SAVONIA

OPINNÄYTETYÖ - **AMMATTIKORKEAKOULUTUTKINTO**
TEKNIIKAN JA LIIKENTEEN ALA

MOBIILIROBOTIN LIITTÄMINEN TUOTANNONOHJAUKSEEN

TEKIJÄ: Hannele Ranta

SAVONIA-AMMATTIKORKEAKOULU

OPINNÄYTETYÖ
Tiivistelmä

Koulutusala Tekniikan ja liikenteen ala	
Koulutusohjelma/Tutkinto-ohjelma Sähkötekniikan tutkinto-ohjelma	
Työn tekijä(t) Hannele T Ranta	
Työn nimi Mobiilirobotin liittäminen tuotannonohjaukseen	
Päiväys 21.5.2020	Sivumäärä/Liitteet 55
Ohjaaja(t) Lehtori Pasi Lepistö ja Lehtori Jari Ijäs	
Toimeksiantaja/Yhteistyökumppani(t) Manu Pernu	

Tiivistelmä

Työn alkuperäinen toimeksiantaja oli Servicepoint Kuopio Oy, joka tuotti erilaisia teollisuuden koneautomaatioratkaisuja. Yhtiössä oli vahvasti panostettu uudenlaisten ratkaisujen toteutukseen ja löytämiseen, mikä oli yksi kimmoke opinnäytetyölle. Toimeksianatajan konkurssi kesken opinnäytetyöprosessin purki toimeksiantosopimuksen ja muutti suorituksen oppilaitospohjaiseksi.

Työn taustalla oli sisälogistiikkaan keskittyneen Mobile Industry Robotsin mobiilirobotit ja niiden keskitetty ohjausjärjestelmä MiR Fleet. Sekä robotit että Fleet on toteutettu perinteisistä automaattitrukeista poiketen REST-protokollaa hyödyntäen. Protokollaa oli hyödynnetty myös käyttöliittymien implementoinnissa mobiililaitteisiin. Ratkaisuille on lisäksi rakennettu REST API -kirjastot, joiden avulla laitteisto voidaan liittää osaksi muita järjestelmiä. Työssä taustalla haluttiin myös tunnistaa automaatiotason tuotannonohjauksen elementtejä, joita voisi ohjata lean -menetelmiä hyödyntäen.

Tavoitteena työssä oli löytää käyttökelpoisia ratkaisumalleja liittää MiR-moobiilirobotit ja järjestelmät osaksi perinteisiä automaatioratkaisuja sekä tuotannon- ja toiminnanohjausjärjestelmiä. Tavoitteena oli myös tutustua erilaisissa ympäristöissä käytettyihin tiedonsiirron rajapintoihin ja kuvata niiden toimintaa sekä käyttömahdollisuuksia teollisessa ympäristössä, jonka tiedonsiirto ja käytetyt protokollat vaihtelevat. Tuotannonohjauksen lean-näkökulma haluttiin sisällyttää työhön, jotta ohjauksen mallit tukisivat myös tuotannon virtausta.

Työn tuloksena saatiin koostettua paljon teoreettista taustatietoa tiedonsiirrosta, ohjelmointialustoista, arkkitehtuurimalleista sekä lean -filosofiasta. Tärkeimpänä tuloksena kuitenkin topologiatason mallit tiedonsiirron rakentamisesta REST API -rajapintaa hyödyntäen. Työ sisältää myös pohdintaa mallien käyttöönottamisen hyödyistä yritystoiminnan vahvistamisessa ja tukemisessa, kun malleja arvioidaan lean-filosofian näkökulmasta.

Avainsanat
Tuotannonohjaus, Mobiilirobotti, REST API, Lean

Julkinen

SAVONIA UNIVERSITY OF APPLIED SCIENCES

THESIS
Abstract

Field of Study Technology, Communication and Transport			
Degree Programme Degree Programme in Electrical Engineering			
Author(s) Hannele T Ranta			
Title of Thesis Mobile Robot Integration to the Production Control			
Date	21 May 2020	Pages/Appendices	55
Supervisor(s) Mr. Pasi Lepistö, Senior Lecturer and Mr. Jari Ijäs, Senior Lecturer			
Client Organisation /Partners Servicepoint Kuopio Oy / Manu Pernu			

Abstract

This thesis was commissioned by Servicepoint Kuopio Oy. At the point of starting this thesis, Servicepoint Kuopio Oy was producing automatic machinery equipment widely to different industry sectors. Servicepoint was investing in new technologies which was one kick to start investigating this subject. Unfortunately for this thesis, Servicepoint Kuopio Oy went bankrupt in the middle of the process. After that the project changed to an independent student work.

The project was launched with a development and enquiry aspect to find new possibilities to use MiR (Mobile Industry Robots) mobile robots and their control system in industry. MiR robots and MiR Fleet control system utilize REST protocols between a single robot and the control system, which differs from more traditional AGV truck solutions. With REST, the user interface is available for all mobile devices and personal computers through standard web browsers. The REST API libraries have been opened for implementing MiR systems as a third-party subject to all industry systems.

The scope of the project was to model the topology of the modern mobile robot interface connecting the traditional control systems used in industry. It was also interesting to look widely different environmental solutions to build-up interfaces between different protocols. The lean based production control is widely recognized, and in this thesis can be recognized production automation control elements that could be controlled with lean philosophy.

As a result, a lot of theoretical knowledge of interfaces, protocols, different architectural models and lean philosophy was collected. The most important outcome of this thesis project are the models, a useful sample of REST usage and review from the lean and business point of view.

Keywords
Production control, Mobile robot, REST API, Lean

Public

ESIPUHE

Tämän työn toimeksiantaja oli Servicepoint Kuopio Oy, jonka yrittäjille haluan osoittaa kiitokseni luottamuksesta ja mahdollisuuksista niin tämän työn kuin ammatillisen kehittymisen osalta. Vuodet ovat kasvattaneet minua valtavasti. Isot kiitokset haluan osoittaa myös työtä ohjanneelle Manu Pernulle: kiitos, että jaksoit selittää ja ohjata kiireenkin keskellä. Kiitokset kuuluvat myös Servicepointilla työskennelleille kollegoille, joiden apuun on aina voinut luotta ja joiden apu on ollut korvaamatonta tämän työn tekemisessä ja viimeistelyssä. Matka on ollut pitkä ja määränpäätä arvokkaampi.

Haluan esittää kiitokseni myös äidilleni, joka on mahdollistanut varsinkin opiskelun alkutaipaleen työssäkäynnin ohella. Kiitokset Sami, että olet jaksanut huolehtia kodista ja minusta opiskelun keskellä. Veljelleni Anssille kiitokset teknisestä rautalanka väännöstä, jota olen välillä niin kovasti kaivannut ymmärtääkseni. Lopuksi mutta ei vähäisimmäksi haluan kiittää poikaani Väinöä loputtomasta ideoinnista ja asioiden näkemisestä oikeassa perspektiivissä. Päätän esipuheeni lainaten Väinön ehdotusta tämän esipuheen alkusanoiksi: ”Hyvät insinöörit.”

Kuopiossa 21.5.2020
Hannele Ranta

SISÄLTÖ

LYHENTEET JA MÄÄRITELMÄT

API	Application Programming Interface. Eng. Ohjelmointirajapinta.
BI	Business Intelligence. Eng. Liiketoimintatiedon hallinta.
CRM	Customer Relationship Management. Eng. Asiakastietojärjestelmä.
DB	Database. Eng. tietokanta.
DDE	Dynamic Data Exchange. Eng. Dynaaminen tiedonvaihto, sallii objektien manipuloinnin eri ohjelmista käsin.
ERP	Enterprise Resource Planning. Eng. Toiminnanohjausjärjestelmä.
HMI	Human Machine Interface. Eng. Ihmiselle suunnattu koneen käyttöliittymä.
IoT	Internet Of Things. Eng. Esineiden internet.
IP	Internet Protocol address. Eng. Internetin protokollaosoite tai - yhteiskäyttöosoite.
ISO	The International Organization for Standardization. Kansainvälinenstandardointi organisaatio.
JSON	JavaScript Object Notation. Eng. JavaScript pohjainen objekti esitys tiedon välitykseen.
LEAN	Eng. laiha, heikko. Lean on prosessijohtamisen filosofia. Lean on käsite, joka on kehittynyt vuosien saatossa Toyotan ympärillä.
LLC	Logical Link Control Eng. Siirtoyhteyskerros.
MAC	Media Access Control (MAC-address) Eng. Verkkosovittimen yksilöivä osoite.
MES	Manufacturing Execution Systems. Eng. Tuotannonohjausjärjestelmä.
MQTT	Message Queuing Telemetry Transport. Eng. Julkaisu ja viestinvälitys protokolla.

ODBC	Open Database Connectivity. Eng. Avoin ohjelmointirajapinta tietokantajärjestelmille.
OSI	Open Systems Interconnection model. Eng. OSI-malli tiedonsiirtoprotokollien kuvaamiseen.
PLC	Programmable Logic Controller. Eng. Ohjelmoitava logiikka.
PLM	Product Lifecycle Management. Eng. Tuotteen elinkaaren hallinta.
PP&C	Production Planning & Control Eng. Tuotannonsuunnittelu ja -ohjaus. (Kinra 2019. s.1)
REST	Representational State Transfer. Eng. Yleinen arkkitehtuurimalli rajapintojen toteuttamiseen.
SCM	Supply Chain Management. Eng. Toimitusketjun hallinta.
SRM	Supplier Relationship Management. Eng. Toimittajatietojärjestelmä.
SCADA	Supervisory Control and Data Acquisition Eng. Valvomo-ohjelmisto.
TPS	Toyota Production System. Toyotan keittämä ja käyttämä toiminnanohjauksen malli.
TPC	Transmission Control Protocol. Eng. Tietoliikenneprotokolla.
UAV / UAS	Unmanned Aerial Vehicles / Uncrewed Aircraft Systems Eng. Miehittämätön ilma-ajoneuvo / miehittämätön ilma-alus.
UGV	Unmanned Ground Vehicles Eng. Miehittämätön maassa kulkeva ajoneuvo.
WMS	Warehouse Management System. Eng. Varastonhallintajärjestelmä.

1 JOHDANTO

Tässä opinnäytetyössä käsiteltävät aiheet linkittyvät vahvasti sekä teollisuusautomaation, ohjelmoinnin että tuotannollisten näkökulmien yhteensovittamiseen. Käsiteltävinä aiheina ovat automaatiojärjestelmien tiedonsiirtorajapinnat, niiden vaatimukset sekä tuotannonohjauksen keskittäminen. Aiheet liikkuvat vahvasti tuotantotekniikan sekä tietotekniikan opinto-ohjelmien sisällöissä, mutta ovat tärkeitä ja läsnä myös sähkö- ja automaatiotekniikan osalta.

Tuotannonohjauksen keskittäminen, sekä järjestelmien liittäminen yhdeksi ehyeksi kokonaisuudeksi, lisää tuottavuutta. Tässä opinnäytetyössä pyritään luovasti löytämään uusia mahdollisuuksia tuotannollisen työn kannattavuuden parantamiseen. Mahdollisia lopputuloksia ovat niin mekaanisesti rakennetut järjestelmät, ohjelmistopohjaiset ratkaisut sekä niiden yhdistelmät.

