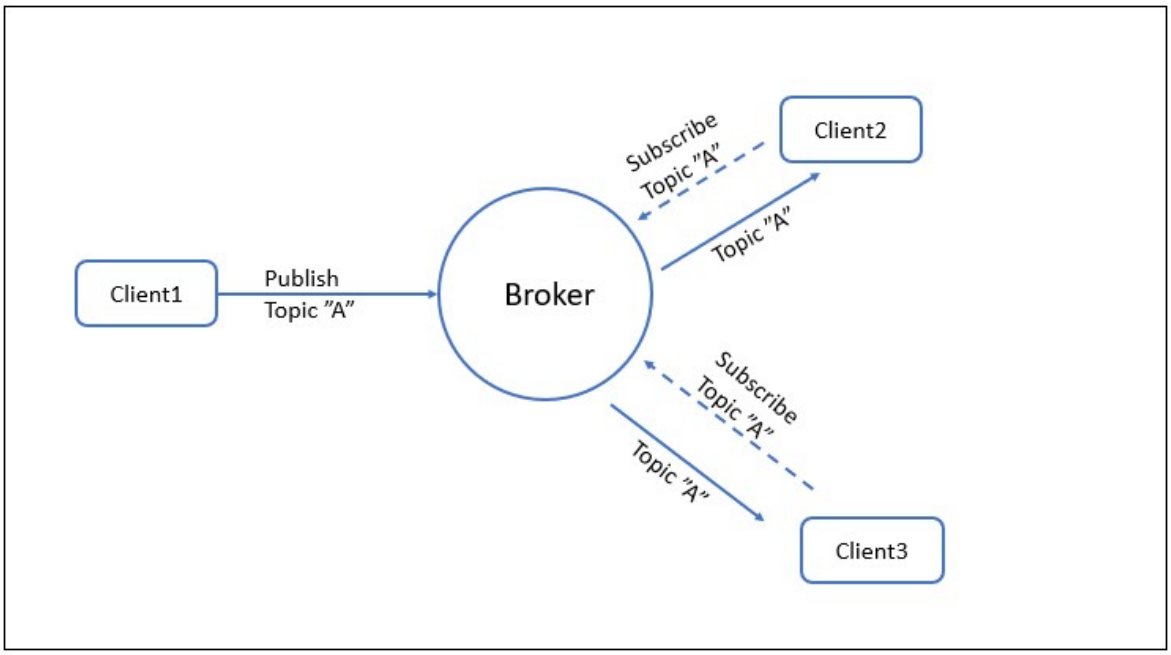
2.2 Sovellustason tekniikat M2M-kommunikointiin

M2M-kommunikointiin on paljon erilaisia sovellustason protokollia, väliohjelmistoja ja ohjelmistokirjastoja. Jokaisella ratkaisulla on omat vahvuutensa. Valittaessa oikeaa ratkaisua omaan sovellukseen on valinta tehtävä oman sovelluksen vaatimusten mukaan. (Meng, Zhipeng & Gray 2017, 4.)

Seuraavaksi esitellään neljä erilaista sovellustason tekniikkaa M2M-kommunikaatioon: MQTT, CoAP, DDS sekä ZeroMQ. Luvussa 3.2 vertaillaan, mikä näistä tekniikoista soveltuu parhaiten Epec Oy:n 6000-sarjan laitteiden väliseen langattomaan M2M-kommunikointiin.

2.2.1 MQTT

MQTT (Message Queuing Telemetry Transport) on Oasis-järjestön standardoima kevytrakenteinen julkaisija–tilaaja (publish–subscribe) -mallinen viestintäprotokolla. Protokollan keskiössä on välittäjä (broker), johon julkaisijat ja tilaajat yhdistyvät. Julkaisijat julkaisevat viestejä välittäjälle aihealueittain, ja tilaajat voivat valita, minkä aihealueen viestejä haluavat heille välitettävän. MQTT käyttää viestien kuljettamiseen TCP-protokollaa. (Oasis 2014.)



Kuva 5. MQTT viestiketju
(Eclipse 2014.)

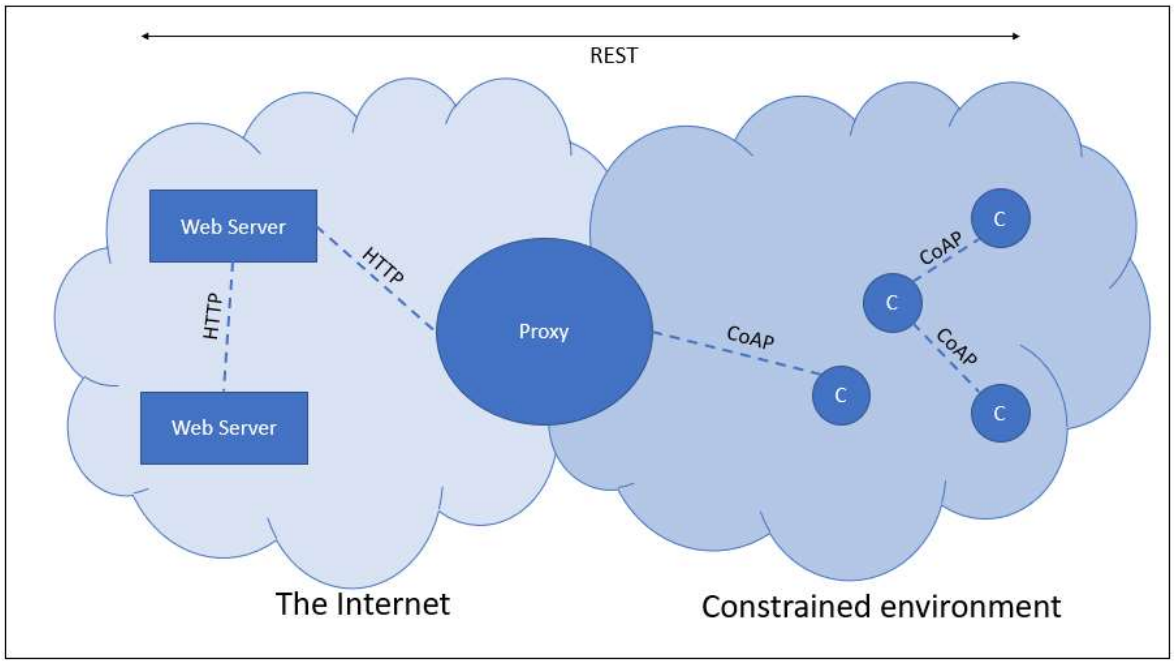
Kuvassa 5 on esimerkki MQTT-protokollan viestiketjusta. Kuvassa ”Client1” julkaisee viestiketjun keskiössä olevalle välittäjälle ”Broker” viestejä aiheella ”A”. ”Client2” sekä ”Client3” ovat tilanneet aiheen ”A” viestit välittäjältä. Kun ”Client1” julkaisee viestin aiheella ”A”, välittäjä välittää sen edelleen ”Client2”- ja ”Client3”-tilaajalle.

