Katsaus sulautettujen järjestelmien ohjelmistoarkkitehtuureihin

Turun yliopisto
Tietotekniikan laitos
LuK-tutkielma
Tietojenkäsittelytiede
Joulukuu 2025
Hans Imberg

Turun yliopiston laatujärjestelmän mukaisesti tämän julkaisun alkuperäisyys on tarkastettu Turnitin OriginalityCheck -järjestelmällä.

TURUN YLIOPISTO
Tietotekniikan laitos

Hans Imberg: Katsaus sulautettujen järjestelmien ohjelmistoarkkitehtuureihin

LuK-tutkielma, 20 s.
Tietojenkäsittelytiede
Joulukuu 2025

Sulautettuja järjestelmiä on kaikkialla, mutta niiden ohjelmistoarkkitehtuurit ja taustalla olevat vaatimusmäärittelyt ovat monille vieraita. Tutkielman tavoitteena on tarjota korkean tason yleiskuva sulautettujen järjestelmien yleisimmistä arkkitehtuureista ja vaatimusmäärittelyistä.
Tutkielma toteutettiin kirjallisuuskatsauksena, jossa käsiteltiin 24 aineistoa vastatakseen tutkielman tutkimuskysymyksiin. Tutkimuskysymyksinä on selvitetty sulautetuissa järjestelmissä yleisimmät vaatimusmäärittelyt, ohjelmistoarkkitehtuurit ja näiden välinen yhteys.
Tuloksina aineistojen pohjalta on tunnistettu viisi yleisintä ei-funktionaalista vaatimusmäärittelyä, ainakin seitsämän arkkitehtuuria ja muita arkkitehtuurin tarkastultapoja. Lopuksi käsiteltyjen aineistojen pohjalta on muodostettu synteettinen näkemys, joka tiivistää vaatimusmäärittelyiden, arkkitehtuurien ja järjestelmäkerroksien väliset yhteydet.

Asiasanat: sulautetut järjestelmät, IoT, ohjelmistoarkkitehtuurit, vaatimusmäärittelyt, ohjelmistosuunnittelu

Sisällys

Kuvat

Taulukot

1 Johdanto

Sulautetut järjestelmät ja esineiden internet (engl. Internet of Things, IoT) ovat keskeinen osa nykypäivää. Sulautetuilla järjestelmillä tarkoitetaan pienikokoisia rajatuilla resursseilla toimivia laskennallisia laitteita, jotka tekevät tarkkaan rajattuja tehtäviä [1, s. 3]. Näiden järjestelmien asema on keskeinen ajoneuvoissa, kodinkoneissa, kuluttajaelektroniikassa, teollisuudessa ja terveydenhoidossa [2, s. 5]. Monille aiheeseen perehtymättömille saattaa kuitenkin olla haasteellista hahmottaa, miten näiden laitteiden ohjelmistot rakentuvat. Niinpä tämän tutkielman aiheena tarkastellaan näitä ohjelmistojen rakenteita kirjallisuuskatsauksen muodossa.

Tässä työssä sulautetuilla järjestelmillä tarkoitetaan myös IoT-järjestelmiä. IoT-järjestelmien katsotaan olevan sulautettuja järjestelmiä, mutta ne ovat lisäksi yhteydessä toisiin laitteisiin tai verkkoon viestintäprotokollien avulla [1, s. 14]. Vaatimusmäärittelyillä tarkoitetaan ominaisuuksia, joita laitteelta tai laitteen ohjelmistolta vaaditaan. Rajoituksilla tarkoitetaan teknisiä rajoituksia, jotka ovat lähtöisin sulautettujen järjestelmien rajallisista laitteistoresursseista.

Sulautetuilla järjestelmillä on korkeammat laatu- ja luotettavuusvaatimukset verrattuna muihin järjestelmiin. Esimerkiksi jos ajoneuvon moottorinhallintajärjestelmä tai terveydenhoidossa käytettävä laite lakkaa toimimasta kriittisessä vaiheessa, seuraukset voivat olla vakavia [2, s. 4]. Vahvan teknisen perustan aikaansaaminen sulautetulle järjestelmälle edellyttää kehitystiimin jäsenien ymmärrystä järjestelmän arkkitehtuurista. Arkkitehtuurilla tarkoitetaan korkean tason yleistys-

tä järjestelmästä, sisällyttämättä tietoa yksityiskohtaisesta toteutuksesta [2, s. 4]. Arkkitehtuuritason tietoa ilmaistaan rakenteina, joita ovat esimerkiksi moduulit, kerrokset ja komponentit [2, s. 5]. Organisaatioiden näkökulmasta arkkitehtuurisen ajattelutavan omaksuminen osaksi yrityksen strategiaa voi olla hyödyksi. Siksi ohjelmistoarkkitehdin osallistuminen aikaisessa vaiheessa tuotesuunnittelua voi auttaa tunnistamaan riskejä, uusia parempia ominaisuuksia ja mahdollisuuksia ohjelmiston uudelleenkäytölle [3]. Arkkitehtuurien osalta tämä tutkielma ei kuitenkaan käsittele laitteistoarkkitehtuureja, sillä tarkastelun kohteena ovat ohjelmistoon liittyvät arkkitehtuuriset ratkaisut.

Tutkielman tarkoituksena on tarjota korkean tason yleiskuva sulautetuille järjestelmille tyypillisistä vaatiumusmäärittelyistä ja ohjelmistoarkkitehtuureista. Lopputuloksena lukija voi ymmärtää paremmin sulautettuja järjestelmiä ja käyttää tuloksia parempien arkkitehtuuristen valintojen tekemiseen. Tutkimuskysymykset on määritelty näiden tavoitteiden pohjalta seuraavalla tavalla:

TK1: Minkälaisia ovat sulautetuille järjestelmille tyypilliset vaatimusmäärittelyt?
TK2: Mitkä ovat sulautetuille järjestelmille yleisimpiä arkkitehtuureja?
TK3: Minkälainen yhteys on vaatimusmäärittelyiden ja arkkitehtuurien välillä?

Kirjallisuuskatsauksen aineistoina on käytetty IEEE- ja ACM-tietokantoja sekä kahta aihetta käsittelevää kirjaa. Tiedonhakumenetelmissä on käytetty kolmeen kategoriaan asettuvia hakusanoja. Hakusanoista on koostettu seuraavanlainen hakulauseke, jota on käytetty IEEE- ja ACM-hakukannoissa, ilman hakusuodattimia:

1. ("embedded system*"OR "embedded software"OR "IoT"OR "internet of things"OR "embedded computer systems")
AND
2. ("software architect*"OR "architectu* evaluation"OR "architectu* knowledge"OR "architectu* decisio*"OR "architectu* design"OR "architectu* options"OR "architecture decision-making")
AND
3. ("software requirement*"OR "requirement* engineering"OR "software design"OR "functional requirements"OR "non-functional requirements"OR “taxonomy”)

Ensimmäinen hakulauseke perustuu tutkielman aiheeseen ja koskee sulautettuihin järjestelmiin ja IoT-järjestelmiin liittyviä hakusanoja. Toinen hakulauseke perustuu toiseen tutkimuskysymykseen, sisältäen ohjelmistoarkkitehtuureihin liittyviä hakusanoja. Kolmas hakulauseke perustuu ensimmäiseen tutkimuskysymykseen sekä täydentäviin hakusanoihin, sisältäen vaatimusmäärittelyyn ja ohjelmistosuunnitteluun liittyviä hakusanoja.