Heti alkuun tulee kuitenkin tehdä selväksi ero toiminnan- ja tuotannonohjausjärjestelmien välillä, vaikkakin ne saattavat olla myös samassa järjestelmäkokonaisuudessa. Toiminnanohjausjärjestelmä eli ERP on yritystasoinen ratkaisu, jonka painotusalueet ovat taloudessa ja luvuissa, kun taas tuotannonohjausjärjestelmä eli MES-ympäristö on tuotannonsuunnittelun ja kuormituksen ylläpitoon tarkoitettu järjestelmä. Järjestelminä nämä kokonaisuudet voivat olla, joko erilliset moduulit tai MES voi olla osa ERP-järjestelmää. Tässä opinnäytetyössä käsitellään tuotannonohjausta siten kuin ERP ja MES käsitteinä olisivat sama järjestelmä, koska tiedonsiirtotekniikat toimivat molempien osalta samoin.

Mobiilisovellukset ovat yleisesti rakennettu toimimaan verkkoympäristössä ja niiden liityntäpinnat on avattu vapaasti käytettäviksi, tämän tullessa vasta viime vuosina teollisiin automaatiojärjestelmiin. Uudet valmistajat ovat olleet vauhdittamassa perinteisten valmistajien kehittymistä, tuomalla markkinoille ihmisille helposti lähestyttäviä mobiiliratkaisuja. Vanhemmat toimijat pääsevät tai joutuvat tämän myötä opettelemaan ja kehittämään itselleen vapaita tai geneerisiä lähdekoodeja, joilla on vastaavia mobiilinen yhteensopivuus. Ohjelmistoautomaation kehittymistä tuetaan myös ohjelmistorobotiikan avulla, kun vanha ja uusi maailma yhdistetään älykkäästi ennen järjestelmäkehityksen seuraavaa vaihetta.

1.1 Yhteistyökumppanit ja tekijänoikeuksien haltijat tai muut tahot

Pernu, Manu. Työnohjaaja.

2 TEORIATAUSTAA

Seuraavissa kappaleissa esitetään tuotannonohjausjärjestelmän periaatteita, mobiilirobotin ideologiaa ja API-rajapinnan toiminnallista taustaa. Koska tuotannonohjaus on vahvasti sidoksissa myös yrityksen prosesseihin, tulee myös tuon prosessin periaatteita avata.

Teknistä aspektia ja kehityskulkua selventämään on syytä ymmärtää, että teollisenvalankumouksen aikakausia kuvataan kielikuvalla *Industry n.0*, jossa n korvataan numerolla. Tällä hetkellä menossa on neljäs aikakausi eli *Indusrty 4.0.* Jokainen aikakausi on tuonut jotain uutta teolliseen tuotantoon. (Kinra, 2019. s.481-484)

Tässä opinnäytetyössä ollaan neljännen ja viidennen vallankumouksen kynnyksellä eräänlaisessa siirtymä. Michael Rada kuvaa Industry 5.0 tulevaisuudeksi ja läpitunkevaksi trendiksi, jossa ihminen ja kone tekevät lähempää yhteistyötä. Myös kaikenlaista hukkaa pyritään välttämään. (Rada, 2018)

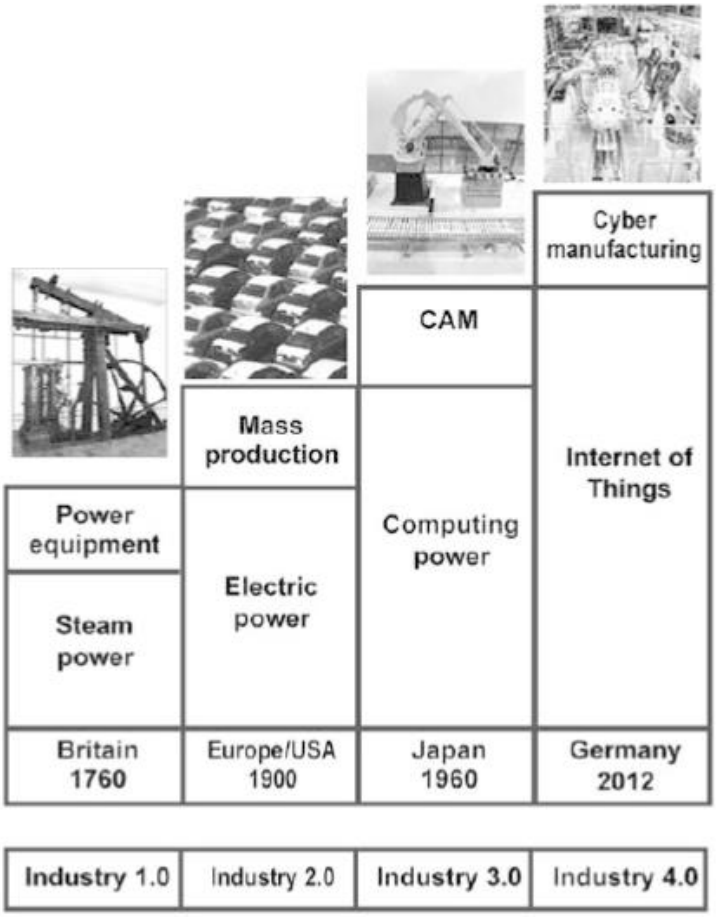


FIGURE 34.2 The dynamic face of industrial manufacture.

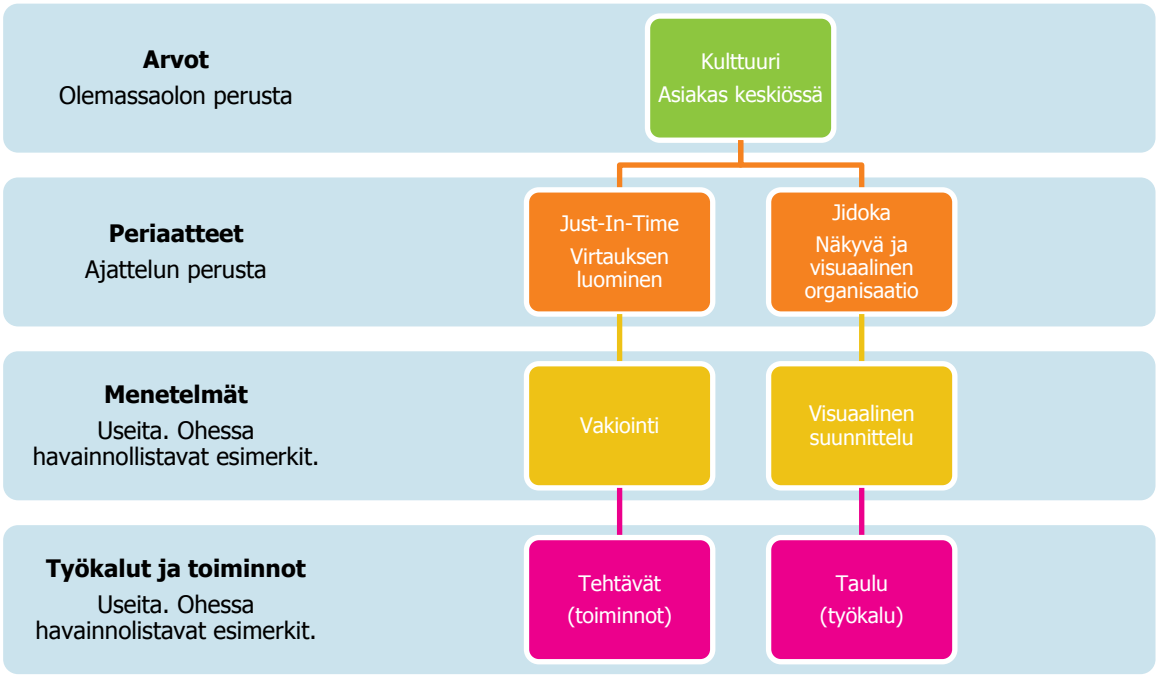
Kuva 1. Teollisentuotannon aikakaudet. (Kinra 2019. s.484 Figure 34.2)

2.1 Yritysprosessit ja Lean

Jyväskylän Yliopiston Laatujärjestelmän kuvauksessa määritellään prosessi seuraavasti: *Toiminta-tapa, jonka avulla panoksista syntyy tuotoksia, lisäarvoa tuottavia tuloksia ja vaikutuksia.* Prosessille luonteenomaista on, että sille erottuu selkeästi alku ja loppu. (Jyväskylän Yliopisto, 2020)

Six Sigma sivustolla tiivistetään Leanin olevan *asiakaslähtöinen prosessijohtamisen malli. Se perustuu virtauksen (exit rate) maksimointiin ja hukan (menetetty aika) poistamiseen."* (Quality Knowhow Karjalainen Oy, 2020)

Modig ja Åhlström määrittävät leanin omassa teoksessaan toimintastrategiaksi. Toyotan tuotantojärjestelmä (TPS) on filosofia, jonka perimmäinen tarkoitus on toteuttaa organisaation arvoja. (Modig, 2018. s.117, 77 ja 127-130) Prosessi siis kuvaa tehtäviä toimintoja ja lean on filosofia prosessin toteuttamiseen.



Kuva 2. Lean -filosofian abstraktitasojen kuvaus (mukaillen tekstistä Modig 2018 s.127-142)

Lean -filosofian abstraktitason kuvauksen mukaisesti prosessiin pyritään luomaa virtausta (Just-In-Time) ja se pyritään tekemään näkyväksi kaikille (Jidoka), jotta hukka eli poikkeama saadaan näkyväksi. (Modig, 2018. s. 127-142 ja 75)

Hukalle on määritetty kahdeksan perusmuotoa: ylituotanto, odottelu, kuljettaminen, tarpeeton käsittely, varastointi, tarpeeton liikkuminen, viat ja viimeisimpänä työntekijöiden ideoiden ja luovuuden hyödyntämättä jättäminen. (CERIFFI, 2020)

2.2 Toiminnan- ja tuotannonohjausjärjestelmät

Professori D. R. Kinran *Production Planning and Control* - kirjassa tuotannonsuunnittelua ja tuotannonohjausta (PP&C) kuvaillaan käsitteenä tuotannon aivoiksi ja hermojärjestelmäksi. Tuotannonsuunnittelu on dynaamista ja olosuhteiden muutokset vaikuttavat suunniteltuun kuitenkin huomioiden, että niillä pyritään saavuttamaan *materiaalien saatavuus, kokoonpano oikeaan aikaan ja oikeassa paikassa sekä oikeat määrät huomioiden, jotta toiminnot edistyvät ennakoidussa ajassa ja mahdollisimman pienin kustannuksin.* (Kinra, 2019. s.1)

Toiminnanohjausjärjestelmällä (ERP) pyritään käytännössä tehostamaan ja parantamaan tiedonkulkua. Toiminnanohjausjärjestelmät kattavat useita yritystoiminnan osa-alueita kuten talous, tiliöinti, resurssit, toimitusketju ja asiakastiedot. ERP:n on yleensä liitetty erilaisia moduuleita, joilla on käytössään yksi tietokanta (Database). (Samara, 2015. s.13)