Viestejä voidaan julkaista kolmella eri laatutasolla (QoS). Ensimmäisellä laatutasolla viestien välittymistä ei varmisteta. Toisella laatutasolla varmistetaan, että viestit pääsevät perille, mutta sama viesti voi tulla tilaajalle useita kertoja. Kolmannella laatutasolla varmistetaan, että viestit kulkeutuvat vastaanottajalle täsmälleen kerran. (Oasis 2014.)

2.2.2 CoAP

CoAP (Constrained Application Protocol) on erityisesti kiinteistöautomaatiossa ja energian mittaamisessa käytetty viestintäprotokolla. Protokolla on suunniteltu pienen laskentatehon omaaville ja pienen virrankulutuksen vaativille laitteille. Se jäljittelee web-tiedonsiirrossa yleisesti käytettyä HTTP-protokollaa ja REST-arkkitehtuuria, joten tiedon välittäminen web-palvelimille sujuu ketterästi. (IETF 2014.)

CoAP-palvelin sekä asiakas keskustelevat pyyntö–vastaus-mallilla (request–response), jossa asiakas pyytää palvelimelta tietoa, johon palvelin vastaa. Tiedonsiirtoketjussa jokainen solmu voi toimia sekä palvelimena että asiakkaana. Viestejä voidaan lähettää kahdella eri laatutasolla. Ne voivat olla joko vahvistettuja, jotka vastaanottaja vahvistaa saapuneiksi, tai vahvistamattomia, joihin ei vaadita kuittausta. Viestit välitetään TCP-tiedonsiirtoa kevyemmällä UDP-protokollalla. (IETF 2014.)



Kuva 6. CoAP arkkitehtuuri
(U-blox 2016.)

Kuvassa 6 on malli CoAP-protokollan arkkitehtuurista. Kuten kuvassa on esitetty, CoAP-laitteet voivat kommunikoida suoraan toistensa kanssa, tai ne voivat olla yhteydessä muihin verkkoihin välityspalvelimen (Proxy) kautta.

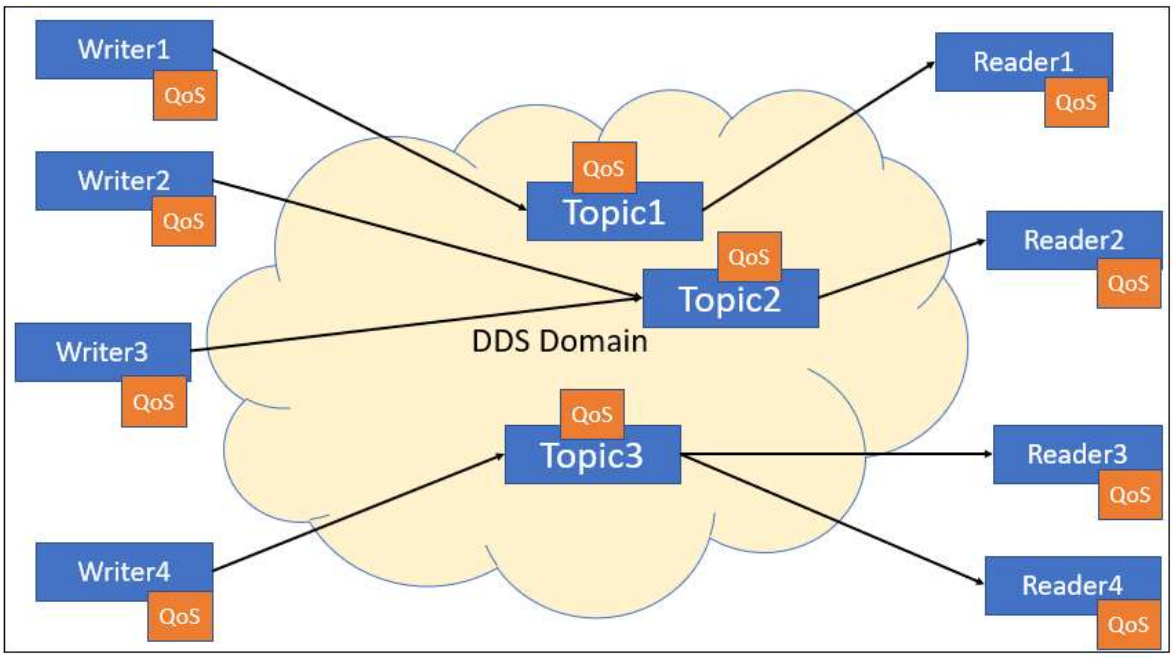
2.2.3 DDS

DDS (Data Distribution Service) on Object Management Groupin kehittämä väliohjelmistoprotokolla ja ohjelmointirajapintojen standardi datakeskeiseen kommunikointiin. Se yhdistää hajautetun järjestelmän eri osat toisiinsa, mahdollistaen helpon ja luotettavan, samanaikaisesti useankin liittyjän, välisen tiedonsiirron. Viestintä ta-

pahtuu julkaisija–tilaaja-mallin mukaisesti. Kaikki tätä väliohjelmistoa käyttävät laitteet näkevät julkaistun datan niin kuin se olisi paikallisesti tuotettua tietoa. (DDS-foundation [Viitattu 1.3.2019].)

Ohjelmiston takaama viestien luotettavuus perustuu sen todella laajaan viestien luokittelu- ja priorisointivalikoimaan. Kommunikaation laatutasolle voidaan määritellä yli 20 erilaista vaikuttavaa tekijää. Eri laatutasoja voivat olla esimerkiksi viestin toimitusaika, viestin päivitystaajuus tai viestin elinaika. DDS sisältää myös dynaamisen haun, joka automaattisesti etsii verkossa olevat julkaisijat ja tilaajat ja yhdistää ne toisiinsa. (OpenDDS [Viitattu 1.3.2019].)

DDS-protokollalle löytyy useita kaupallisia toteutuksia eri ohjelmointikielille, sekä avoimen lähdekoodin toteutus c++-ohjelmointikielelle. (DDS-foundation [Viitattu 2.3.2019].)