IEEE-tietokannasta saatiin 425 hakutulosta ja ACM Full-Text Collection -tietokannasta 129 hakutulosta, yhteensä 554 hakutulosta. Tiedonhakumenetelmät perustuivat kolmeen aineiston karsintavaiheeseen. Ensimmäisenä otsikon ja abstraktin perusteella tehtävä karsinta, jossa karsinta tehtiin seuraavin perustein:

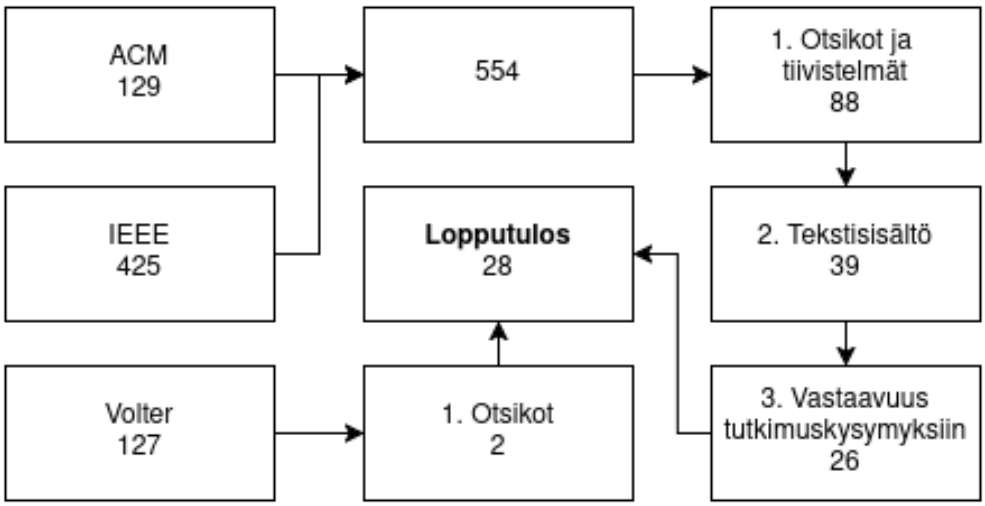
- Jos materiaali ei koske riittävän yleisellä tasolla sulautettujen järjestelmien ohjelmistoarkkitehtuureita ohjelmistosuunnittelun näkökulmasta.
- Jos materiaali koskee jotain tiettyä kapean alueen ongelmaa, menetelmää, arviointimenetelmää, uutta konseptia tai metodologiaa tai kapeaa sovelluskohdetta.
- Jos materiaali ei muuten liity aiheeseen tai tutkimuskysymyksiin.

Ensimmäisen aineistokarsinnan lopputuloksena 466 materiaaleista karsittiin ja 88 jäi jäljelle. Toisessa vaiheessa käytiin läpi materiaalien tekstisisällöt, joista 39 karsittiin ja jäljelle jäi 49. Aineistojen karsinta toisessa vaiheessa tehtiin seuraavin perustein:

- Sisältö ei vastannut aiheeseen tai tutkimuskysymyksiin (31 karsittu)
- Koko sisältöä ei ollut saatavilla (12 karsittu)
- Sisältö oli teknisesti yksityiskohtaista eikä vastannut tutkielman yleistettävyyskriteeriin (6 karsittu)

Kolmannessa aineiston karsinnassa vertailtiin jäljelle jääneitä materiaaleja keskenään ja niiden vastaavuutta tutkielman tutkimuskysymyksiin. Vastaavuudet arvioitiin pisteyttämällä ne vaatimusmäärittelyyn, arkkitehtuuriin ja yleiseen soveltuvuuteen perustuen. Arvioidut pisteytykset laskettiin yhteen ja aineistot, jotka saivat vähemmän kuin kuusi pistettä karsittiin. Materiaaleista 13 karsittiin ja 26 jäi jäljelle.

Tutkielman taustatietona käytettävät tekstikirja-aineistot on haettu Turun yliopiston Volter-kirjaston hakukannasta nimikesisällöllä "embedded architecture", aineistotyyppinä kirjat ja kielenä englanti. Tuloksia saatiin 127 kappaletta, joista kaksi valittiin tutkielmaan ja loput karsittiin otsikoiden perusteella. Tiedonhakumenetelmien lopputuloksena käytiin kokonaisuudessaan läpi 681 lähdemateriaalia, joista lopulliseksi aineistoksi valittiin 28 lähdemateriaalia (Kuva 1.1).



Kuva 1.1: Aineiston yhteenveto.

Johdannon jälkeen (Luku 2) tutkielmassa tarkastellaan tiiviisti vaatimusmäärittelyihin, rajoituksiin ja arkkitehtuureihin liittyvää taustatietoa tekstikirja-aineistoihin, yleiseen materiaaliin ja yhteen verkkolähteeseen perustuen. Tämän jälkeen (Luku 3) käsitellään varsinaisen kirjallisuuskatsauksen tulokset, joissa käsitellään IEEE- ja ACM-hakukantojen tutkimuskysymyksiä koskevat 24 aineistoa. Lopuksi esitetään (Luku 4) kirjallisuuskatsauksen yhteenveto.

2 Sulautetetut järjestelmät

2.1 Vaatimusmäärittelyt ja rajoitteet

Ohjelmistojen vaatimusmäärittelyt on jaoteltu kahteen kategoriaan: funktionaalisiin ja ei-funktionaalisiin vaatimuksiin. Funktionaaliset vaatimukset määrittävät ominaisuudet, joita järjestelmän pitää toteuttaa saavuttaakseen liiketoiminta- ja käyttäjätarpeet. Ei-funktionaaliset vaatimukset tarkoittavat vaatimuksia, jotka määrittävät miten järjestelmän toimii, painottaen enemmän käyttäjäkokemusta. Ei-funktionaalisia ominaisuuksia ovat ovat esimerkiksi: suorituskyky, tietoturva, luotettavuus, skaalattavuus, ylläpidettävyys ja siirrettävyys [4][5]. Ei-funktionaaliset vaatimukset voidaan katsoa rajoitteina, jotka määrittävät miten funktionaaliset vaatimukset tulisi saavuttaa [6].