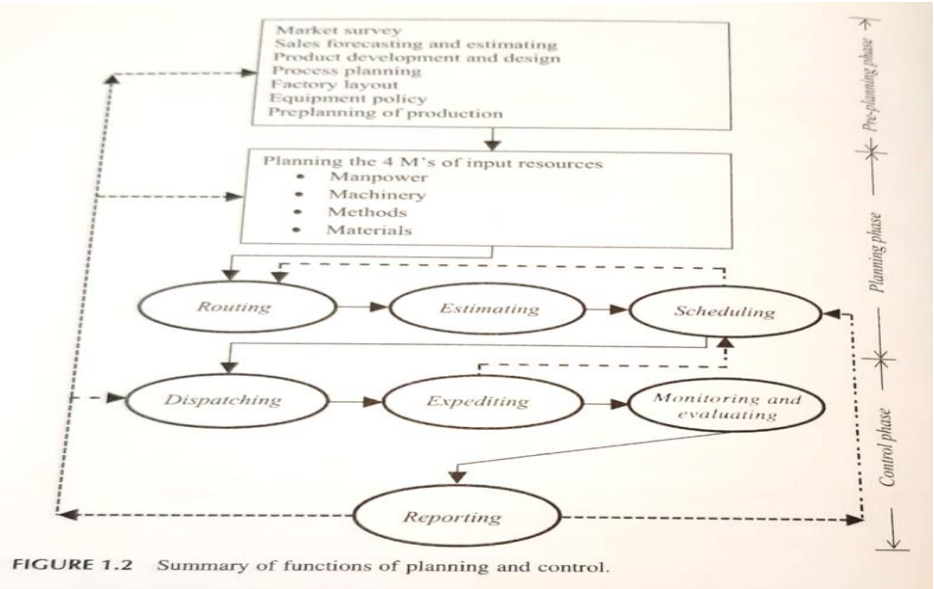
"The ERP system integrates various operational data into one database. The best known and most frequently used system is SAP." (PLC DATA TOOLS, 2020)

ERP järjestelmät luokitellaan kahteen sukupolveen ensimmäiseen (1st generation) ja toiseen (2nd generation). Näiden erona on järjestelmien kattavuus eriosa-alueilla. Ensimmäinen sukupolvi pitää sisällään *talouden, tiliöinnin, kontrolloinnin, omaisuuden, ihmisresurssit, tuotannon, materiaalinhallinnan, myynnin ja toimituksen, tehdaskunnossapidon, projektinhallinnan ja laadunhallinnan.* Toinen sukupolvi pitää sisällään sekä ensimmäisen sukupolven toiminnot että uudet moduulit *CRM, SRM, SCM, PLM, BI, sähköinen liiketoiminta yms.* (Samara, 2015. s.13)

Jotta ymmärtää miksi tuotannonohjauksen järjestelmät yleisesti ovat niin monimuotoisia, täytyy ymmärtää tuotannon toimintoja, joita järjestelmällä pyritään hallitsemaan. Professori Kinra kuvaa kirjassaan hyvin eri toimintoja, joita tuotannonsuunnittelussa ja -ohjauksessa suoritetaan. Kirjan ensimmäisessä kappaleessa käsitellään niin erilaisia rooleja, tarkoitusta sekä tavoitteita, mitkä kaikki vaikuttavat tuotannonohjaukseen. (Kinra 2019. s.2-5)

Perinteinen tuotannonohjausjärjestelmä ja koneohjaus ovat erilliset integroimattomat järjestelmät. Edellä mainittu rakenne on kolmatta teollisen vallankumouksen kautta, modernisti ilmaistuna Industry 3.0. Nykyinen teollisenvallankumouksen aika pitää sisällään ensi askeleen koneoppimisen hyödyntämiseen, mikä tuo järjestelmät integraatioon reaalimaailman ilmiöiden kanssa, niiden liittyessä mobiilirajapintoihin ja sitä kautta sensoreihin. (Kinra, 2019. s.482-483)

Kinra esittää kirjassaan tuotannon suunnittelun jakamista kolmeen selkeään vaiheeseen, jotka on esitetty kuvassa 3. Jaottelu ylhäältä alas on esisuunnittelu-, suunnittelu- ja ohjausvaihe. Suunnittelua voidaan taas jakaa pitkän -, keskipitkän - ja lyhyen tähtäimen suunnitelmiin. Tuotannonsuunnittelussa, joka koskee koneita ja lataamisen aikataulutusta, ollaan vahvasti lyhyentähtäimen suunnittelun tasolla. (Kinra, 2019. s.1-9)



Kuva 3. Tuotannonsuunnittelu jaettuna kolmeen selkeään vaiheeseen. (Kinra, 2019. s.4 Figure 1.2)

2.2.1 Asiakastarpeen määrittäminen

Asiakastarve uudelle sovellukselle perustuu nykyisten järjestelmien jäykkyyteen. Perinteisten ohjelmistojen liitettävyys perustuu rajoitettuihin rajapintoihin. Asiakastarpeen osalta määriteltiin suunnittelu vaiheessa tutkimuskysymyksiksi: *Millainen tuotannonohjaus? Millaisia tavoitteita ja odotuksia?* (Ranta, 2019)

Tuotannonohjaukselle asetetaan yhä enemmän odotuksia. Toisaalta tuotannonohjauksen rakenne on myös riippuvainen tuotantolaitoksen tai yleisesti ympäristön automaatiovalmiudesta. Pienimmässä mittakaavassaan tuotannonohjaus on osittain automaattisten linjojen ja varaston hallintaa PLC:llä. Isommassa mittakaavassa tuotannonohjaus on monitahoinen järjestelmä, jonka varaan on rakennettu yrityksen ydintoiminnot.

Mobiilirobotit muodostavat yhden osan tehtaan sisälogistiikkaan. Sisälogistiikka taas luo perustan tuotannon sujuvuudelle. Oikea-aikaisesti suoritettu materiaalinhallinta ja siirtäminen vähentävät hukkaa ja lisää varaston kiertoa. Sisälogistiikan liittyminen tuotannonohjaukseen tukee reaaliaikaista varastonhallintaa sekä tuotteiden jäljitettävyyttä tehtaan sisäisissä toiminnoissa.

Voidaan siis määritellä asiakastarpeeksi muovautuvuus hyvin vaihteleviin olosuhteisiin ja ohjelmistoihin. Tuotannonohjauksesta tulee löytyä, joko valmius REST protokollan mukaisten viestien välittämiseen tai järjestelmään pitää rakentaa olemassa olevan (ei REST) rajapinnan ja kolmannen osapuolen palvelinohjelmiston kautta yhteys mobiilirobottiin tai niiden ohjausjärjestelmään.

Tässä tapauksessa odotusarvona on, että ohjelmisto tukee avointa REST API -rajapintaa tai asiakkaalla on mahdollisuus ottaa käyttöön tiedonsiirtoa ohjaava kolmannen osapuolen ohjelmisto. Toimittajien arviointi ja haastaminen nousevat tärkeäksi osaksi ratkaisun yksityiskohtaisessa määrittelyssä ja implementoinnissa asiakkaalle.

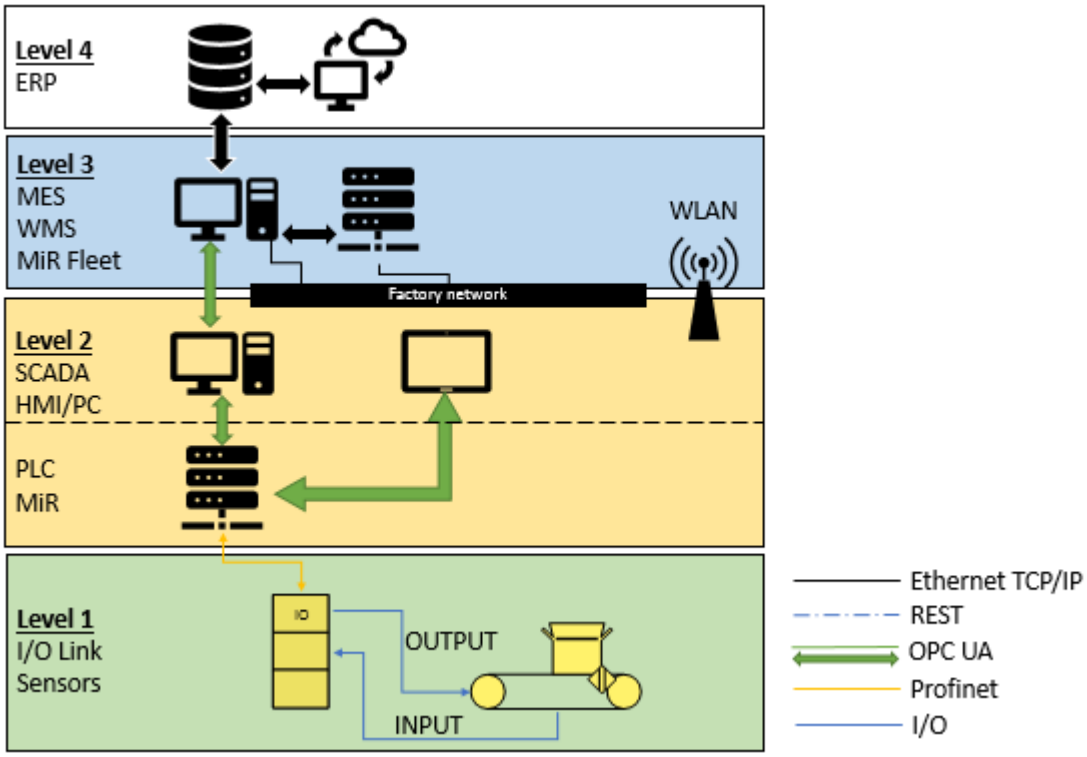
2.3 Ohjausjärjestelmien hierarkia

Mitä yksinkertaisemmin ja reaaliaikaisemmin voimme antaa tietoa tuotantoa ohjaavalle järjestelmälle, sitä paremmin ohjaukseen liittyvät muut tasot toimivat. Voidaan siis vaikuttaa reaaliaikaisesti aikatauluihin ja tehdä päätöksiä vallitsevien tilanteiden mukaan.

Tuotannonohjaus liittyy Ethernet verkkoon, joka on TCP/IP protokollan mukainen tehdasverkko. Tehdasverkko on itsessään erotettu palomuurein omaksi kokonaisuudeksi, jossa on omat serverit ja kytkimet. Palomuureilla pyritään huolehtimaan tietoturvasta. Palomuureja on verkossa myös yksittäisillä koneilla, jolloin palomuureista syntyy portaittainen suoja tietoverkon kautta kohdistuviin uhkiin. (LEWIS [2], 2018)

Teollisenympäristön tiedonsiirtoa ja järjestelmää kuvaavassa ISA-95 hierarkiamallista saa selkeän kuvan erilaisista käytettävistä tiedonsiirtoverkoista. Mallin virallinen nimi on *ANSI/ISA-95* ja se kuvaa

toiminnanohjausjärjestelmän (ERP), tuotannonohjauksen (MES) ja ohjausjärjestelmien (SCADA, PLC) integraatiota ja termistöä. Mallilla pyritään kuvaamaan isoa järjestelmää ymmärrettävässä muodossa. (Siemens, 2020)



Kuva 4. ISA-95 -mallia mukaillen. (Ranta [1], 2020)

Alimmalla tasolla yksi (1) kuvauksessa esitetään prosessin ohjausta ja tilatietojen tunnistamista, kyseessä on I/O-taso. Tasolla kaksi kuvataan ohjausjärjestelmät, jotka ohjaavat ja lukevat tasolta yksi (1) olevia laitteita ja sensoreita. Tasolla kaksi (2) on siis koneohjaus eli PLC ja prosessiohjaus eli SCADA. SCADA on kuitenkin tasolla kaksi (2) ylempänä kuin PLC, koska se ohjaa koko tuotantoprosessia koneiden osalta.