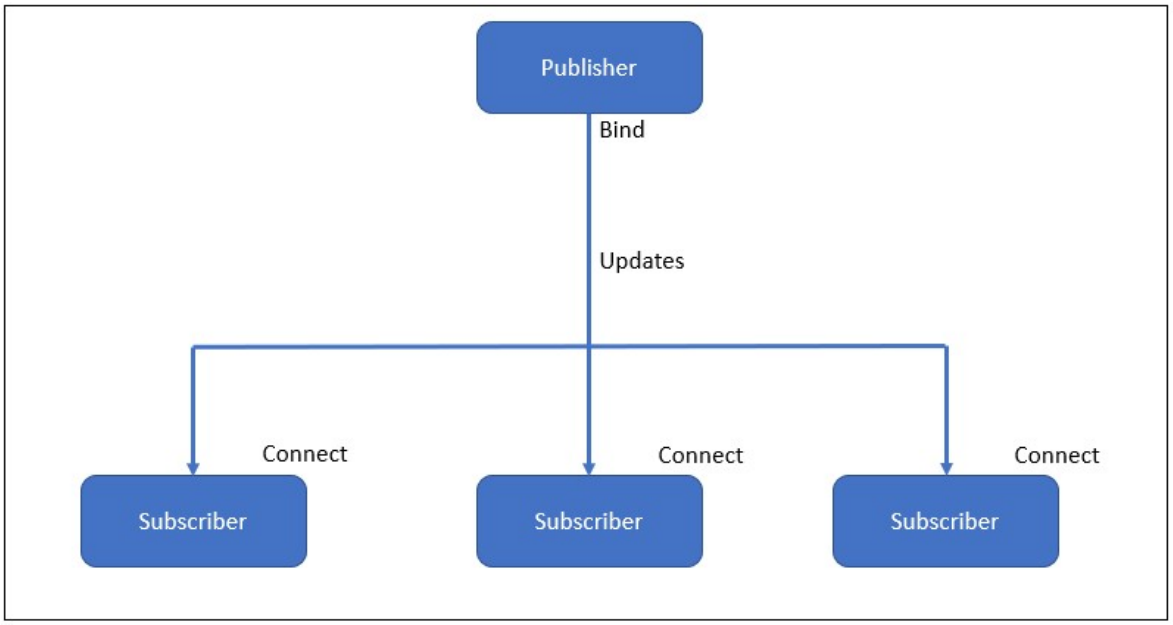


Kuva 7. DDS viestiketju
(DDS-foundation [Viitattu 1.3.2019].)

Kuvassa 7 on havainnollistettu DDS-viestiketjun toimintaa. DDS-viestiketjussa sovellukset kommunikoivat tilaamalla ja julkaisemalla tietoa aihealueittain. Tilaajat ja julkaisijat voivat asettaa aihealueen tiedolle yksityiskohtaisia luokitteluja ja prioriteetteja (QoS). Tietoa voidaan suodattaa esimerkiksi sen elinajan tai päivitystaajuuden mukaan. (DDS-foundation [Viitattu 1.3.2019].)

2.2.4 ZeroMQ

ZeroMQ, tai lyhyemmin ZMQ, on sovellusten väliseen viestintään tarkoitettu vapaan lähdekoodin ohjelmistokirjasto. Kirjaston ohjelmointirajapinta muistuttaa perinteistä socket-rajapintaa, mutta tarjoaa siihen nähden huomattavan määrän viestintää monipuolistavia ja helpottavia ominaisuuksia, kuten viestien jonottamisen, aiheiden suodattamisen ja eri viestintämallit. ZeroMQ-socket-rajapinta mahdollistaa julkaisija–tilaaja-viestinnän (publish–subscribe), pyyntö–vastaus-viestinnän (request–response) tai putkistomallisen (pipeline) viestinnän. ZeroMQ ei vaadi välittäjää viestiketjun keskelle, vaan laitteet voivat keskustella suoraan keskenään. Libzmq-niminen ydinkirjasto on ohjelmoitu c++-ohjelmointikielellä, mutta sen päälle on rakennettu laajennuksia todella kattavasti muille ohjelmointikielille. Näitä ovat esimerkiksi C-kielelle czmq, Pythonille pyzmq ja C#-kielelle netmq. (ZeroMQ [Viitattu 2.3.2019].)



Kuva 8. ZeroMQ-julkaisija–tilaaja-viestiketju
(ZeroMQ [Viitattu 2.3.2019])

Kuvassa 8 on esitetty julkaisija–tilaaja-mallinen viestiketju. Viestiketjussa tilaajat yhdistyvät julkaisijaan ja tilaavat julkaisijalta tietyn aihealueen viestit. Tämän jälkeen julkaisija alkaa päivittämään julkaistavaa tietoa siihen yhdistyneille tilaajille. Sama laite voi olla sekä julkaisija että tilaaja. Yksi julkaisija voi julkaista viestejä useilla aihealueilla. Vastaavasti yksi tilaaja voi tilata usean aihealueen viestejä.

3 SOVELLUSTASON PROTOKOLLIEN VERTAILU

Kuten jo luvussa 2.2. todettiin, M2M-kommunikaation tekniikoita on olemassa paljon erilaisia ja oikean tekniikan valinta täytyy perustua toteutettavan sovelluksen vaatimuksiin. Tässä luvussa määritellään vaatimukset Epecin ohjausjärjestelmien välillä käytävälle M2M-kommunikaatiolle langattomassa lähiverkossa, sekä vertaillaan, mikä sovellustason tekniikka vastaa parhaiten näihin vaatimuksiin.

3.1 Vaatimukset

Opinnäytetyön alkuvaiheessa määriteltiin vaatimukset, joiden pohjalta mahdollisia M2M-tekniikoita voitiin lähteä vertailemaan. Vaatimukset perustuivat Epecin asiakkaiden tarpeisiin ja Epecin omiin näkemyksiin siitä, minkälaisiin käyttökohteisiin M2M-kommunikaatiota langattomassa lähiverkossa voitaisiin tulevaisuudessa käyttää. Opinnäytetyön tarkoituksena ei ole tutkia mahdollisia käyttökohteita, vaan selvittää ratkaisu, millä tämän hetken tarpeisiin sekä yrityksen näkemysten mukaisiin tulevaisuuden tarpeisiin pystytään vastaamaan.

Epecin WLAN-verkkoa tukevat 6000-sarjan laitteet, joiden välillä M2M-kommunikaatio on tarkoitus toteuttaa, on esitelty myöhemmin luvussa 4.2.

3.1.1 Arkkitehtuuri

Tällä hetkellä Epecin WLAN-laitteet tukevat pelkästään perinteistä WLAN-topologiaa, jossa on yksi tukiasema ja ulkopuoliset liittyjät yhdistyvät siihen. Tulevaisuudessa voi olla mahdollista, että laitteilla voidaan tehdä myös Mesh-topologian mukaisia verkkoja, joista kerrottiin aiemmin luvussa 2.1.1. Mesh-verkoille on tyypillistä, että verkkotopologia voi muuttua jatkuvasti. Verkkoon voi liittyä uusia jäseniä tai niistä voi irtautua jäseniä koska vain. M2M-kommunikointi tulee siis toteuttaa niin, että se toimii infrastruktuuriin perustuvan verkon lisäksi Mesh-topologian mukaisissa verkoissa.

Verkko voidaan muodostaa satunnaisesti kahden tai useamman laitteen välille. Verkko on joko infrastruktuuriin perustuva tai MANET-verkko. Tämä vaatimus rajaa valittavista M2M-kommunikaation toteutustavoista sellaiset, joissa kommunikaatio perustuu viestiketjun keskellä olevaan välittäjään. Välittäjäkeskeinen toteutustapa toimisi tapauksessa, joissa verkkotopologiassa on yksi palvelin, johon laitteet voivat yhdistyä.