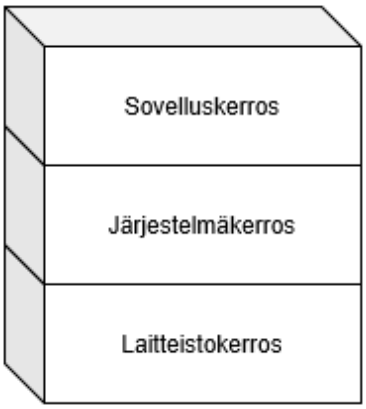
Sulautetut järjestelmät eroavat monin tavoin muista järjestelmistä. Suorituskykyvaatimukset ovat hallitsevassa osassa funktionaalisiin vaatimuksiin nähden, sillä järjestelmiltä vaaditaan lähes optimaalista suorituskykyä, reaaliaikaisuutta, luotettavuutta, tietoturvaa ja hajautuneisuutta. Samanaikaisesti sulautetuilla laitteilla on rajatut resurssit laskennassa, välimuistin määrässä ja virrankulutuksessa. [7]

Funktionaalisten vaatimusten saavuttaminen voi vaatia kompromisseja, joiden syynä on laitteiston resurssirajoitukset. Esimerkiksi laitteen flash-muisti, eli sulautetun laitteen tallennusmuistikapasiteetti saattaa loppua uuden ominaisuuden käyttöönotossa, RAM (random access memory) muistin määrä saattaa loppua komplek-

sien tietorakenteiden käsittelyssä, prosessori saattaa olla liian hidas tai akkutoiminen laite saattaa vaatia matalaa energiankulutusta. [1, s. 8]

2.2 Arkkitehtuurit

Sulautetuille järjestelmille yleisiä arkkitehtuurisia ominaisuuksia voidaan tarkastella seuraavan mallin avulla (Kuva 2.1). Malli koostuu kolmesta kerroksesta: sovelluskerros (engl. application layer), järjestelmäkerros (engl. system layer) ja laitteistokerros (engl. hardware layer). Kaksi ylintä kerrosta koskevat järjestelmän ohjelmistoja ja alin kerros laitteistoa [2, s. 10]. Tässä tutkielmassa tarkastellaan maillin kahta ylintä kerrosta, eli sovelluskerroksen ja järjestelmäkerroksen ohjelmistoarkkitehtuureja.



Kuva 2.1: Sulautettujen järjestelmien malli. [2, s. 10]

Keskimmäisen järjestelmäkerroksen tehtävänä on tukea sovelluskerroksen toimintaa tarjoamalla korkeamman abstraktion rajapinta laitteistokerroksen hallintaan. Tähän kerroksen kuuluvat esimerkiksi ajurit, käyttöjärjestelmät ja väliohjelmistot (eng. middleware) [2, s. 315-316]. Sovelluskerroksen ohjelmisto määrittää sulautetun järjestelmän varsinaisen funktionaalisen käyttötarkoituksen ja käsittelee suurimman osan varsinaisesta vuorovaikutuksesta laitteen käyttäjien kanssa [2, s. 456].

3 Kirjallisuuskatsauksen tulokset

3.1 Materiaaliluokitukset

Kirjallisuuskatsauksen aineistot (Taulukko 3.1) on lajiteltu yleisiin (Y) ja soveltaviin (S) materiaaleihin. Sisällöltään (Sis.) materiaalit on luokiteltu kolmeen kategoriaan: yleiseksi taustatiedoksi (Y), vaatimusmäärittelyihin (V) tai arkkitehtuureihin (A). Sisällön luokitteluissa on kuitenkin päällekkäisyyksiä ja siksi kyse on enemmän painotuksista kuin sisällön tarkoista rajauksista. Lisäksi on arvioitu aineistojen vastaavuus vaatimusmäärittelyihin (Req), arkkitehtuureihin (Arc) ja tutkielman yleiseen aiheeseen (Sov.) pisteytyksillä. Lopuksi taulukossa on yhteen lasketut kokonaispisteet (Kok.). Pisteytykset on tehty nollan ja viiden välillä siten, että viisi pistettä tarkoittaa eniten asiaan soveltuvaa sisältöä ja nolla vähiten.

Käsiteltäviä soveltavia materiaaleja on 12 kappaletta ja yleisiä 14. Aiheeltaan yleiseen taustatietoon painottuvia on kaksi kappaletta, vaatimusmäärittelyihin painottuvia kahdeksan ja arkkitehtuureihin painottuvia 16 kappaletta. Tässä luvussa on käsitelty vaatimusmäärittelyihin ja arkkitehtuureihin liittyvät 24 aineistoa. Ensin vaatimusmäärittelyihin liittyvät materiaalit, aloittaen sovelluskohteista ja yleisistä materiaaleista. Tämän jälkeen arkkitehtuurit samassa järjestyksessä. Lopuksi tarkastellaan vaatimusmäärittelyiden ja arkkitehtuurien välistä yhteyttä käsiteltyihin aineistoihin perustuen.

Taulukko 3.1: Aineistotaulukko. Lyhenteet on selitetty alaluvussa 3.1.

Aineisto	**S/Y**	**Kohde / kuvaus**	**Req**	**Arc**	**Sis.**	**Sov.**	**Kok.**
[8]	S	Vedenalaiset laitteet	3	2	V	3	8
[9]	S	Ajoneuvon valot	4	3	V	4	11
[10]	Y	Ohjelmistosuunnittelu	0	5	A	5	10
[11]	S	Älytehtaat	2	2	A	2	6
[12]	S	Akunhallinta	2	2	A	2	6
[13]	Y	Arkk. taksonomia	0	5	A	5	10
[14]	S	Ajoneuvot	3	4	A	3	10
[15]	S	Miehittämätön ilmailu	2	2	A	2	6
[16]	S	Älyrakennukset	2	2	A	2	6
[17]	S	Lento-ohjelmistot	2	2	A	2	6
[18]	S	Terveysteknologia	1	3	A	3	7
[19]	Y	Kehysvalinta	3	0	V	3	6
[20]	Y	Komponenttipoh. arkk.	0	3	A	3	6
[21]	Y	Vaatimuksista ja arkk.	4	5	A	5	14
[22]	Y	Vaatimuksista	4	0	V	3	7
[23]	Y	Älykaupungeista	0	5	A	4	9
[24]	Y	IoT-suunnittelu	0	5	A	5	10
[25]	Y	pilvi- ja reunalask.	4	0	V	3	7
[26]	S	Maatalousrobotiikka	3	3	A	3	9
[27]	S	Maatalous	4	4	A	4	12
[28]	Y	Suunnittelumallit	1	5	A	4	10
[5]	Y	Teolliset järjestelmät	4	0	V	3	7
[6]	Y	Mallipohjainen lähest.	3	0	V	3	6
[29]	S	Mainlux	3	1	V	3	7

3.2 Yleiset vaatimusmäärittelyt

Tässä osiossa käsitellään materiaaliluokituksen (Taulukko 3.1) mukaiset vaatimusmäärittelyihin (V) liittyvät materiaalit. Käsiteltäviä materiaaleja on kahdeksan kappaletta, joista suurin osa käsittelee ei-funktionaalisia vaatimusmäärittelyitä. Tarkastelua on voitu tapauskohtaisesti rajata ei-funktionaalisiin paremman yleistettävyyden saavuttamiseksi sekä vastaamaan paremmin tutkimuskysymyksiin. Ensin tarkastellaan ei-funktionaalisia vaatimuksia ja lopuksi yleisellä tasolla funktionaalisiin vaatimusmäärittelyihin vaikuttavia tekijöitä.