Tasolla kolme (3) tuotannonohjausjärjestelmä eli MES, joka hyödyntää koneiden osalta SCADA:n dataa tasolta kaksi (2) ja ohjaa henkilöiden kuormitusta tuotannossa, materiaaleja kuin tuotantotietojen välitystä toiminnanohjausjärjestelmään tasolle neljä (4). Tasolla neljä (4) on siis koostavana järjestelmänä yritystason ohjaus eli toiminnanohjausjärjestelmä ERP.

Tasojen yksi (1) ja kaksi (2) välillä tiedonsiirtoa tehdään melko usein Profinet TCP/IP protokollalla, joka on riittävän nopea ja perustuu Ethernet TCP/IP protokollaan, mutta on kehitetty teollisuuden kenttäväyläksi. Profinet on suljettu ja omalla IP avaruudella määritelty alusta, jossa jokaisella laitteella on kiinteä IP osoite. Väylälle on myös määritetty vasteaika. Vasteaikaan vaikuttaa myös PLC:n ohjelmakierto ja ohjelmarakenne.

Tasolla kaksi (2) ja tasojen kaksi (2) ja kolme (3) välillä tiedonsiirto tapahtuu yleensä hyödyntäen turvattua OPC UA protokollaa. OPC UA avulla voidaan varmistaa tiedon oikeellisuus (mistä-minne

menossa) ja turvallinen tiedonsiirto järjestelmien välillä. Tason kaksi (2) ja kolme (3) välille on rakennettu jo yleensä palomuureja, joilla pyritään suojaamaan tuotantolaitoksen toiminta kaikissa tilanteissa.

Tason kolme (3) ja neljä (4) välillä järjestelmä toimii Ethernet TCP/IP verkossa. Tasolla neljä (4) oleva toiminnanohjausjärjestelmä toimii yritystasolla, joka isossa mittakaavassa voi tarkoittaa järjestelmän tietojen sijaitsevan toisessa kaupungissa tai maassa, jolloin julkisessa verkossa voidaan lähettää tietoa palomuurien kautta TCP/IP protokollaa hyödyntäen.

2.4 Mobiilirobotti

IoT Agendan sivustolla mobiilirobotti määritellään koneeksi, jota ohjaava ohjelma hyödyntää sensoreita ja muita teknologioita havainnoidakseen ympäristöään ja liikkuakseen siellä. Mobiilirobotit hyödyntävät niin perinteisten robottien elementtejä kuin tekoälyä toiminnassaan. Mobiilirobotteja voidaan jaotella erialisiin ryhmiin niiden toimintaympäristön mukaan. Esimerkiksi UAV:t toimivat ilmassa ja UGV:t toimivat maassa. (Brush, 2019.)

Kuva 5. MiR Hook 200 mobiilirobotti. (Mobile Industry robots 2020)

Perinteiset niin sanotut vihivaunut (AGV) eivät lukeudu mobiilirobotteihin. Vihivaunujen reitit ovat ennalta määrättyjä. (SOLVING, 2020) Mobiilirobottien ja vihivaunujen eroa voidaan ajatella seuraavan esimerkin kautta:

Vihivaunu vie tavaran paikasta A paikkaan B: *Vihivaunun reitille on jätetty osittain lava tielle. Vihivaunu hidastaa ja pysähtyy, koska skanneri huomaa esteen reitillä. Ihmisen täytyy poistaa este ja mahdollisesti kuitata vaunu jälleen liikkeelle.*

Mobiilirobotti vie tavaran paikasta A paikkaan B: *Mobiilirobotin reitille on jätetty osittain lava tielle. Mobiilirobotti skannaa koko ajan ympäristöään ja laskee että sille sallitulla alueella voi kiertää*

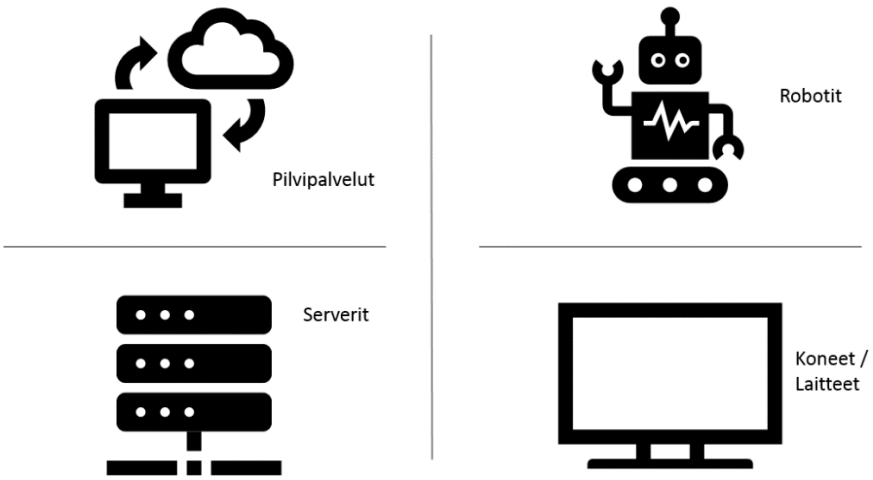
esteen ja jatkaa matkaa. Robotti väistää esteen ja vie kuormansa perille ilman että ihminen puuttuu asiaan. Käyttäjä voi tarkastella etänä robotin skannaamaa ympäristöä ja sen muutoksia ja huomata että kielletylle alueelle on jätetty tavaraa, joka tulee siirtää.

Edellä olevista esimerkeistä käy ilmi mobiilirobotin tekemien päätösten vaikutus toimintaan verraten vihivaunun rajattuun mahdollisuuteen tehdä päätöksiä. Mobiilirobotti siis hyödyntää tekoälyn avulla kaiken datan, minkä se saa ympäristöstään sensoreillaan sekä esisyötettyinä tietoina.

MiR eli Mobile Industry Robots on Tanskassa perustettu logistisia yhteistyörobotteja eli mobiilirobotteja valmistava kansainvälinen yritys. Robottien kehityksessä on keskitytty ketterään käyttöönottoon, turvallisuuteen ja helppokäyttöisyyteen. Yksittäisten robottien käyttöönotto luonnistuu nopeasti mobiilin web-pohjaisen käyttöliittymän avulla, joka mahdollistaa robotin ohjaamisen niin puhelimella kuin tietokoneella. Tuoteperheeseen kuuluu robotteja 100kg maksimikuormasta aina 1000kg maksimi kuormaan. Lisäksi toimittaja tarjoaa erilaisia toiminnallisuutta lisääviä osia robotteihin (lava pisteet, latauspisteet), kuten myös ohjelmiston hallinnoimaan useampia robotteja eli MiR Fleet sovelluksen. (MiR [2], 2020 & MiR koulutus kevät 2019)

2.5 REST -arkkitehtuurimalli

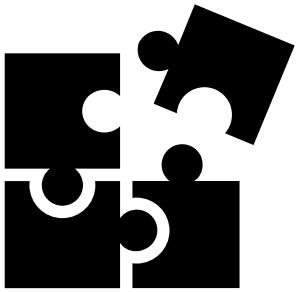
REST on arkkitehtuurimallin nimitys internet pohjaisille rajapinnoille. Roy Fielding on luonut alkuperäisen määritelmän 2000 luvulla. Rajapinnalle on määritelty joukko ehtoja, jotka sen tulee täyttää. REST on tilaton, mikä tarkoittaa rajapinnan vain välittävän tietoa varsinaisille sovelluksille. (Mikkonen, 2017) REST välittää dataa verkossa yksinkertaisessa muodossa. API toimii käyttäjäohjelman ja REST protokollan tulkkina, joka vastaanottaa sekä välittää dataa ohjelmalle käsiteltäväksi.



Kuva 6. Erilaisia laitteita ja järjestelmiä, jotka käyttävät verkon rajapintoja.

API ohjelmointirajapintana voi olla vaikea hahmottaa. Kuvan 6 avulla voi havainnollisesti ajatella mitä API on ja mitä se ei ole. API ei siis ole palapelin yksittäinen pala vaan se on osa jokaista palapelin palaa. Kun palapelin palat on yhdistetty, toimii rajapinnat keskenään. API on toisin sanoen

tässä esimerkissä palapelin reunat, jotka liittyvät toisiin paloihin, kun muodot vastaavat toisiaan. Tuo sauma on sitten REST protokollan mukainen väylä, jossa tieto kulkee. Itse palapelin pala kokonaisuutena edustaa laitetta tai virtuaalista ohjelmaa, joka toimii itsenäisesti muihin nähden, olipa kyseessä robotti, PLC tai vaikka verkkoselain.



Kuva 7. Yksinkertaistettu ajatusmalli API:n ymmärtämiseksi.

Ohjelmistotoimittaja Visman blogissa 4.2.2019 Kari Heikkilä tiivistää API:n määritelmän seuraavasti: ” Ohjelmointirajapinta (engl. Application programming interface, API) on määritelmä, jonka mukaan eri ohjelmat voivat tehdä pyyntöjä ja vaihtaa tietoja, eli keskustella keskenään. Rajapinnan dokumentaatiosta ja testiaineistoista ei peritä maksua, mutta palvelun varsinaiseen tietosisältöön käsiksi pääsemisestä voidaan periä maksu tai pääsy voi edellyttää esimerkiksi palvelusopimusta.” (Heikkilä, 2019)

3 TIEDONSIIRTOTEKNIIKAT

Tietoverkko rakentuu palvelimista (server) ja asiakkaista (client). Palvelimet ovat tietoverkossa useamman asiakkaan käytössä olevia resursseja. Palvelimet ovat tehokkaita ja ne voivat toimia myös asiakkaina. Asiakas on taas palvelinta hyödyntävä, ohjelma, tietokone tai muu laite, jonka resurssit ovat huomattavan paljon rajallisemmat. (Dummies, 2020) Asiakas on aktiivinen komponentti palvelimen ollessa passiivinen. Palvelin siis vaatii jonkin triggerin toimiakseen, joka saadaan asiakkaalta. (Schiekofer, 2018, s.1)

Internet pohjaiset palvelut kuten REST API toimivat http rajapinnan kautta. Jotta rajapinta saadaan toimimaan, on ymmärrettävä mitkä laitteista tai ohjelmista toimivat asiakkaina tai palvelimina. MiR Fleet toimii palvelimena, joten esimerkiksi uudentehtävän julkaiseminen vaatii, että lähettävänä osapuolena on REST asiakas. (MIR, 2017, MIR [1] & [2] & [3], 2020) Vapaita alustoja tarvitaan siis rakentamaan siltoja toisistaan poikkeavien protokollien välille.