3.1.2 Luotettavuus ja todentaminen

Työkoneiden välisessä kommunikoinnissa on tärkeää, että viesti pystytään toimittamaan määritetyssä ajassa, eikä tieto ole vanhentunutta. Tämä tarkoittaa vaatimuksena sitä, että tiedon vastaanottajan täytyy viestin sisällön lisäksi pystyä tunnistamaan, jos viesti ei tule perille halutussa ajassa. Verkossa olevien laitteiden täytyy myös pystyä tunnistamaan toinen toisensa, että tiedetään, mistä tieto on lähtöisin.

3.1.3 Vikasietoisuus

Langattoman lähiverkon yhteyden laatu voi heiketä, ja verkko saattaa olla epävakaa, kun toimitaan haastavissa olosuhteissa sekä pitkillä toimintaetäisyyksillä. Tässä yhteydessä vikasietoisuusvaatimuksella tarkoitetaan sitä, että toteutustavan mukaisen järjestelmän tulee selvitä verkossa, jossa laitteet saattavat välillä menettää yhteyden toisiinsa.

3.2 Vertailu

Luvussa 2.2 on esitelty toimintaperiaatteet neljästä erilaisesta M2M-kommunikaation sovellustason toteutustavasta: MQTT, CoAP, DDS sekä ZeroMQ. MQTT on kevyen viestirakenteen omaava välittäjäkeskeinen protokolla. CoAP on hyvin pienen laskentatehon omaaville laitteille suunniteltu HTTP-protokollaan ja REST-arkkitehtuuriin perustuva protokolla. ZeroMQ on sovellusten väliseen viestintään kehitetty ohjelmistokirjasto, ja DDS on M2M-kommunikaatioon tehty väliohjelmisto.

Näistä neljästä vain ZeroMQ ja DDS pystyivät vastaamaan tälle toteutukselle asetettuihin vaatimuksiin. MQTT-protokolla ei sovellu tähän toteutukseen, koska se perustuu viestiketjun keskellä olevaan välittäjään. Luvussa 3.1.1 on kerrottu, miksi välittäjäkeskeinen arkkitehtuuri ei sovellu tähän käyttötarkoitukseen. CoAP rajautui ulos valittavasta toteutustavasta, koska sen ominaisuudet, kuten REST-arkkitehtuuri ja HTTP-pohjainen protokolla, soveltuvat enemmänkin web-sovelluksille, kuin työkoneiden väliseen M2M-kommunikaatioon.

ZeroMQ ja DDS sopivat molemmat toteutustavaksi arkkitehtuuriin, jossa ei ole keskinäistä välittäjää. Lisäksi molemmilla tekniikoilla pystytään toteuttamaan viestien luotettavuudelle asetetut vaatimukset. ZeroMQ on täysin avoimen lähdekoodin ohjelmistokirjasto, kun taas DDS-protokollan toteutukset ovat enimmäkseen kaupallisia. Molemmille toteutustavoille löytyy internetistä hyvät dokumentaatiot sekä ohjelmointiesimerkit.

Koska ZeroMQ-kirjastosta on saatavilla avoimen lähdekoodin toteutus C-ohjelmointikielelle, sekä sen käyttöönotto vaikutti huomattavasti yksinkertaisemmalta kuin DDS-protokollan, valikoitui se toteutustavaksi, jolla lähdettiin tekemään testisovellusta M2M-kommunikaatiosta.

4 TOTEUTUSTAVAN TESTAUS

M2M-kommunikaatiosta tehtiin testisovellus käyttäen ZeroMQ-ohjelmistokirjastoa kommunikointiin Epecin 6000-sarjan laitteiden välillä. Testisovelluksella haluttiin todentaa kyseisen toteutustavan toimivuus Epecin 6000-sarjan laitteissa, sekä testata kuinka kyseinen toteutustapa suoriutuu langattomassa lähiverkossa. Tässä luvussa esitellään Epecin laitteet, jotka tukevat WLAN verkkoa, sekä käydään läpi M2M-kommunikaation testisovelluksen toteutus.

4.1 Epecin WLAN-laitteet

Epec Oy:n valmistamia WLAN-verkkoa tukevia laitteita ovat Epec 6107 sulautettu näyttö ja Epec 6200 -etäyhteysyksikkö. Laitteissa on WLAN-yhteys, USB-, RS-232-liitynnät, CAN-väylä ja Ethernet-liitynnät sekä gsm/umts-yhteys. Lisäksi laitteet mahdollistavat GPS/GLONASS-paikannuksen. Laitteet pystyvät toimimaan WLAN-verkossa asiakkaana (client) tai tukiasemana (access point). Laitteiden käyttöjärjestelmä on Linux-pohjainen. Ohjelmointi tapahtuu Codesys-version 3.5 -ohjelmointiympäristössä. (Epec [Viitattu 14.2.2019].)

Kuva 9. Epec 6107 sulautettu näyttö
(Epec [Viitattu 14.2.2019])

Kuva 10. Epec 6200 -etäyhteysyksikkö
(Epec [Viitattu 14.2.2019].)

Yläpuolella ovat kuvat Epec 6107 sulautetusta näytöstä sekä Epec 6200 -etäyhteysyksiköstä. Nämä laitteet poikkeavat toisistaan vain sillä, että 6200-etäyhteysyksikössä ei ole näyttöä. Vaikka 6200-yksikössä ei ole näyttöä, sille voidaan kuitenkin toteuttaa käyttöliittymä käyttäen web-yhteydellä toimivaa visualisointia. (Epec [Viitattu 14.2.2019].)

4.2 ZeroMQ-ohjelmistokirjaston kääntäminen

ZeroMQ-ohjelmistokirjaston kääntäminen Epecin 6000-sarjan laitteiden Linux-pohjaiselle käyttöjärjestelmälle tehtiin Cygwin-emulaattorilla. Ennen kirjaston kääntämistä, Cygwinille täytyi asentaa 6000-sarjan näyttöjen Arm-prosessorin työkaluketju, joka määritteli kääntöasetukset kyseiselle prosessorille. ZeroMQ-kirjaston lähdekoodit ladattiin Github-versionhallintasivustolta. Lisäksi näytölle käännettiin myös CZMQ-kirjasto, joka on ZeroMQ-kirjaston päälle rakennettu korkeampitasoinen ohjelmistokirjasto. CZMQ yksinkertaistaa ZeroMQ-kirjaston käyttöä entisestään, ja sisältää myös lisäominaisuuksia, kuten muiden ZeroMQ-solmujen etsinnän verkosta. Testisovelluksessa käytettiin kuitenkin pelkästään ZeroMQ-kirjaston funktioita, sillä tässä tapauksessa ne riittivät kirjaston testaamiseen.