3.2.1 Sovelluskohteet

Tässä käsitellään materiaaliluokituksen (Taulukko 3.1) aineistot, jotka ovat samanaikaisesti soveltavia (S) ja vaatimusmäärittelyyn (V) liittyviä. Käsiteltäviä materiaaleja on kolme kappaletta.

Soveltaviin vaatimusmäärittelyihin liittyvissä materiaaleissa tunnistettiin keskeisiä ja odotettavia teemoja. Näitä olivat esimerkiksi modulaarisuuden, päivitettävyyden ja luotettavuuden ei-funktionaaliset vaatimukset [8]. Materiaaleista kaikissa kolmessa mainittiin myös turvallisuudesta hieman eri näkökulmista. Näitä olivat turvallisuuskriittinen näkökulma ajoneuvoissa [9], tietoturvallisuus [29] sekä jo edellä mainittu luotettavuuden näkökulma [8]. Ajoneuvoja käsittelevässä sovelluskohteessa tunnistettiin rajoitteita, jotka johtuivat ajoneuvojen ristikkäislinkitetyistä järjestelmistä ja tuotannon yksikköhintojen minimoinnista. Budjettirajoitteet johtivat rajoituksiin laitteistoresursseissa ja ajoituksessa (reaaliaikaisuudessa) [9]. Arkkitehtuurisesti merkittävien vaatimusten tunnistaminen todettiin myös tärkeänä vaiheena järjestelmän suunnittelua [29].

3.2.2 Yleiset materiaalit

Tässä käsitellään materiaaliluokituksen (Taulukko 3.1) aineistot, jotka ovat samanaikaisesti yleisiä (Y) ja vaatimusmäärittelyyn liittyviä (V). Näitä käsiteltäviä materiaaleja on viisi kappaletta.

Yleisiä vaatimusmäärittelyitä koskevissa materiaaleissa nousi esiin muutamia jo soveltavissa materiaaleissa havaittuja teemoja. Näitä olivat reaaliaikaisuuteen, tietoturvaan, yksityisyyteen ja laajennettavuuteen liittyvät seikat [19]. Lisäksi nousi esiin myös uudelleenkäytettävyys ja luotettavuus [22]. 2020 tehdyssä kyselytutkimuksessa kysyttiin ohjelmistoarkkitehdeilta IoT-järjestelmien käyttöönotosta pilvipalveluissa. Kyselytutkimuksessa tärkeimpiä tunnistettuja ajureita olivat luotettavuus ja tehokkuus, hinta, järjestelmän koko ja tietoturva [25]. Vaatimuslähtöistä suunnittelua ja arkkitehtuurin mallinnusta tarkastelevassa näkökulmassa mainittiin taas arkkitehtuurianalyysi ja kuvauskielellä (engl. Architecture Analysis & Description Language, AADL) tehtävän analyysin helpottavan reaaliaikaisuuden, luotettavuuden, datan laadun, resurssienkäytön ja tietoturvallisuuden arviointia [5].

Mallipohjaista lähestymistapaa kartoittavassa konferenssipaperissa taas jaoteltiin vaatimusmäärittelyt kolmeen kategoriaan: luotettavuusvaatimuksiin, suorituskykyvaatimuksiin ja tietoturvallisuuteen. Luotettavuutta koskevassa osiossa esitettiin lyhenne RAMS vaatimukset, joka sisältää luotettavuuden (engl. reliability), saatavuuden (engl. avaibility), ylläpidon (engl. maintainance) ja turvallisuuden (engl. security). Suorituskykyvaatimukset katsottiin avainasemassa oleviksi vaatimuksiksi persutavanlaatuisten resurssirajoitteden takia sulautetuissa järjestelmissä. [6]

Funktionaalisesta näkökulmasta katsottuna sulautetuille järjestelmille on tunnistettu useita keskeisiä ohjelmistoarkkitehtuurin suunnitteluun vaikuttavia piirteitä. Näistä keskeisimpiä ovat esimerkiksi tunnistaminen sensoreilla, laitteen profilointi (esim. sijainti, laiteidentiteetti, sensorien tila), laitteen hallinta, tiedon varastointi ja analytiikka sekä fyysisinä elementteinä kytkimet ja toiset laitteet [19].

3.2.3 Vaatimusmäärittelyiden yhteenveto

Alla (Taulukko 3.2) on yhteenveto yleisimmistä sulautettujen järjestelmien ei-funktionaalisista vaatimusmuksista, joita havaittiin käsitellyissä aineistoissa.

Taulukko 3.2: Yhteenveto tunnistetuista vaatimusmäärittelyistä.

Vaatimus	**Materiaali**
Modulaarisuus/ uudelleenkäytettävyys	[8][22]
Päivitettävyys/ skaalattavuus/ ylläpito	[8][29][19][22][6]
Luotettavuus/ turvallisuus	[8][9][22][25][5][6]
Suorituskyky/ reaaliaikaisuus/ latenssi	[29][9][19][25][5][6]
Tietoturva/ yksityisyys	[29][19][25][5][6]

Taulukossa on tiivistetty aineistoissa mainittuja samankaltaisia vaatimusmäärittelyitä samoihin kenttiin. Yleisimpinä vaatimuksina tunnistettiin suorituskyky ja reaaliaikaisuus (6), luotettavuus (5), päivitettävyys ja skaalautuvuus (5), tietoturva (5) ja modulaarisuus (2).

3.3 Yleiset arkkitehtuurit

Tässä alaluvussa käsitellään materiaaliluokituksen (Taulukko 3.1) mukaisesti kaikki arkkitehtuureihin (A) liittyvät materiaalit. Käsiteltäviä materiaaleja on yhteensä 16 kappaletta, aloittaen ensin soveltavista (S) materiaaleista, joita on yhdeksän kappaletta ja siirtyen tämän jälkeen yleisiin materiaaleihin.