3.1 OSI-mallin kerrokset

OSI tulee sanoista Open System Interconnection ja se on viitekehys tiedonsiirtojärjestelmien arkkitehtuurille. Malli on julkaistu 1984 kansainvälisen standardointi organisaation ISO:n toimesta. Malli kuvataan kerroksina, jossa jokaisella kerroksella on omat ominaispiirteensä. Mallia voi käyttää myös verkkopohjaistenjärjestelmien vian selvittämisessä, jolloin pinoa lähdetään tutkimaan alimmasta kerroksesta ylempiin. OSI-mallissa on seitsemän kerrosta ja jokaisella kerroksella on oma tehtävänsä ja erilaiset verkko järjestelmät käyttävät verkon eri kerroksi. (Shaw, 2018)

OSI-mallin alin kerros on fyysinen ja se pitää sisällään kaiken verkkoon liittyvän fyysisen ja sähköisen verkontoiminnon linkit ja kaapeloinnit (radiotaajuudet ja kaapeleissa kulkevat signaalit). Todennäköisin syy verkon ongelmiin löytyy yleensä täältä. (Shaw, 2018)

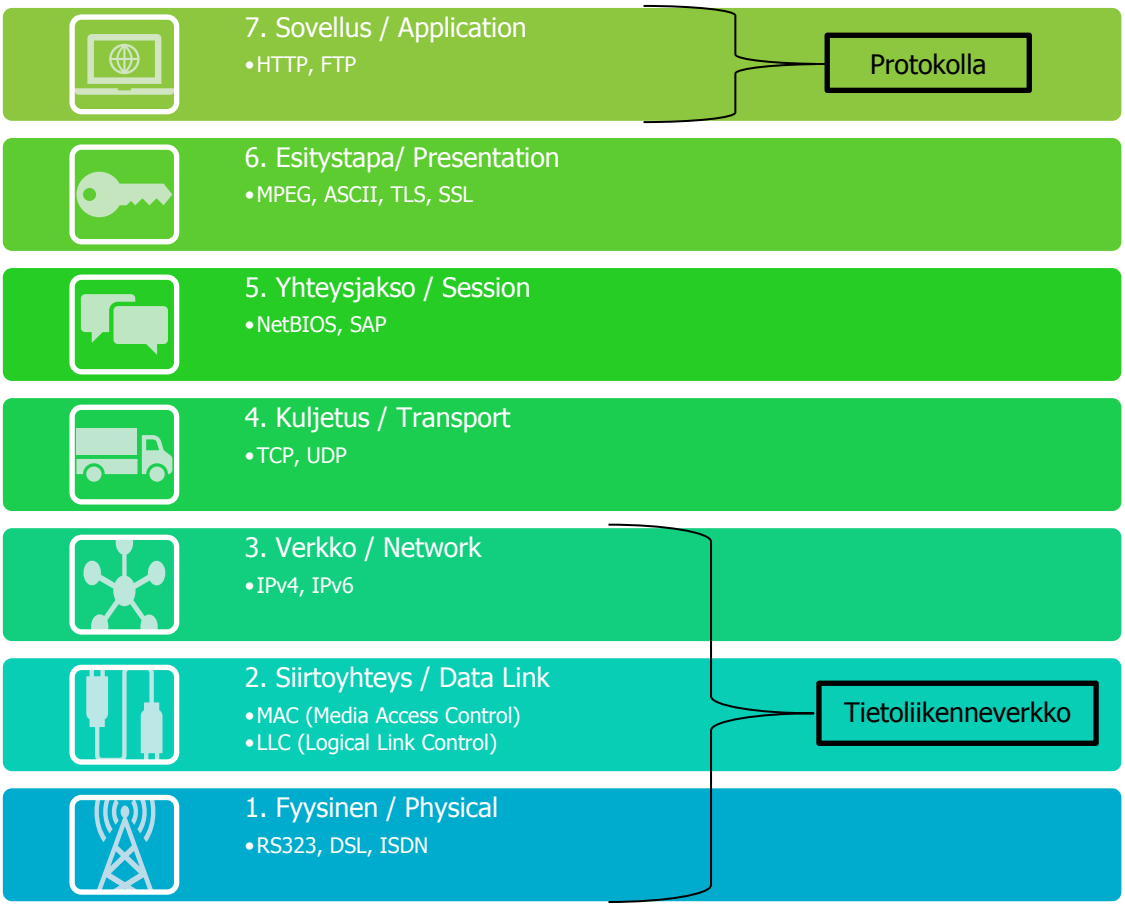
Siirtoyhteyskerros mahdollistaa kahden noodin välisen tiedonsiirron. Tällä toiseksi alimmalla kerroksella on myös omat alikerrokset: MAC ja LLC. MAC-kerros hoitaa tiedonsiirron kanavoinnin verkossa ja LLC-kerros huolehtii fyysisestä tiedonsiirrosta ja linjan tiedonsiirtotavasta (protokollasta). (Forcepoint, 2020) Useimmat kytkimet (Switches) toimivat tässä kerroksessa. (Shaw, 2018)

Verkkokerroksessa toimivat reitittimet, jotka ohjaavat verkonliikennettä tehokkaasti haluttujen osoitteiden välillä. Osoitteina toimivat tässä kerroksessa esimerkiksi internet protokollan mukaisesti IP osoitteet. (Forcepoint, 2020)

Kuljetuskerros hallinnoi järjestelmien ja palvelimien välisestä tietueiden kuljettamisesta. Tätä mallia käyttää esimerkiksi TCP-protokolla, yleensä internet protokollan päälle rakennettuna (TCP/IP). Edellä mainitussa mallissa kuljetuskerroksessa toimivat porttien numeroinnit, kun taas IP osoite käyttää alempaa verkkokerrosta. (Shaw, 2018.)

Yhteysjaksokerroksessa huolehditaan kahden laitteen välisien istuntojen luomisesta esimerkiksi koneen ja palvelimen tai kahden palvelimen välille. Kerroksen kautta hoidetaan myös yhteyden muu hallinta ja määrittelyt. (Shaw, 2018)

Esitystapakerros toimii itsenäisenä datan toistimena. Se kääntää verkosta saatavaa dataa sovelluskerrokselle tai toisinpäin sovelluskerroksesta verkkoon. Turvallisen tiedonsiirron varmistamiseksi data voidaan salata tai salaus voidaan purkaa, mikäli lähettävän ja vastaanottavan järjestelmän esitystapakerros omaa tarvittavat tiedot tähän toimintaan. (Shaw, 2018)

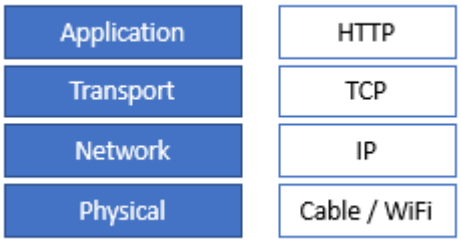


Kuva 8. OSI-mallin visualisointi (Mukaillen Shaw, 2018 & ALA-MUTKA, RINTALA, SAVIKKO, PALVIAINEN, 1996-2002)

Sovelluskerroksessa toimiviin sovelluksiin käyttäjillä on suora yhteys. Tässä kerroksessa toimivat sovellukset ovat suorassa vuorovaikutuksessa käyttäjäsovellusten kanssa. Käyttäjäsovelluksia ovat esimerkiksi selaimet ja ohjelmat, joita käytetään erilaisilla päätteillä. (Shaw, 2018)

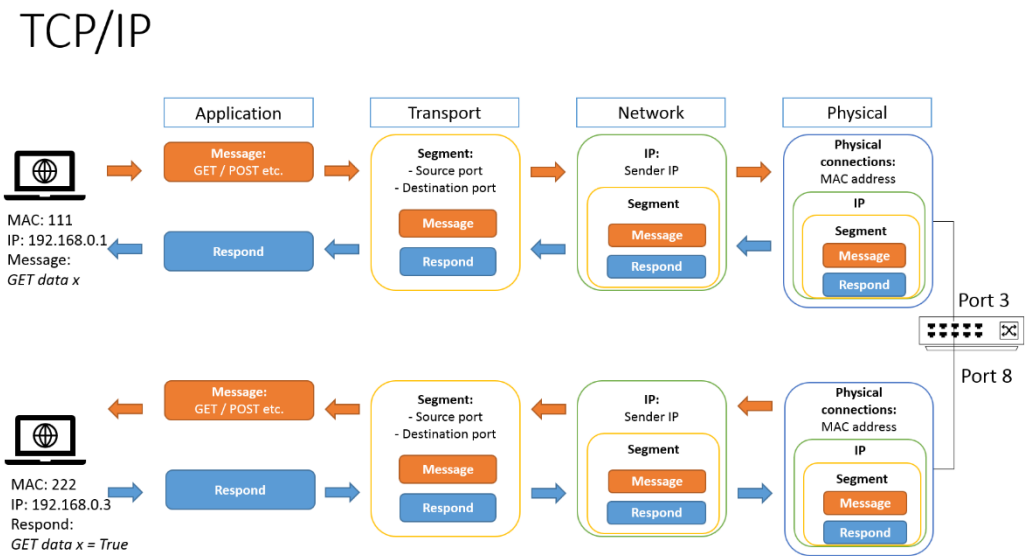
3.2 TCP tai UDP /IP

TCP/IP-mallin muodostamaa rakennetta kutsutaan protokolla pinoksi. OSI-malli on luotu korvaamaan alun perin juuri TCP/IP-malli, kuitenkin mallin jo laajalle levinnyt käyttö jätti sen edelleen käyttöön. (Wikipedia [1] & [2], 2020)



Kuva 9. TCP/IP-mallin kerrokset ja ominaisuudet. (Mukaillen LEWIS, 2018.)

TCP ja UDP ohjaavat eli välittävät tietoa järjestelmästä omien porttiensa kautta. Molemmilla tavoilla on käytössään reilu 65 000 porttia. Jokaiselle yhteyden vaativalla sovelluksella on oma porttinsa käytettävissä. Portit liitetään osaksi lähettävän ja vastaanottavan IP osoitetta. Lähetettävä data pilkotaan ja pilkotuille osille annetaan TCP/UDP header, jonka alla paketissa kulkeva tieto segmentoidaan mukaan. TCP otsikkoon tulee tiedot niin lähettävästä portista kuin vastaanottavasta portista. Tämän jälkeen segmentoitu paketti liitetään osaksi IP protokollaa. (Lewis, 2018)



Kuva 10. TCP/IP mukainen tiedon välitys. (Mukaillen LEWIS, 12.3.2018.)