4.3 Nanopb

ZeroMQ ei sisällä ratkaisua viestien sisältämän datan jäsentelyyn ja serialisointiin, joten siinä käytettiin avuksi Nanopb-ohjelmistokirjastoa.

Nanopb on C-kielinen toteutus Googlen protobuf-viestiformaatille. Nanopb-kirjastolla pystytään toteuttamaan viestien serialisointi ja serialisoidun tiedon muuntaminen takaisin alkuperäiseen formaattiin. (Kapsi [Viitattu 10.4.2019].)

Protobuff on joustava ja tehokas tapa serialisoida jäsenneltyä tietoa. Protobuf-viesti määritellään ”.proto”-päätteisessä tiedostossa, jonka jälkeen tiedostosta generoidaan protoc-generaattorilla funktiot, joilla voidaan lukea ja kirjoittaa serialisoituja viestejä. (Google [Viitattu 10.4.2019].)

```
// simple protobuff message

syntax = "proto2";

message WeatherMessage {
    required int32 Temperature = 1;
    required int32 Humidity = 2;
}
```

Kuva 11. Protobuf-viestin määritys ”.proto”-tiedostossa

Kuvassa 12 on määritelty hyvin yksinkertainen protobuf-viesti. Viestin nimi on ”WeatherMessage”, ja se sisältää kaksi kokonaisluku arvoa, ”Temperature” ja ”Humidity”. Viestin arvojen perässä olevat numerot ”1” ja ”2” määrittelevät kyseisen muuttujan sijainnin viestin rakenteessa. Esimerkiksi ”Temperature = 1” tarkoittaa, että ”Temperature”-niminen arvo sijaitsee viestissä rivillä yksi.

4.4 Testisovellus

Testisovelluksessa Epec 6107 -näyttö toimi WLAN-verkon tukiasemana, ja 6200-ohjausyksikkö yhdistyi siihen. Tukiaseman perustaminen ja siihen yhdistyminen toteutettiin Epecin valmiilla Codesys-kirjastolla. 6200-yksikön yhdistymisen jälkeen alkoi tukiasemana toiminut näyttö julkaista verkkoon kahta liukusäätimen arvoa, jotka 6200-yksikkö tilasi.

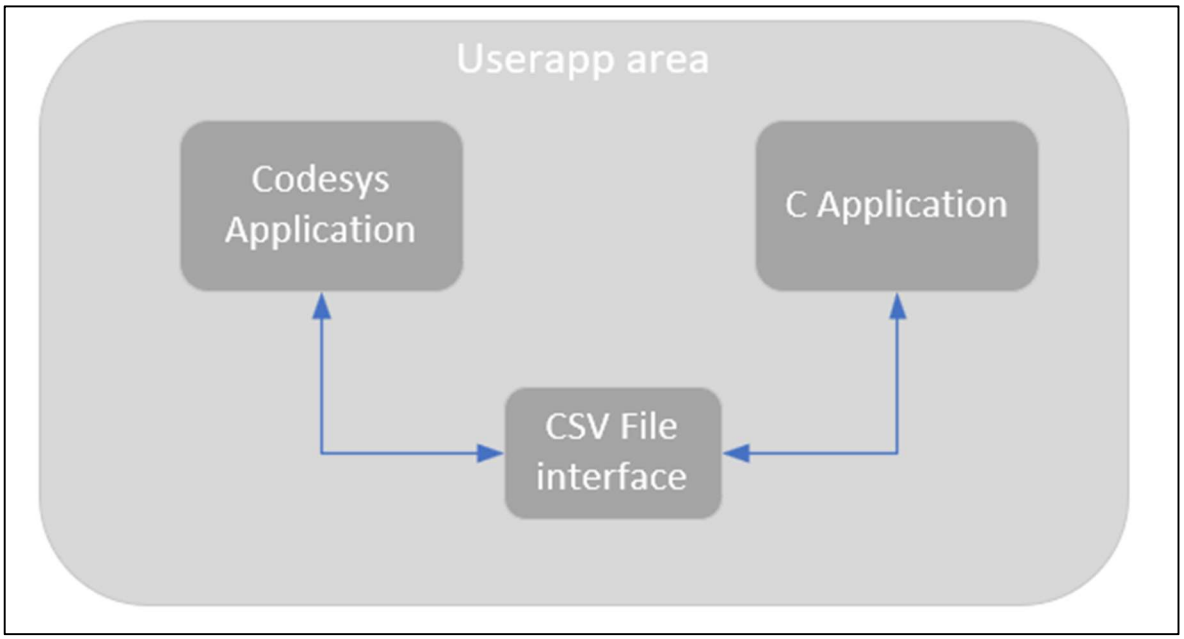
Kuva 12. Testisovelluksen laitteisto

Kuvassa 12 testisovelluksen laitteet, vasemmalla WLAN-tukiasemana toiminut 6107-näyttö ja oikealla WLAN-verkon asiakkaana toiminut 6200-etäyhteysyksikkö.

4.4.1 Sovelluksen ohjelmistoarkkitehtuuri

Kuten jo aiemmin tässä opinnäytetyössä on todettu, M2M-kommunikaation toteutukseen käytettiin ZeroMQ-ohjelmistokirjastoa, ja viestien serialisointi tapahtui Nanopb-kirjastolla. ZeroMQ ja Nanopb-toiminnallisuus toteutettiin C-kielellä ohjelmoidussa ohjelmassa.

Epec 6000 -sarjan laitteiden ohjelmointi tapahtuu Codesys-version 3.5 -ohjelmointiympäristössä, joten M2M-kommunikaation toteuttavan C-sovelluksen ja Codesys-sovelluksen välille tarvittiin jonkinlainen rajapinta. Tässä tapauksessa rajapinta toteutettiin csv-tekstitiedostoilla, joiden kautta Codesys-sovellus sekä M2M-kommunikaation toteuttava C-sovellus kommunikoivat.