3.3.1 Sovelluskohteet

IoT-älytehtaan ratkaisua tarkastelevassa materiaalissa hyödynnettiin mikropalveluarkkitehtuuria järjestelmän funktionaalisten vaatimusten toteuttamiseen. IoT-järjestelmässä hyödynnettiin MQTT:tä (Message Queuing Telemetry Transport) viestintäprotokollana. Järjestelmä suunniteltiin jakamalla se useisiin mikropalveluihin, kuten varaston seurantaan, tilauksille, NFC-tunnistimille (near-field communication) ja työpisteiden statusviestintään. [11]

Akunhallintajärjestelmissä on sovellettu kerrostettua arkkitehtuurista mallia, joka on jaoteltu ohjelmistoarkkitehtuurien osalta kolmeen kerrokseen. Alimpana kerroksena oleva ohjelmiston taustalevy (engl. Software Backplane) vastaa datan käsittelystä sekä akuston ja järjestelmän tilan hallinnasta. Keskimmäinen kerros käsittelee sovelluspalveluita, kuten ajonaikaisia, sekä ylläpito- ja monitorointipalveluita. Ylimpänä kerroksena toimii varsinainen sovelluskerros, joka toimii vuorovaikutuksessa alempien kerroksien kanssa. [12]

Ajoneuvojärjestelmiä takastelevassa näkökulmassa ohjelmistoarkkitehtuurin suunnittelua lähestyttiin funktionaalisten vaatimuksien näkökulmasta ja niiden toteutuksesta loogisena osiona [14]. Miehittämättömissä ilma-aluksissa ja älyrakennuksissa molemmissa hyödynnettiin taas palvelu-orientoitunutta arkkitehtuuria (engl. service-oriented architecture, SOA) [15][16].

Terveysteknologisten sovelluskohteiden standardiarkkitehtuureja tarkastelevassa kirjallisuuskatasuksessa tunnistettiin viisi yleistä terveydenhuoltoon erikoistunutta arkkitehtuuria. Näitä olivat mallipohjaiset, palvelu-orientoituneet, modulaariset, elämän kannalta kriittiset ja avustetun elämisen arkkitehtuurit [18]. Kahdessa maatalouden sovelluskohteessa hyödynnettiin myös kerrosarkkitehtuurista näkökulmaa [26], joista toisessa käytettiin sovelluskerroksessa MVC-arkkitehtuuria (Model-View-Controller) [27].

3.3.2 Yleiset materiaalit

Tässä käsitellään materiaaliluokituksen (Taulukko 3.1) aineistot, jotka ovat samanaikaisesti yleisiä (Y) ja arkkitehtuuriin liittyivä (A). Näitä on yhteensä seitsemän kappaletta.

Sulautettujen järjestelmien suunnittelua ja testausta käsittelevässä materiaalissa tarkasteltiin ohjelmistosuunnittelua kerrosarkkitehtuurisesta näkökulmasta. Keskeisinä arkkitehtuurisina osina mainittiin reaaliaikainen käyttöjärjestelmä (engl. real-time operating system, RTOS), laiteajurit ja verkkoviestintä [10]. RTOS mainittiin myös sulautettujen järjestelmien ohjelmistoarkkitehtuureja koskevassa taksonomiassa ja tämän lisäksi käsiteltiin kuusi muuta arkkitehtuuria. Näitä olivat no-OS (no-operating system), suorituksenaikaiset (engl. language-runtime), kokonaisen käyttöjärjestelmän (full-OS) sekä sovellus-, palvelin-, ja konttikäyttöjärjestelmälliset arkkitehtuurit [13]. Komponenttipohjaisia malleja tarkastelevassa materiaalissa taas käytettiin väliohjelmistoja sisältävää kerrosarkkitehtuurista tarkastelutapaa [20].

Vuonna 2003 julkaistussa artikkelissa tutkittiin sulautettujen järjestelmien suunnittelua yrityksissä. Tutkimuksen tuloksena muodostettiin hierarkkinen vaatimuslähtöinen näkökulma laajempien arkkitehtuurien suunnitteluun. Tässä hierarkkisessa mallissa on kolme kerrosta. Jokaisessa kerroksessa voi olla useampia vaatimusmäärittelyiden pohjalta suunniteltuja järjestelmäinstansseja ja näiden alijärjestelmiä, jotka koostuvat komponenteista. [21]

Älykaupunkien arkkitehtuureja tarkastelevassa materiaalissa tunnistettiin yhdeksän arkkitehtuuria. Suurin osa näistä oli hyvin alakohtaisia, mutta yhtenä yleisenä arkkitehtuurina mainittiin SOA [23]. IoT-suunnittelumalleista (engl. design pattern) tehdyssä kirjallisuuskatsauksessa todettiin, että IoT-järjestelmien ohjelmistot suunnitellaan usein perinteisten arkkitehtuurien ja suunnittelumallien avulla, jotka eivät ole erikseen IoT-järjestelmille räätälöityjä. Näitä katsauksessa tunnistettuja arkkitehuurisia suunnittelumalleja olivat esimerkiksi asiakas-palvelin-arkkitehtuuri,

SOA ja MVC. Lisäksi mainittiin myös esimerkki mikropalveluarkkitehtuurista IoT-järjestelmässä [24]. Toisessa arkkitehtuurisia suunnittelumalleja käsittelevässä materiaalissa mainittiin myös asiakas-palvelin-arkkitehtuuri ja MVC [28].

3.3.3 Arkkitehtuurien yhteenveto

Alla (Taulukko 3.3) on yhteenveto yleisimmistä sulautettujen järjestelmien arkkitehtuureista, joita havaittiin käsitellyissä aineistoissa.

Taulukko 3.3: Yhteenveto tunnistetuista ohjelmistoarkkitehtuureista.

Arkkitehtuuri	**Materiaali**
SOA	[15][16][18][24][23]
MVC	[27][24][28]
Mikropalvelu-arkkitehtuuri	[11][24]
Modulaarinen	[18]

Arkkitehtuuri	**Materiaali**
RTOS	[10][13]
Kerrosarkk-itehtuuri	[12][26][27][10][20]
Asiakas-palvelin-arkkitehtuuri	[24][28]
Komponentti-pohjainen, väliohjelmistot	[20]

3.4 Analyysi: vaatimusten ja arkkitehtuurien suhde

Käsitellyt materiaalit ovat vaihtelevia ja niissä on paljon eroja sovelluskohteesta sekä näkökulmasta riippuen. Vastaavanlaista tutkimusta tai suoraan tutkielman tutkimuskysymyksiin vastaavaa materiaalia ei ollut saatavilla. Tuloksista voidaan kuitenkin tunnistaa toistuvia piirteitä ja yleisiä havaintoja, joita on käsitelty tässä alaluvussa. Ensin käsitellään vaatimusten ja arkkitehtuurien välistä yhteyttä ja lopuksi ehdotetaan tiivistettyä näkökulmaa, perustuen käsiteltyihin materiaaleihin.