TCP varmistaa yhteyden lähettävän ja vastaanottavan laitteen välillä, kun taas UDP on yhteydetön. Ohjelmat, joilla on oma järjestelmä huolehtimassa tiedon vastaanottamisesta ja sen oikeellisuudesta tai ovat nopeuden suhteen kriittisiä, käyttävät UDP:ta. TCP ja UDP eroavat toisistaan niiden käyttämän data määrän mukaan. UDP ei säilytä pakettien järjestystä vaan lähettää paketit nopeinta reittiä ja paketit ovat luettavissa sitä mukaa kun ne ovat saapuneet perille. Lähettävä osapuoli ei kuitenkaan saa koskaan tietoa onko paketti saapunut perille. (Lewis, 2018)

TCP-paketit järjestyvät lähetysmuotoonsa vastaanottajalle. TCP-paketti myös vaatii kuittauksen vastaanottajalta, että paketti on saapunut perille. Mikäli tätä varmistusta ei tule ilmoittaa järjestelmä virheestä tiedonsiirrossa, yleensä aikakatkaisun mukaan eli vastauksen tulee saapua tietyssä ajassa. (Lewis, 2018) Esimerkki tällaisesta tilanteesta on nettisivun lataaminen suojatussa yhteydessä (pankkipalvelut), palvelin ilmoittaa aikakatkaisusta, mikäli paketti ei saavu perille ajoissa.

Kuvassa 10 on yksinkertaistettu esimerkki TCP /IP -mallin tiedonsiirrosta. Koska TCP vaikuttaa erityisesti kuljetus (Transport) kerroksessa ja API:t toimivat sovelluskerroksessa (application), voidaan REST -rajapinnan viestiä siis viedä verkossa ohjelmien välillä hyödyntäen TCP/IP -protokollaa. Kuvassa 11 on kuvattuna http-protokollan toiminta käyttäjän näkökulmasta.

Kuva 11. HTTP-protokollan toiminta käytännössä. (Ranta [1], 2020.)

3.3 OPC UA

OPC standardi on kehitetty teollisuusautomaation luotettavaan ja turvalliseen tiedonsiirtoon ja se on julkaistu käyttöön 1996. Perinteiset kenttäväyläratkaisut kuten Profibus ja Modbus eivät soveltuneet juurikaan käyttöliittymien ja prosessiohjauksen tarpeisiin. OPC:lla luotiin edellytykset PLC pohjaisiin järjestelmiin liittymiselle ja niiden hallinnalle uusista laitteista ja järjestelmistä. Tämä uudistus mahdollisti myös loppukäyttäjän käyttöönottaa järjestelmiä käyttäen parhaita tuotteita, jotka toivat tietoa OPC:n kautta. OPC tulee sanoista OLE (object linking and embending) for Process Control eli objekti pohjainen prosessi ohjaus. Standardi on alun perin kehitetty Windowsin käyttöjärjestelmän käyttöön. (OPC Foundation [1], 2020) Teollisuus PC:t automaatiojärjestelmien käyttöön ovat usein Windows pohjaisia, mikä on seurausta Windows pohjaisesta ohjelmoinnista.

OPC UA on OPC, jolla on laajennettu Unified Architechture ominaisuus eli alusta riippumattomuus. Vuonna 2008 lanseerattu UA versio tukee mikropiiripohjaisia sekä pilvipohjaisia järjestelmiä, kun alkuperäinen oli sidottu Windows käyttöjärjestelmään. Tämä kehittyneempi versio pyörähtää toimintaan esimerkiksi Linux, Android ja Apple OSX käyttöjärjestelmissä. (OPC Foundation [2] & [3], 2020) Molemmat OPC mallit toimivat asiakas/palvelin (client/server) mallilla eli asiakas on ohjelma tai ohjelman osa, joka kysyy tietoja tai pyytää niiden päivittämistä palvelimelle.

OPC UA voi hyödyntää tiedonsiirrossa niin binääri -, hybridi - kuin WebService pohjaisia palveluita. Binäärinen toimii parhaiten, koska se käyttää vähiten resursseja. Binäärisessä protokollassa UA käyttää omaa TCP porttia 4840, jonka kautta tiedonsiirto tapahtuu turvallisesti. (Ascolab [1], 2020)

OPC UA:n ominaisuudet määritellään OSI malliin sijoitettuna sovelluskerroksessa (application layer), jossa se käytännössä varmistaa yhteydessä olevan laitteen ja käyttäjän autentikoinnin ja salasanat. Mallin tiedonsiirtokerroksessa (transport layer) on huomioitu turvallinen tiedonsiirto laitteiden välillä salausten ja koodikielen avulla. (Ascolab [2], 2020)

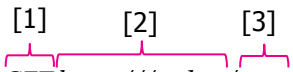
3.4 REST

REST tai RESTful (Representational State Transfer) on ohjelmisto pohjainen arkkitehtuuri malli, jolla voidaan luoda web pohjaisia sovelluspalveluita eli rajapintoja. REST pyrkii rakentamaan yhteydet verkkokomponenttien roolien mukaan jättäen huomiotta fyysisen liitynnän ja protokollien syntaksit. Se on kehitetty vähentämään tiedonsiirron määrää ja viiveitä verkossa. (Fielding, 2000 s. 2-3)

REST luo tilattomien palveluiden säännöstön, jonka mukaan palveluita voidaan rakentaa. Kaikki resurssit ovat käytettävissä ja tunnistettavissa URI koodin avulla, joka on itseasiassa merkkijono. REST:llä kuvataan yksinkertaisia rajapintoja, jotka hyödyntävät tiedonsiirrossa standardoituja rajapintoja. REST ei siis pidä sisällään mitään erillistä viestitasoa. Resurssin rajapintaan päästään kiinni URI:n avulla http metodeilla, josta resurssi sitten palauttaa toiminnon mukaisen vastineen tai virhetiedon asiakkaalle (Client). (Lahoti, 2019)

Arkkitehtuurilla on kuusi rajoitetta tai sääntöä, joiden perusteella määritellään, onko kyseessä REST. Sopimuksen mukainen liityntä (uniform interface), määrittää asiakkaiden ja palvelimien välistä sopimusta. Sopimuksen mukaan molempien osapuolien pitää pystyä kehittymään toisesta riippumatta tai siitä huolimatta. Kommunikaatio perustuu rajapintaan, joka on resurssipohjainen, käsittelee resursseja kuvauksen mukaisesti itseään kuvaavilla viesteillä. Hyperteksti toimii sovellusmoottorina rajapinnalle, jonka avulla asiakas voi lukea vastauksien yksityiskohtaisempia tietoja. (Lahoti, 2019)

Protokollan ollessa tilaton, lähettää asiakas kaikki tarvittavat tiedot palvelimelle, jolloin palvelimen ei tarvitse ylläpitää istuntoa asiakkaan ja palvelimen välillä. (Lahoti, 2019) Lahotin artikkelissa esitetty esimerkki asiakkaan lähettämästä pyynnöstä palvelimelle ja sen vastauksesta asiakkaalle. Asiakkaan lähettämä pyyntö tilauksesta 1234. Pyynnössä ensimmäisessä osassa [1] GET on toiminta verbi, toinen osa [2] on URI ja lopussa on kohdennus osa [3]:



(Lahoti, 2019)

Palvelin lähettää asiakkaalle navigointilinkin, joka on voimassa, kunnes asiakas on saanut kaikki nimikkeet, jotka kuuluvat tilaukselle 1234. Navigointi linkki eli palvelimen vastaus asiakkaalle on seuraavanlainen:

```
{
  id : 1234,
  any-other-json-fields...,
  links": [
    {
      "href": "1234/items",
      "rel": "items",
      "type" : "GET"
    }
  ]
}
```

(Lahoti, 2019)

Välimuistia hyödynnetään palvelimen resurssi varausten vähentämiseen ja verkon nopeuden nostamiseen, joka johtuu vähentyneestä tiedonsiirto tarpeesta, kun vastaus kulkee vain kerran pyyntöä kohden. (Lahoti, 2019)

Rakenteellisesti asiakkaalla ja palvelimella on omat tehtävänsä: asiakas huolehtii yhteydestä palvelimelle ja palvelin huolehtii tiedon varastoinnista ja ylläpidosta. Jokaisen kerroksen on toimittava itsenäisesti ja voivat olla vuorovaikutuksessa ainoastaan niiden kerrosten kanssa, jotka ovat suorassa yhteydessä toistensa kanssa. Mikäli useat asiakkaat käyttävät saman resurssin tuottamaa palvelinta, voidaan välissä käyttää myös välitin palvelinta, joka tällöin ohjaa pyynnöt ja viestit oikeille asiakkaille takaisin. (Lahoti, 2019)

REST käyttää http protokollastakin tuttuja toiminta komentoja (action verbs). Jokaisen pyynnön mukana sen on siis lähetettävä toimintaverbi (REST verb), Otsikko tiedot (Header information) eli URI ja kokoelma tiedot (body). Kuusi yleisintä toimintaverbiä ovat GET, POST, PUT, PATCH, DELETE ja OPTIONS. Jokaiselle verbin suoritukselle on http protokollasta tulevat status koodit onnistuneelle (success) yhteydelle ja epäonnistuneelle (failure) yhteydelle. (Lahoti, 2018)

Taulukko 1. REST verbit ja niiden selitykset. (Lahoti, 2018)

REST Verb	**Action**	**Success**	**Failure**
GET	Fetches a record or set of resources from the server	200	404
OPTIONS	Fetches all available REST operations	200	–
POST	Creates a new set of resources or a resource	201	404, 409
PUT	Updates or replaces the given record	200, 204	404
PATCH	Modifies the given record	200, 204	404
DELETE	Deletes the given resource	200	404

Vastausten mukana tulevien statuskoodien sisältö määräytyy koodiryhmän ja alatunnuksen mukaan. 2XX koodit ovat onnistuneen yhteystilan päätteksi tulevia vastauksia. 4XX sarjan koodit taas ovat standardeja asiakas virheitä. 5XX sarjan koodit ovat palvelin virheitä. Näiden tiedostaminen auttaa ymmärtämään yhteyden rakentamisen ongelma kohdissa, missä vika mahdollisesti on. (Lahoti, 2018)

4 TIEDONSIIRRON RAJAPINNAT JA TUOTANNONOHJAUKSEN LEAN -FILOSOFIA

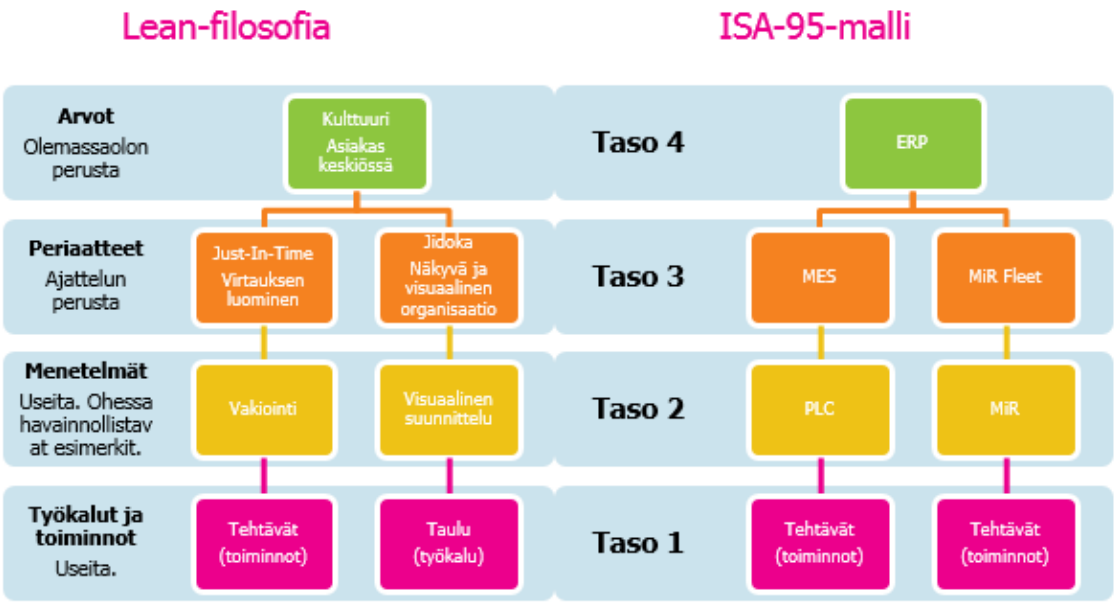
Tuotannonohjauksella pyritään parhaaseen mahdolliseen tulokseen pienimmillä mahdollisilla panoksilla. Tämä tarkoittaa, että kaikkien tuotannon osa-alueiden tulee toimia saumattomasti yhteen ompa kyse sisälogistiikasta tai yksittäisen tuotantolinjan toiminnasta.