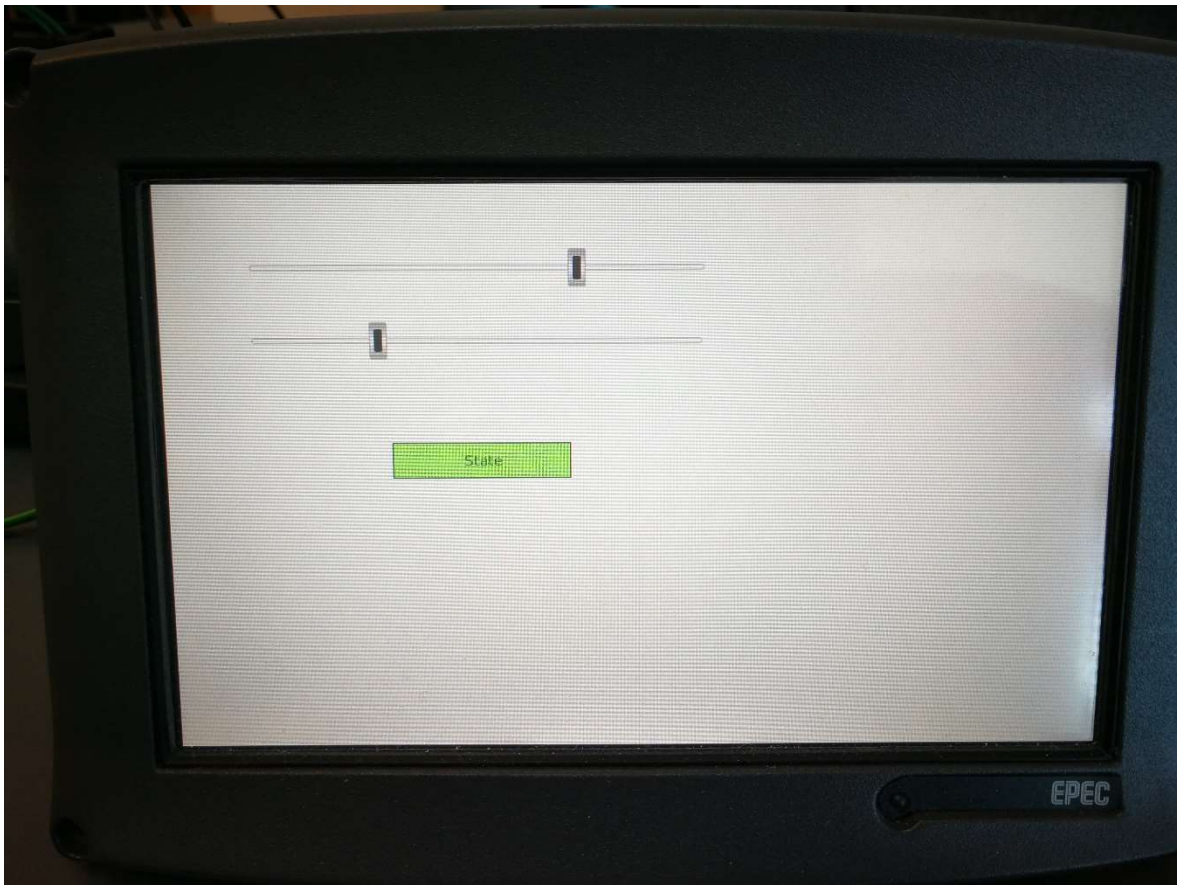


Kuva 13. Testisovelluksen ohjelmistoarkkitehtuuri

Kuvassa 11 on esitetty testisovelluksen ohjelmistoarkkitehtuuri. Codesys-sovellus ja C-sovellus kommunikoivat csv-tiedostorajapinnan kautta.

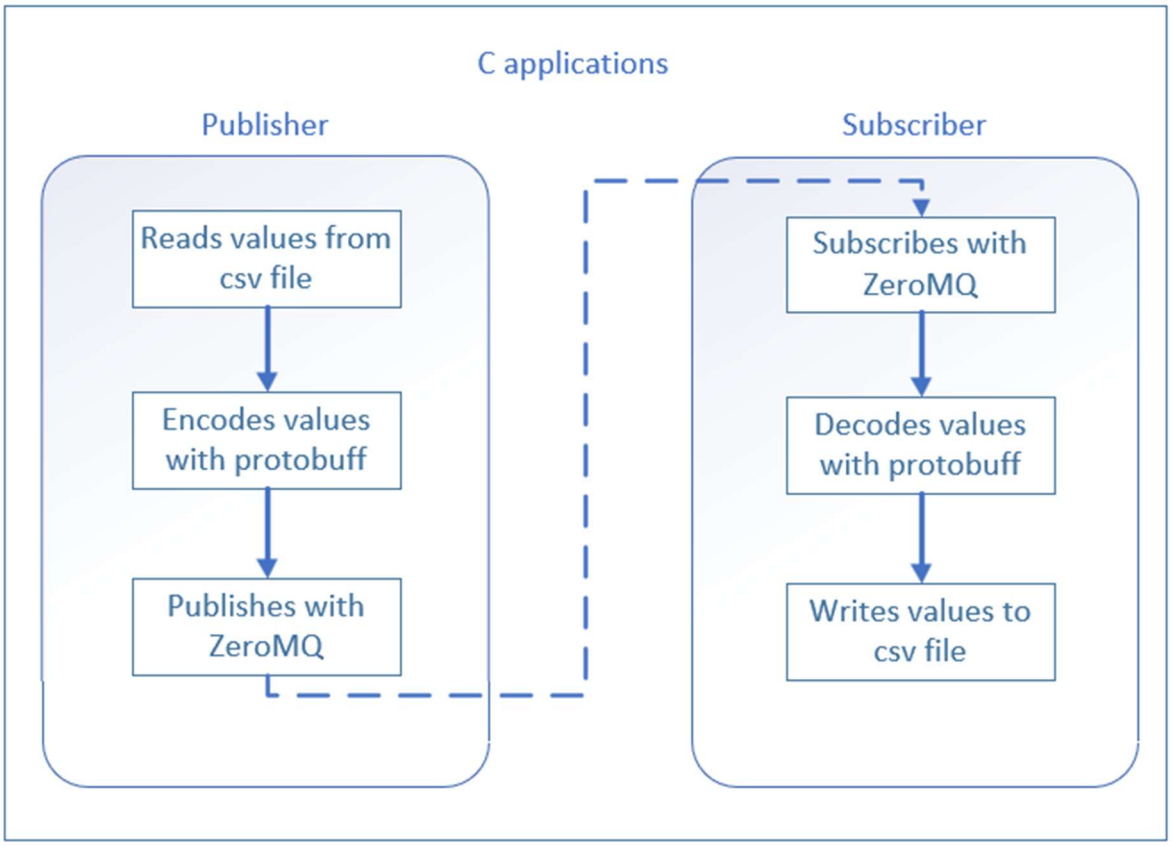
4.4.2 Sovelluksen viestiketjun kuvaus

Testisovelluksessa Epec 6107 -näytön Codesys-käyttöliittymään tehtiin kaksi liukusäädintä, joilla saatiin säädettyä kokonaislukuarvoja välillä 0–100. Codesys-sovelluksesta arvot kirjoitettiin csv-tiedostoon, jota M2M-kommunikaation toteuttava C-sovellus luki. C-sovelluksen luettua arvot csv-tiedostosta se serialisoi arvot protobuff binäärimuotoon nanopb-kirjastolla ja tämän jälkeen lähetti ne WLAN-verkkoon ZeroMQ-kirjaston avulla. Verkkoon yhdistyneessä Epec 6200 -ohjausyksikössä ollut oma C-sovellus tilasi julkaistut arvot, purki serialisoidun datan takaisin kokonaislukuarvoiksi ja kirjoitti kokonaislukuarvot omaan csv-tiedostoonsa. Lopulta 6200-yksikön Codesys-sovellus luki arvot C-sovelluksen kirjoittamasta csv-tiedostosta, ja näytti ne käyttöliittymässä kokonaislukuarvoina.