Käsitellyissä aineistoissa funktionaaliset ominaisuudet näyttivät painoittuvan

enemmän sulautetun mallin (Kuva 2.1) mukaiseen sovelluskerrokseen, kun taas ei-funktionaaliset vaatimusmäärittelyt koskevat enemmän järjestelmäkerrosta. Havainto ei kuitenkaan ole tarkkarajainen ja lähtökohtaisesti arkkitehtuuria suunnitellessa on tärkeää ottaa huomioon kaikki järjestelmän kerrokset. Monissa tilanteissa sovellus- ja järjestelmäkerroksella on ollut myös erilliset arkkitehtuurit, vaikka näitä molempia arkkitehtuureja ei oltaisi mainittu käsitellyissä aineistoissa. Käytännössä järjestelmäkerroksen ja sovelluskerroksen arkkitehtuurien erot näkyvät esimerkiksi siten, että ajurit tai laitteisto-ohjaimet sijaitsevat lähtökohtaisesti aina järjestelmäkerroksessa, kun taas sovelluskerroksessa voidaan hyödyntää esimerkiksi MVC arkkitehtuuria [27]. Sovelluskerros siis hyödyntää järjestelmäkerroksen abstraktiota ja sen avulla toteuttaa funktionaalisia vaatimuksiaan. Samalla laitteisto- ja järjestelmäkerros käytännössä asettaa ei-funktionaalisisten vaatimusten kautta rajoitteita sovelluskerrokselle. Laitteistokerroksessa voi puuttua esimerkiksi fyysinen antenni, jota vaaditaan langattomiin yhteyksiin.

On tärkeää huomioida kuitenkin, että rajaus arkkitehtuuristen kerrosten (Kuva 2.1) välillä käytännön toteutuksissa tai lähdekoodissa ei välttämättä ole tarkkaan rajattu. Käytännön toteutuksissa tai lähdekoodissa laitteisto-ohjaimet voivat olla samassa hakemistorakenteessa omina tiedostomoduuleina tai alihakemistoina ilman selkeää erottelua sovellus- tai järjestelmäkerrokseen. Käytännössä se tarkoittaisi sitä, että lähdekoodia tarkastelevan tulee pystyä erottamaan moduulien merkitys järjestelmän arkkitehtuurissa. Järjestelmäkerroksen toteutukset ovat kuitenkin toiminnan kannalta välttämättömiä ja siksi paremman uudelleenkäytettävyyden ja selkeyden saavuttamiseksi niiden erottaminen voi olla suositeltavaa.

Alla (Taulukko 3.4) on yhteenveto arkkitehtuurien ja vaatimusmäärittelyiden välisistä yhteyksistä. Taulukossa on listattu arkkitehtuurit, vaatimusten näkökulma ja arkkitehtuurinen fokus järjestelmä- tai sovelluskerrokseen. Taulukko on koosetettu materiaalitaulukoon (Taulukko 3.1) arkkitehtuureihin liittyvistä materiaaleista.

Taulukko 3.4: Analyysi arkkitehtuurien ja vaatimusmäärittelyiden yhteydestä.

Aineisto	**S/ Y**	**Arkkitehtuuri**	**Vaatimusten näkökulma**	**Fokus**
[11]	S	Mikropalvelu-arkkitehtuuri	Funktionaalinen	Sovelluskerros
[12]	S	Kerrosarkkitehtuuri	Funktionaalinen, Ei-funktionaalinen	Sovellus- ja järjestelmäkerros
[14]	S	Ei selvää mallia	Funktionaalinen	Sovelluskerros
[15][16]	S	Palveluorientoituntut arkkitehuuri (SOA)	Funktionaalinen, Ei-funktionaalinen	Sovelluskerros
[18]	S	SOA, modulaarinen	Funktionaalinen, Ei-funktionaalinen	Sovelluskerros
[26]	S	Kerrosarkkitehtuuri	Ei-funktionaalinen	Sovelluskerros
[27]	S	Kerrosarkkitehtuuri, MVC	Funktionaalinen	Sovellus- ja järjestelmäkerros
[10]	Y	Kerrosarkkitehtuuri, RTOS	Ei-funktionaalinen	Järjestelmäkerros
[13]	Y	RTOS, no-OS, full-OS, app-, server-, & container-OS	Ei-funktionaalinen	Järjestelmäkerros
[20]	Y	Komponenttipohjainen, kerrosarkkitehtuuri, väliohjelmistot	Ei-funktionaalinen	Sovellus- ja järjestelmäkerros
[21]	Y	Arkkitehtuurinen hierarkia, ei yleistä mallia	Funktionaalinen, Ei-funktionaalinen	Sovellus- ja järjestelmäkerros
[23]	Y	SOA	Funktionaalinen, Ei-funktionaalinen	Sovellus- ja järjestelmäkerros
[24]	Y	SOA, MVC, asiakas-palvelin, mikropalveluarkkitehtuuri	Funktionaalinen	Sovelluskerros
[28]	Y	Asiakas-palvelin, MVC	Ei-funktionaalinen	Sovelluskerros

Voidaan todeta, että funktionaaliset vaatimusmäärittelyt koskivat useammin soveltavia aineistoja sekä sovelluskerroksen arkkitehtuureja. Ei-funktionaalisia vaatimukset näkyivät useammin yleisiksi luokitelluissa materiaaleissa ja järjestelmäkerrosta tarkastellessa. Muutamia poikkeuksia kuitenkin havaittiin. Funktionaalisiin määrityksiin liittyi useammin sovelluskerroksen ratkaisuja, joita voidaan käyttää myös muissa, kuin sulautetuissa järjestelmissä. Järjestelmäkerroksen arkkitehtuureista yksi keskeisimmistä voidaan todeta olevan RTOS, joka on sulautetuille järjestelmille erityinen ja harvoin sovellettu muihin kuin sulautettuihin järjestelmiin.

Voidaan todeta myös, että sovelluskerrokseen voi soveltaa lähes mitä tahansa yleistä arkkitehtuurista mallia, joista varsin yleisinä havaittiin SOA, MVC ja mikropalvelut. Tässä tutkielmassa käytetty kerrosarkkitehtuurinen näkökulma on myös hyvin yleinen käsitellyissä aineistoissa. Hyvin monissa näkökulmissa yhdisteltiin kuitenkin useampia arkkitehtuurisia malleja, jotka helpottavat merkittävästi abstraktion hahmottamista. Laajemmissa ja monimutkaisemmissa järjestelmissä hierarkkinen arkkitehtuurin suunnittelutapa voi olla myös hyödyllinen [21].

Alla (Taulukko 3.5) ehdotetaan näkökulmaa, joka perustuu kirjallisuuskatsauksessa käsiteltyihin aineistoihin ja toimii tiivistelmänä tutkielman tuloksista. Näkökulma listaa yleisimmät tunnistetut arkkitehtuurit eri järjestelmän kerroksille sekä painotuksen vaatimusmäärittelyiden näkökulmasta.

Taulukko 3.5: Analyysin yhteenveto ja ehdotettu näkökulma.