ISA-95 -mallin mukaiset hierarkian osat voidaan yhdistää lean-filosofiaan, sillä ne ovat hyvin saman oloisia ja niissä pätevät osin samat lainalaisuudet. Kuvassa 10 on esitettynä samassa kuvassa niin

kappaleessa kaksi (2) esitetty lean-mallin abstraktitasojenkuvaus sekä ISA-95 -malli. Näiden mallien sitominen toisiinsa edistää molempien aspektien toteutumista lean maailman tuodessa ihmismäisen lähestymistavan automatisoituun tuotantoon.

4.1 Järjestelmä hierarkian ja filosofian sulautuminen

Kuvassa 13 ylimmällä eli yritystasolla määritellään yrityksen arvot ja mittarit, joita vasten arvojen toteutumista mitataan. Toiminnanohjausjärjestelmällä (ERP) mitataan ja seurataan näiden arvojen toteutumista, myös suurimman arvon tuottamaa asiakkaan kokemusta asiakastietojärjestelmän kautta.



Kuva 12. Lean-filosofian ja ISA-95 -mallin yhteys. (Ranta [1], 2020)

Tasolla kolme (3) luodaan periaatteita, joiden mukaan toimintaa ohjataan. Periaatteet pohjautuvat arvoihin ja niiden toteutumista seurataan aina toiminnanohjausjärjestelmässä asti mittarein, jotka edelleenkin kuvastavat yrityksen arvoja. Periaatetasolla yrityksessä tulisi kuvata prosessit sekä tehdä tavoitteet näkyväksi. Tuotannonohjaus on työn suunnittelua ja tulosten näkyväksi tekemistä mitä suurimmissa määrin.

MiR Fleet, joka asettuu johonkin tason kolme (3) ja kaksi (2) välille, edustaa sekä suunnitelmallista virtausta JIT (Just-In-Time) että Jidokan edustamaa visuaalisuutta. MiR Fleet toimii reaaliaikaisena sisälogistiikan tuotannonohjauksena, joka keskustelee useampaan suuntaan, myös käyttäjille vaunujen havaitsemista toimintaa haittaavista poikkeamista ympäristössä.

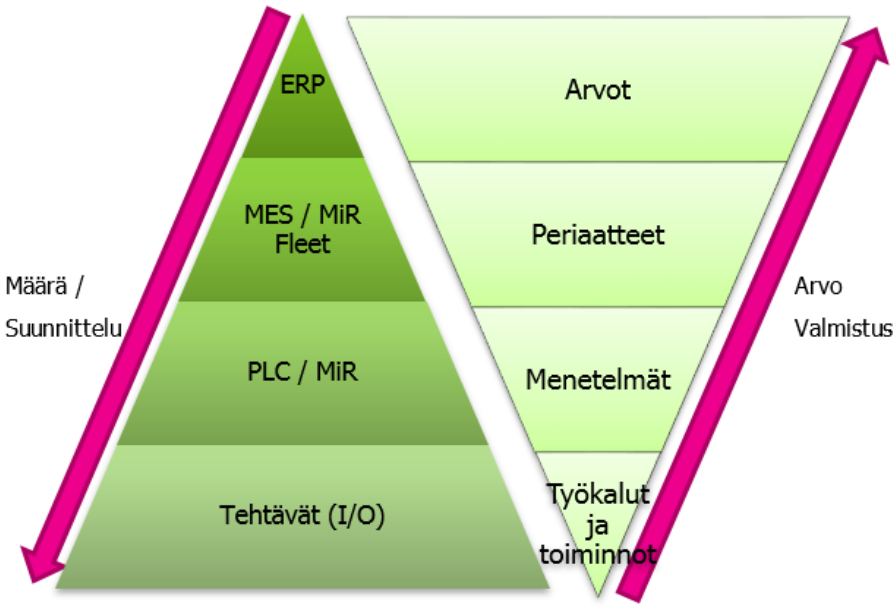
Tasolla kaksi lean -filosofiassa ovat edustettuna menetelmät ja niissä esimerkkeinä vakiointi ja Visuaalinen suunnittelu. Perinteiset automatisoidut tuotantolinjat edustavat vakiointia. Niissä tehdään rutiinin omaisia toimia (toistoja) koneilla tai koneiden avustuksella. Visuaalisuutta tuotannossa tuodaan koneille erilaisina tila- ja ajotietoina, joilla kerrotaan koneiden suoriutumisesta sille asetetuista tavoitteista ja antaa siten inputin myös käyttäjille tarvittavista toimista tavoitteen saavuttamiseksi.

Edelleen tasolla kaksi (2) MiR vaunut edustavat älykästä automaatiota, joka kykenee tekemään päätöksiä sille sallittujen rajojen sisällä edelleen sen toimiessa myös joustavasti ja ihmislähtöisesti erilaisissa tilanteissa. Ne edustavat sekä vakiointia että yllättävissä tilanteissa visuaalisten ja paikannukseen perustuvien elementtien avulla tapahtuvaa päätöksen tekemistä yhdessä ohjaavan järjestelmän kanssa. Järjestelmä voi siis löytää keinoja parantaa tuottavuutta pienillä panoksilla ilman ihmisen puuttumista tehtäviin. Toisaalta se edelleen mahdollistaa päätöksien rajaamisen ihmisille.

Alimmalla tasolla yksi (1) ollaan vahvasti tekemisessä kiinni. Tällä tasolla Lean määrittää kuinka toimitaan yksittäisissä tehtävissä, jotta tikapuita kiipeämällä tavoitetaan asetetut tavoitteet ja sitä myöden myös toteutetaan yrityksen arvoja, jotka asiakas on sille osin generoinut. Tällä tasolla ilmenevät niin päivittäisjohtamisentaulu kuin yksittäisten tuotannon tehtävät eri pisteillä, riippumatta suorittaako tehtävät kone vai ihminen. MiR robottien osalta tasolla yksi ovat sen sensorit, apulaitteet ja koko sen fyysinen olemus yksittäisiksi toiminnoiksi purettuna. Toisin sanoen räjäytyskuva kuinka eri tuotannon osat toimivat.

4.2 Määrän ja arvon suhde

Kuten lean -filosofiaa tulisi ajatella Toyotan oppien mukaan, arvot luovat koko toiminnan pohjan eli se on maa, josta puu kasvaa. Kuvassa 14 esitettynä mallien yhteneväisyys määrän tai suuruuden mukaan. Oikealla kuvassa Lean -filosofia ja vasemmalla ISA-95 -malli. Tämä kuvastaa kuinka filosofia toimii reaalimaailman peilinä.



Kuva 13. Merkitystä kuvaavampi mallinnus. (Ranta [1], 2020)

Reaalimaailmassa järjestelmien näkökulmasta tehtävätasolla on määrällisesti eniten erilaisia komponentteja. Näiden komponenttien suhde leanin arvon määritykseen on pieni, mutta jos kolmiokärki kuvaa yksittäisen tehtävän arvoa on kaikkien tehtävien tuottama arvo huomattavasti isompi.

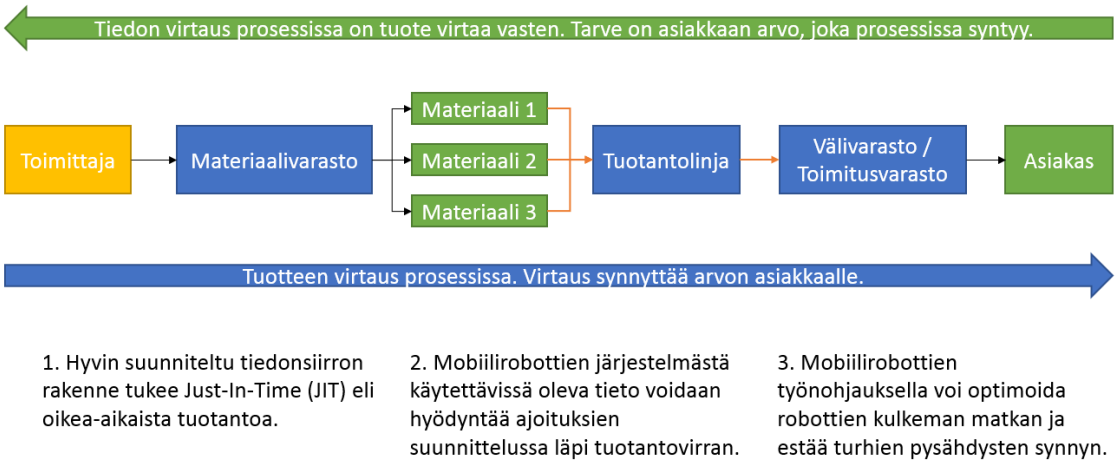
Komponenttien määrä pienenee menetelmätasolla se kuitenkin kerää yksittäisten tehtävien tuottaman arvon yhteen. Arvoa tuotetaan siis aina kohti seuraavaa tasoa tai työvaihetta. Samalla kun arvoa tuotetaan ja noustaan ylemmäs, muuttuu järjestelmä komponenttien määrä edelleen pienemmäksi.

Toisaalta voidaan ajatella niin että kuvassa vasen nuoli alaspäin kuvaa työnsuunnittelua tuotantoon ja oikea nuoli työn valmistumista tuotannosta. Arvoista tehtäviin ja tehtävistä arvoihin kurottaen.

4.3 Tiedonsiirronrajapinnat

Kuten edellä olevissa kappaleissa kuvattiin, kulkevat lean ja järjestelmät käsikädessä. Jokaisen tasonkin tulee toimia saumattomasti yhteen parhaan lopputuloksen saavuttamiseksi. Lopulta haluamme tuotannossa kuin tuotannossa tuottaa arvoa, josta asiakas on valmis maksamaan. Samalla haluamme mahdollistaa pienimmän vaivan tien eli tehdä vain välttämättömän arvoa tuottavan työn ilman ylimääräistä panostusta.