Kuva 14. Testisovelluksen arvojen säätimet Epec 6107 -näytöllä

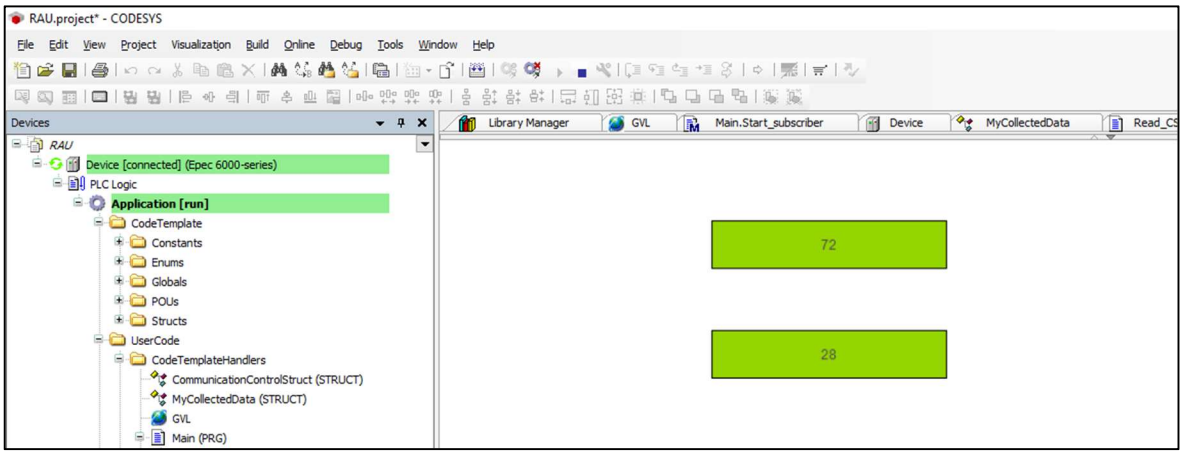
Testisovelluksen arvojen säätäminen tapahtui näytölle ohjelmoiduilla kahdella liukusäätimellä. Liukusäätimien alapuolella oli ”State” ruutu, joka esitti WLAN-tukiaseman tilaa.



Kuva 15. Julkaisijan ja tilaajan C-sovellusten rakenne

Kuvassa 15 on esitetty M2M-kommunikaation toteuttavien C-kielellä ohjelmoitujen sovellusten toimintakaavio. Vasemmalla on Epec 6107 -näytöllä olleen julkaisuja tehneen sovelluksen rakenne ja oikealla Epec 6200 -yksikön tilauksia hoitaneen sovelluksen rakenne.

Epec 6200 -yksikön C-sovelluksen purettua tilatun viestin takaisin viestin alkuperäiseen muotoon, eli binäärisestä muodosta kahdeksi kokonaislukuarvoksi, kirjoitettiin kokonaislukuarvot csv-tiedostoon. Lopulta kokonaislukuarvot luettiin csv-tiedostosta Codesys-sovellukseen, ja luetut arvot näytettiin jälleen käyttöliittymässä.



Kuva 16. Testisovelluksen arvojen lukeminen Epec 6200 -yksikössä

Epec 6200 -yksikön vastaanottamia arvoja pystyi lukemaan joko Codesys webvisu -käyttöliittymästä tai kirjautumalla sisään 6200-yksikön Codesys-ympäristöön. Kuvassa 16 on kirjauduttu sisään Codesys-ympäristöön. Käyttöliittymässä on kaksi ruutua, jotka näyttävät vastaanotettuja kokonaislukuarvoja. Ruutujen väri muuttui valkoisesta vihreäksi, kun viestien vastaanottaminen oli toiminnassa.

4.4.3 ZeroMQ-julkaisija ja -tilaaja

Tässä luvussa esitellään otteet C-sovellusten osista, joissa viestien julkaisu ja tilaus tapahtuu käyttäen ZeroMQ-ohjelmistokirjastoa.

```
30      // Prepare our context and publisher
31      void *context = zmq_ctx_new ();
32      void *publisher = zmq_socket (context, ZMQ_PUB);
33      int rc = zmq_bind (publisher, "tcp://*:5556");
34      assert (rc == 0);
35
```

Kuva 17. ZeroMQ-julkaisijan luominen

Julkaisija-sovelluksen alussa tehtiin ZeroMQ-konteksti, jolle luotiin ”ZMQ_PUB”-pistoke. Kun pistoke oli luotu, se kiinnitettiin johonkin tunnettuun porttiin, tässä tapauksessa porttiin 5556 ”zmq_bind”-funktiolla.

```
78          // Send message to all subscribers
79          zmq_send(publisher, topic, sizeof(topic), ZMQ_SNDMORE);
80          zmq_send(publisher, buffer, sizeof(buffer),0);
```

Kuva 18. ZeroMQ-viestin julkaisu

Viestin julkaisu tapahtui ”zmq_send()”-funktiolla. Funktiolle annettiin parametreina käytettävä julkaisija, julkaistava viesti, viestin koko sekä lopuksi viestin muut asetukset. Tässä tapauksessa viesti lähetettiin kaksiosaisena. Ensimmäinen osa sisälsi viestin aiheen ja jälkimmäinen osa Nanopb-kirjastolla serialisoidut kokonaislukuarvot. Viestin ollessa kaksiosainen ensimmäisessä osassa täytyi määritellä ”ZMQ_SNDMORE”-asetuksella, että viesti tulee vielä jatkumaan. Jälkimmäisen osan lopussa asetus ”0” määritteli, että viesti loppui siihen.

```
31      //  Socket to talk to server
32      void *context = zmq_ctx_new ();
33      void *subscriber = zmq_socket (context, ZMQ_SUB);
34      int rc = zmq_connect (subscriber, "tcp://192.168.10.2:5556");
35      assert (rc == 0);
```

Kuva 19. ZeroMQ-tilaajan luominen

Kuten julkaisijalle sovelluksen alussa, myös tilaajalle tehtiin konteksti. Kontekstille määriteltiin ”ZMQ_SUB”-pistoke, jolla yhdistyttiin ”zmq_connect()”-funktiolla porttiin, johon julkaisija julkaisi viestejä. Tässä tapauksessa Epec 6107 -näyttö toimi julkaisijana, joten Epec 6200 -yksiköllä olevaan tilaaja-sovellukseen määriteltiin 6107-näytön IP-osoite ja portti, johon 6107-näyttö julkaisi viestejä.

```
39      //  Set topic
40      char *filter ="10001";
41      rc = zmq_setsockopt (subscriber, ZMQ_SUBSCRIBE, filter, strlen (filter));
42      assert (rc == 0);
43      // set receive timeout
44      int timeout = 500;
45      rc = zmq_setsockopt (subscriber, ZMQ_RCVTIMEO, &timeout, sizeof (int));
46      assert (rc == 0);
```