	Arkkitehtuurit	**Vaatimukset**
Sovelluskerros	SOA, MVC, Mikropalveluarkkitehtuuri, muut yleiset arkkitehtuurit	Funktionaalisia vaatimuksia painottava
Järjestelmäkerros	RTOS, no-OS, full-OS, app-, server-, container-OS, väliohjelmistot, ajurit	Ei-funktionaalisia vaatimuksia painottava

4 Yhteenveto

Tämän kirjallisuuskatsauksen tarkoituksena on ollut tarjota korkean tason yleiskuva sulautettujen järjestelmien vaatimusmäärittelyistä, ohjelmistoarkkitehtuureista ja näiden välisistä yhteyksistä. Ensin on tarkasteltu tutkielmaan liittyviä taustatietoja, joita ovat funktionaaliset ja ei-funktionaaliset vaatimusmäärittelyt sekä kerrosarkkitehtuurinen sulautettujen järjestelmien malli. Tämän jälkeen on käsitelty kirjallisuuskatsauksen tulokset aloittaen materiaaliluokituksista ja sitten vastattu tutkimuskysymyksiin käsiteltyjen 24 aineiston pohjalta.

Ensimmäinen tutkimuskysymys (TK1) selvittää sulautetuissa järjestelmissä yleisimpiä vaatimusmäärittelyitä. Käsitellyissä aineistoissa korostuivat ei-funktionaaliset vaatimukset. Tunnistettiin viisi yleisintä vaatimusmäärittelyä, joita esiintyi suurimmassa määrässä käsiteltyjä aineistoja. Näitä vaatimuksia olivat suorituskyky, luotettavuus, päivitettävyys, tietoturva ja modulaarisuus.

Toinen tutkimuskysymys (TK2) selvittää sulautetuille järjestelmille yleisimpiä arkkitehtuureja. Yleisimpiä tunnistettuja arkkitehtuureja olivat SOA, MVC, mikropalveluarkkitehtuuri ja RTOS. Arkkitehtuurien tarkastelussa käytettiin usein kerrosarkkitehtuurista näkökulmaa. Lisäksi olennaisina arkkitehtuureina mainittiin myös komponenttipohjainen arkkitehtuuri, asiakas-palvelin-arkkitehtuuri ja väliohjelmistot.

Kolmannessa tutkimuskysymyksessä (TK3) selvitettiin tunnistettujen vaatimusmäärittelyiden ja arkkitehtuurien välistä yhteyttä käsiteltyjen aineistojen perus-

teella. Tarkastelussa tunnistettiin funktionaalisten vaatimusmäärittelyiden koskevan enemmän järjestelmän sovelluskerroksen arkkitehtuureja. Kun taas ei-funktionaaliset vaatimusmäärittelyt todettiin koskevan useammin järjestelmäkerroksen arkkitehtuureja. Sovelluskerroksessa tunnistetut arkkitehtuurit havaittiin olevan myös muissa kuin sulautetuissa järjestelmissä yleisiä ja näitä olivat esimerkiksi SOA, MVC ja mikropalveluarkkitehtuuri. Järjestelmäkerroksen arkkitehtuurit ovat taas useammin vain sulautetuille järjestelmille alakohtaisia, joista tärkeimpänä tunnistettiin RTOS sekä olennaisina komponentteina väliohjelmistot ja ajurit. Havaintojen yhteenvetona koostettiin taulukot, jotka toimivat havaintojen tiivistelmänä ja ehdotettuna näkökulmana.

Tutkielmassa on päästy tavoitteisin ja tutkimuskysymyksiin on pystytty vastaamaan. On tärkeää kuitenkin huomioida, että käsitellyt aineistot perustuvat pääasiassa akateemisiin tietokantoihin ja rajattuun aineiston kokoon, joka voi rajoitta tuloksia ja näkökulmia. Tutkielmassa ei olla tarkasteltu käytännön projekteissa olevia dokumentaatioita tai lähdekoodeja, joka on selkeä puute ja mahdollinen jatkotutkimusaihe. Lisäksi kirjallisuuskatsaus ei ota kantaa moniin muihin kriittisiin päätöksiin, jotka liittyvät keskeisesti sulautettujen järjestelmien ohjelmistosuunnitteluun. Näitä päätöksiä ovat ohjelmointikielien ja -kehyksien sekä laitteiston valinta. Tuloksia voidaan hyödyntää sulautettujen järjestelmien alkuvaiheen suunnittelussa sekä yleisten arkkitehtuurivaihtoehtojen ja vaatimusmäärittelyiden selvittämisessä. Tutkielma sopii myös uusille sulautettuihin järjestelmiin perehtyville lukijoille ja tarjoaa heille kattavan yleiskuvan aiheesta.

Lähdeluettelo

[1] D. Lacamera, *Embedded systems architecture: discover software programming on embedded devices to build safe and secure connected systems*, eng, Second edition. Birmingham: Packt Publishing, 2023, ISBN: 978-1-80323-954-5.

[2] T. Noergaard, *Embedded systems architecture: a comprehensive guide for engineers and programmers*, eng, 2nd ed. Oxford: Newnes, 2013, ISBN: 978-0-12-382197-3.

[3] A. J. Lattanze, "Infusing Architectural Thinking into Organizations", *IEEE Software*, vol. 29, nro 1, s. 19–22, tammikuu 2012, ISSN: 1937-4194. DOI: 10.1109/MS.2012.12.

[4] GeeksforGeeks, *Functional and Non Functional Requirements*, Section: Software Engineering, 2025. viitattu 23. lokakuuta 2025. url: https://www.geeksforgeeks.org/software-engineering/functional-vs-non-functional-requirements/.

[5] C. C. Insaurralde ja A. Zoitl, "System requirements in industrial automation", teoksessa *2013 11th IEEE International Conference on Industrial Informatics (INDIN)*, ISSN: 2378-363X, heinäkuu 2013, s. 572–577. DOI: 10.1109/INDIN.2013.6622947.

[6] C. Ponsard ja M. Delehaye, "Towards a Model-Driven Approach for Mapping Requirements on AADL Architectures", teoksessa *2009 14th IEEE Internatio-*

nal Conference on Engineering of Complex Computer Systems, kesäkuu 2009, s. 353–358. DOI: 10.1109/ICECCS.2009.33.

[7] K. Yimei, "A Graduate Program on Embedded Software Engineering in China", teoksessa *20th Conference on Software Engineering Education & Training (CSEET'07)*, ISSN: 2377-570X, heinäkuu 2007, s. 3–10. DOI: 10.1109/CSEET.2007.7.

[8] A. El Jalaoui, D. Andreu ja B. Jouvencel, "A control architecture for contextual tasks management: application to the AUV Taipan", teoksessa *Europe Oceans 2005*, vol. 2, kesäkuu 2005, 752–757 Vol. 2. DOI: 10.1109/OCEANSE.2005.1513150.

[9] B. Turban, C. Wolff, A. Tsakpinis ja M. Kucera, "A decision model for managing and communicating resource restrictions in embedded systems design", teoksessa *2008 International Workshop on Intelligent Solutions in Embedded Systems*, heinäkuu 2008, s. 1–12. DOI: 10.1109/WISES.2008.4623304.