Kuva 20. ZeroMQ-tilaaja-pistokkeen asetukset

ZeroMQ-pistokkeille pystyy määrittämään asetuksia ”zmq_setsockopt()”-funktiolla. Tilaaja-pistokkeelle määriteltiin tilattava aihe ”ZMQ_SUBSCRIBE”-asetuksella ja viestin vastaanottoajalle takaraja ”ZMQ_RCVTIMEO”-asetuksella.

```
86          int size = zmq_recv (subscriber, topic, 10, 0);
87          int size2 = zmq_recv (subscriber, buffer, 128, 0);
```

Kuva 21. ZeroMQ-viestin vastaanottaminen tilaaja-pistokkeella

Tilaaja vastaanotti kaksiosaisen viestin ”zmq_recv()”-funktiolla. Viestin ensimmäinnen osa sisälsi viestin aiheen, ja jälkimmäinen osa Nanopb-kirjastolla serialisoidut kokonaislukuarvot.

5 YHTEENVETO JA POHDINTA

Tämän opinnäytetyön tavoitteena oli selvittää toteutustapa langattomassa lähiverkossa käytävälle M2M-kommunikaatiolle Epec 6000-sarjan ohjausyksiköillä. Tavoitteena oli myös toteuttaa testisovellus, jossa todennetaan valitun toteutustavan toimivuus.

Opinnäytetyön alussa selvitettiin mahdolliset vaihtoehdot toteutukselle, jonka jälkeen valittiin toteutustapa, jolla testisovellus tehtiin. Vaatimusten perusteella tehdyssä vertailussa löytyi kaksi toteutustapaa, jolla M2M-kommunikaatio olisi järkevää toteuttaa, ZeroMQ sekä DDS. Näistä kahdesta ZeroMQ valikoitui toteutustavaksi, jolla testisovellusta lähdettiin tekemään. Myös DDS olisi varmasti ollut hyvä vaihtoehto, mutta tähän opinnäytetyöhön käytettävä aika ei ollut riittävä siihen, että molempia vaihtoehtoja olisi päästy kokeilemaan.

ZeroMQ-kirjastolla tehtiin testisovellus, jossa Epec 6107 sulautettu näyttö lähetti kahta kokonaislukuarvoa Epec 6200 -etäyhteysyksikölle WLAN-verkossa. ZeroMQ-kirjaston lisäksi testisovelluksessa käytettiin protobuf-viestiformaattia käyttävää Nanopb-kirjastoa viestien serialisointiin. Testisovellus osoitti, että ZeroMQ ja Nanopb yhdessä ovat hyvin toimiva ratkaisu M2M-kommunikaation toteuttamiseen Epec 6000 -sarjan laitteille.

Tämän opinnäytetyön selvityksen sekä toteutetun testisovelluksen toimivuuden pohjalta pystytään toteuttamaan ratkaisu Epecin 6000 -sarjan laitteiden väliseen M2M-kommunikaatioon langattomassa lähiverkossa.

LÄHTEET

Abbasi, I. 2018. A Review of Vehicle to Vehicle Communication Protocols for VANETs in the Urban Environment. [Verkkolehtiartikkeli]. Future Internet 2018 10 (2), 14. [Viitattu 11.2.2019]. Saatavana ProQuest -tietokannasta.

DDS foundation. Ei päiväystä. What is DDS?. [Verkkusivu]. Object Management Group. [Viitattu 1.3.2019]. Saatavana: https://www.dds-foundation.org/what-is-dds-3/

DDS foundation. Ei päiväystä. Where can i get dds? [Verkkusivu] Object Management Group. [Viitattu 2.3.2019]. Saatavana: https://www.dds-foundation.org/where-can-i-get-dds/

Eclipse. 2014. MQTT101 - How to get started with the lightweight IoT protocol. [Verkkojulkaisu]. Eclipse foundation. [Viitattu 6.2.2019]. Saatavana: https://www.eclipse.org/community/eclipse_newsletter/2014/october/article2.php

Epec. Ei päiväystä. Epec company. [Verkkosivu]. Epec Oy. [Viitattu 13.2.2018]. Saatavana: https://epec.fi/company/

Epec. Ei päiväystä. Epec products. [Verkkosivu]. Epec Oy. [Viitattu 14.2.2018]. Saatavana: https://epec.fi/products/

ETSI. 2013. TS 102 689. [Verkkojulkaisu]. European Telecommunications Standards Institute. [Viitattu 1.2.2019]. Saatavana: https://www.etsi.org/deliver/etsi_ts/102600_102699/102689/01.02.01_60/ts_102689v010201p.pdf

Google. Ei päiväystä. Protocol buffers. [Verkkosivu]. Google developers. [Viitattu 10.4.2019]. Saatavana: https://developers.google.com/protocol-buffers/

IETF. 2014. The Constrained Application Protocol (CoAP). [Verkkojulkaisu]. Internet Engineering Task Force (IETF). [Viitattu 15.2.2019]. Saatavana: https://tools.ietf.org/html/rfc7252

Kapsi. Ei päiväystä. Nanopb: Protocol Buffers with small code size. [Verkkosivu]. [Viitattu 10.4.2019]. Saatavana: https://jpa.kapsi.fi/nanopb/docs/index.html

Meng, Z., Zhipeng, W. & Gray, J. 2017. A Collaboration-Oriented M2M Messaging Mechanism for the Collaborative Automation between Machines in Future Industrial Networks. [Verkkolehtiartikkeli]. Sensors 2017 17(11). Saatavana: MDPI open access journals- tietokannasta.

Oasis. 2014. MQTT Version 3.1.1. [Verkkosivu]. OASIS Message Queuing Telemetry Transport (MQTT) TC. [Viitattu 6.2.2019]. Saatavissa: http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html

OpenDDS. Ei päiväystä. QoS Policy Usage. [Verkkosivu]. Object Computing. [Viitattu 1.3.2019] Saatavana: http://opendds.org/about/qosusages.html

U-blox. 2016. IP versus CoAP for IoT Communications. [Verkkojulkaisu]. U-blox. [Viitattu 15.2.2019]. Saatavana: https://www.u-blox.com/en/ip-versus-coap-iot-communications

Wei, H., Rykowski, J. & Dixit, S. 2013. WiFi, WiMAX and LTE Multi-Hop Mesh Networks: Basic Communication Protocols and Application Areas. [Verkkokirja]. New Jersey: John Wiley & Sons. [Viitattu 4.2.2019]. Saatavana: ProQuest Ebook Central -palvelusta. Vaatii käyttöoikeuden.

ZeroMQ. Ei päiväystä. Zguide. [Verkkojulkaisu]. iMatix. [Viitattu 2.3.2019]. Saatavana: http://zguide.zeromq.org/page:all