[10] B. Kang, Y.-J. Kwon ja R. Lee, "A design and test technique for embedded software", teoksessa *Third ACIS Int'l Conference on Software Engineering Research, Management and Applications (SERA'05)*, elokuu 2005, s. 160–165. DOI: 10.1109/SERA.2005.6.

[11] R. Seiger, M. Franceschetti ja A. Abbad-Andaloussi, "A Process to Non-invasively Augment Legacy IoT Systems Using Business Processes and Microservices", teoksessa *Proceedings of the 14th International Conference on the Internet of Things*, sarja IoT '24, New York, NY, USA: Association for Computing Machinery, 2025, s. 1–9, ISBN: 979-8-4007-1285-2. DOI: 10.1145/3703790.3703791. url: https://doi.org/10.1145/3703790.3703791.

[12] H. Turkmanović ja I. Popović, "A systematic approach for designing battery management system for embedded applications", teoksessa *2021 Zooming In-*

novation in Consumer Technologies Conference (ZINC), toukokuu 2021, s. 85–90. DOI: 10.1109/ZINC52049.2021.9499253.

[13] A. Taivalsaari ja T. Mikkonen, "A Taxonomy of IoT Client Architectures", *IEEE Software*, vol. 35, nro 3, s. 83–88, toukokuu 2018, ISSN: 1937-4194. DOI: 10.1109/MS.2018.2141019.

[14] F. G. C. Ribeiro, A. Reuberg, C. E. Pereira ja M. S. Soares, "An Approach for Architectural Design of Automotive Systems using MARTE and SysML", teoksessa *2018 IEEE 14th International Conference on Automation Science and Engineering (CASE)*, ISSN: 2161-8089, elokuu 2018, s. 1574–1580. DOI: 10.1109/COASE.2018.8560415.

[15] E. Pastor, J. Lopez ja P. Royo, "An Embedded Architecture for Mission Control of Unmanned Aerial Vehicles", teoksessa *9th EUROMICRO Conference on Digital System Design (DSD'06)*, elokuu 2006, s. 554–560. DOI: 10.1109/DSD.2006.24.

[16] L. Chamari, E. Petrova ja P. Pauwels, "An End-to-End Implementation of a Service-Oriented Architecture for Data-Driven Smart Buildings", *IEEE Access*, vol. 11, s. 117 261–117 281, 2023, ISSN: 2169-3536. DOI: 10.1109/ACCESS.2023.3325767.

[17] J. S. Fant, H. Gomaa ja R. G. Pettit, "Architectural Design Patterns for Flight Software", teoksessa *2011 14th IEEE International Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing Workshops*, maaliskuu 2011, s. 97–101. DOI: 10.1109/ISORCW.2011.39.

[18] S. Han, R. Sinha ja A. Lowe, "Assessing Support for Industry Standards in Reference Medical Software Architectures", teoksessa *IECON 2020 The 46th Annual Conference of the IEEE Industrial Electronics Society*, ISSN: 2577-1647, lokakuu 2020, s. 3403–3407. DOI: 10.1109/IECON43393.2020.9255309.

[19] L. F. Rahman, T. Ozcelebi ja J. J. Lukkien, "Choosing Your IoT Programming Framework: Architectural Aspects", teoksessa *2016 IEEE 4th International Conference on Future Internet of Things and Cloud (FiCloud)*, elokuu 2016, s. 293–300. DOI: 10.1109/FiCloud.2016.49.

[20] L. Wang, "Component-based performance-sensitive real-time embedded software", teoksessa *24th Digital Avionics Systems Conference*, ISSN: 2155-7209, vol. 2, lokakuu 2005, 10 pp. Vol. 2–. DOI: 10.1109/DASC.2005.1563399.

[21] B. Graaf, M. Lormans ja H. Toetenel, "Embedded software engineering: the state of the practice", *IEEE Software*, vol. 20, nro 6, s. 61–69, marraskuu 2003, ISSN: 1937-4194. DOI: 10.1109/MS.2003.1241368.

[22] A. Ibrahim, L. Zhao ja J. Kinghorn, "Embedded systems development: quest for productivity and reliability", teoksessa *Fifth International Conference on Commercial-off-the-Shelf (COTS)-Based Software Systems (ICCBSS'05)*, helmikuu 2006, 10 pp.–. DOI: 10.1109/ICCBSS.2006.12.

[23] M. Fahmideh ja D. Zowghi, "IoT Smart City Architectures: An Analytical Evaluation", teoksessa *2018 IEEE 9th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, marraskuu 2018, s. 709–715. DOI: 10.1109/IEMCON.2018.8614824.

[24] H. Washizaki, S. Ogata, A. Hazeyama, T. Okubo, E. B. Fernandez ja N. Yoshioka, "Landscape of Architecture and Design Patterns for IoT Systems", *IEEE Internet of Things Journal*, vol. 7, nro 10, s. 10 091–10 101, lokakuu 2020, ISSN: 2327-4662. DOI: 10.1109/JIOT.2020.3003528.

[25] F. Alkhabbas, R. Spalazzese, M. Cerioli, M. Leotta ja G. Reggio, "On the Deployment of IoT Systems: An Industrial Survey", teoksessa *2020 IEEE International Conference on Software Architecture Companion (ICSA-C)*, maaliskuu 2020, s. 17–24. DOI: 10.1109/ICSA-C50368.2020.00012.

[26] R. Raja, ”Software Architecture for Agricultural Robots: Systems, Requirements, Challenges, Case Studies, and Future Perspectives”, *IEEE Transactions on AgriFood Electronics*, vol. 2, nro 1, s. 125–137, maaliskuu 2024, ISSN: 2771-9529. DOI: 10.1109/TAFE.2024.3366335.

[27] J. F. Mendoza, H. Ordonez, C. Figueroa, R. Segovia, D. Munoz ja A. Ordonez, ”Software Architecture for Sensor Nodes in the Internet of Things: A Case of Study in Agriculture”, teoksessa *2018 Fifth International Conference on Internet of Things: Systems, Management and Security*, lokakuu 2018, s. 3–8. DOI: 10.1109/IoTSMS.2018.8554882.

[28] R. Raymond ja S. M. A. Savarimuthu, ”Software Design Patterns and Architecture Patterns –A Study Explored”, teoksessa *2022 5th International Conference on Contemporary Computing and Informatics (IC3I)*, joulukuu 2022, s. 1998–2006. DOI: 10.1109/IC3I56241.2022.10073279.

[29] D. Mijić ja E. Varga, ”Unified IoT Platform Architecture Platforms as Major IoT Building Blocks”, teoksessa *2018 International Conference on Computing and Network Communications (CoCoNet)*, elokuu 2018, s. 6–13. DOI: 10.1109/CoCoNet.2018.8476881.