Kuvassa 15 esitetään tiedon virtaamista vasten tuotannon virtausta. Saatavat tiedot määrittävät tarpeen, jonka mukaan järjestelmät pyytävät omilta resursseiltaan lisää materiaaleja, jotta voivat lopulta toimittaa asiakkaan tilaaman tuotteen. Tätä ohjausmallia kutsutaan myös imuohjaukseksi, jossa tarve määrittää tehtävät toimet. Tässä mallissa hukka minimoidaan valmistamalla tuotteita menekin mukaan.

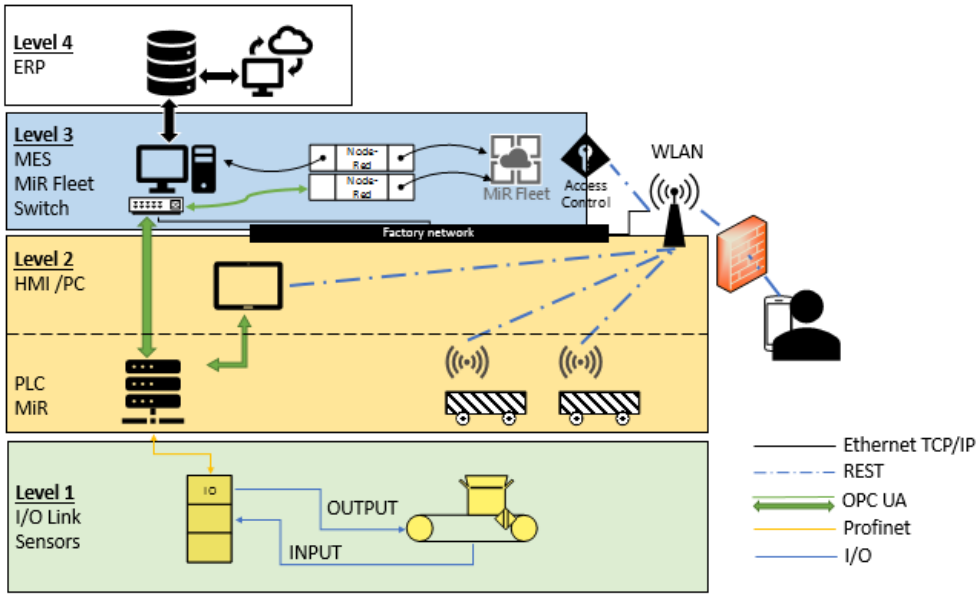


Kuva 14. Tiedon ja tuotannon virtaus. (Ranta [1], 2020.)

Tieto on siis valtaa myös tuotannonohjaamisessa ja mitä enemmän hyödynnämme saamaamme tietoa kaikilla järjestelmätasoilla, sitä paremmin voimme ennustaa ja suunnitella tulevia tapahtumia. Jotta tieto on ajantasaista ja sitä hyödynnetään tehokkaasti, on tiedonsiirto tällöin avainasemassa. Mobiiliratkaisut luovat teknisesti toteutettavissa olevan ekosysteemin, joka vahvistaa ja parantaa perinteisemmän automaation arvon tuottoa. API ratkaisut luovat mobiililaitteiden armeijan, joka toimii järjestelmien kaikilla tasoilla luoden niistä oman ennustavan ekosysteemin.

5 MOBIILIROBOTTIJÄRJESTELMÄN OHJAUSRATKAISU

ISA-95 malliin perustuen tehtaan automaatiojärjestelmä on hahmoteltu vastaamaan niissä käytettyjä yleisimpiä tiedonsiirron rakenteita. Esimerkissä ISA-95 mallin mukaisesti kuvataan tehdasta neljällä päätasolla. Esimerkissä tasolla neljä on toiminnanohjausjärjestelmä (ERP). Tasolla kolme tuotannonohjausjärjestelmä (MES), varastonhallinta (WMS) ja MiR Fleet eli mobiilirobottien tehdastason ohjausjärjestelmä, joka hoitaa tehtävien jakamisen roboteille.



Kuva 15. Järjestelmän kuvaus ISA-95 malliin pohjautuen. (Ranta [1], 2020)

Taso kaksi jaetaan kahteen osaan ylemmässä osassa valvomo-ohjelmisto (SCADA) ja valvomopäätteet (HMI). Tehtaan tietoliikenneverkko (tehdasverkko) linkittyy ylemmän tason ja tämän tason väliin. Tason alempi osa pitää sisällään itsenäiset laitteet, joilla ohjataan itsenäisinä kokonaisuuksina toimivia kenttälaitteita. Kuvaan on esitetty ohjelmoitavalogiikka sekä mobiilirobotit, jotka toimivat itsenäisesti määritettyjen ehtojen mukaan. Tasolla yksi esitetään yleisesti kenttähajautukset ja sensorit, joiden tilatietoja käytetään koneita ohjaavassa päätöksenteossa ylemmällä tasolla.

Edellä mainittu tehdasverkko toimii REST:n välittäjäverkkona. Mobiilirobottien osalta järjestelmään on kuvattu MiR robottien käyttöliittymä, joka toimii selaimen kautta. Itse laitteet voivat toimia tehtaan verkossa, mutta tiedot käyttöliittymälle kulkevat avoimeen rajapintaan, joka ei vaadi samassa verkossa toimimista. Avoimeen rajapintaan liittyminen tapahtuu palomuurien sekä autentikoinnin kautta, jolloin varmistetaan tietoturvallisuus järjestelmien ylläpidossa. Laitetasolle tieto kulkee tehtaassa aina palomuurien kautta.

5.1 Toiminnankuvaus

Järjestelmät generoivat tilaus ja ohjaustietoa ylhäältä alaspäin ja tilatietoa alhaalta ylöspäin. Asiakkaan tilaus käynnistää prosessin, jossa toiminnanohjausjärjestelmä varmistaa ensin tilauksen saata-

vuuden nykyisillä varastotiedoilla. Tilaus asiakkaalle toimitetaan joko varastosta tai suoraan tuotannosta, riippuen tuotannon ohjaustavasta. Mikäli tuotannon käynnistää varaston nimike määrien rajaarvot, on tuote varastotuote, muussa tapauksessa asiakaan tilaus generoituu suoraan tuotanto tarpeeksi.

Toiminnanohjausjärjestelmä tekee varauksen tuotannosta, jonka tuotannonohjausjärjestelmä ohjaa oikealle tuotantolinjalle, missä tuote voidaan valmistaa. Tuotantolinja tekee materiaalivarastosta tilauksen tarvittavista materiaaleista tuotannon valmistusjonoon.

Varasto valmistelee materiaalin ja lähettää mobiilirobottien MiR Fleet ohjausjärjestelmälle kuljetustilauksen. Riippuen työstä ja sen priorisoinnista, ohjautuu työ järjestelmässä joko jonon hännille tai prioriteetin mukaan ylemmäs työjonossa. Työ ohjautuu lähimmälle sopivalle robotille tehtäväksi.

Mobiilirobotti hakee materiaalin ja ilmoittaa samalla Fleet järjestelmälle kuljettavansa tilausta. Varastojärjestelmä ilmoittaa tuotannonohjaukselle, että materiaalit ovat matkalla. Fleet ilmoittaa tarvittaessa käyttäjille havaituista ongelmista kuljetuksessa. Kun mobiilirobotti jättää tilaukseen määränpäähän, ilmoittaa se jälleen tila muutoksesta Fleet järjestelmälle, joka kuittaa tehtävän suoritetuksi. Tuotantolinja ilmoittaa tuotannonohjausjärjestelmälle vastaanottaneensa tilauksen.

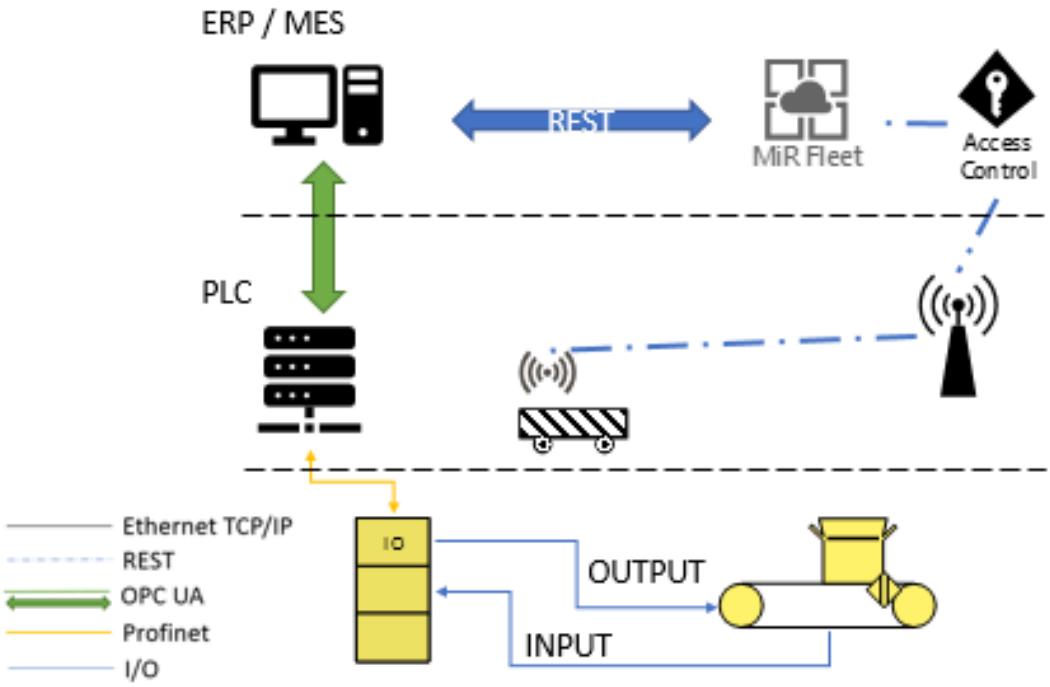
Kun tuotantolinjalta valmistuu tuotteita kuljetettavaksi toimitusvarastoon lähettää linjanohjaus tiedon valmistuneesta kuljetuserästä tuotannonohjaukseen ja kuljetus pyynnön Fleet järjestelmälle. Riippuen työstä ja sen priorisoinnista, ohjautuu työ järjestelmässä joko jonon hännille tai prioriteetin mukaan ylemmäs työjonossa. Työ ohjautuu lähimmälle sopivalle robotille tehtäväksi.

Mobiilirobotti hakee tilauksen ja ilmoittaa samalla Fleet järjestelmälle kuljettavansa tilausta. Tuotantolinja ilmoittaa tuotannonohjaukselle, että tilaus on matkalla. Fleet ilmoittaa tarvittaessa käyttäjille havaituista ongelmista kuljetuksessa. Kun mobiilirobotti jättää tilaukseen määränpäähän, ilmoittaa se jälleen tila muutoksesta Fleet järjestelmälle, joka kuittaa tehtävän suoritetuksi. Toimitusvarasto ilmoittaa tuotannonohjausjärjestelmälle vastaanottaneensa tilauksen.

Tarvittaessa tuotannonohjausjärjestelmä voi pyytää MiR Fleet järjestelmältä tilausten tilatietoja ja päivittää niistä tietoja käyttäjien nähtäville tuotantolinjoille, tai tuotantolinjoilla voi olla omat seuranta näkymät suoraan MiR Fleet järjestelmään, joilta voidaan seurata tuotantolinjaa palvelevia robotteja ja niiden tiloja.

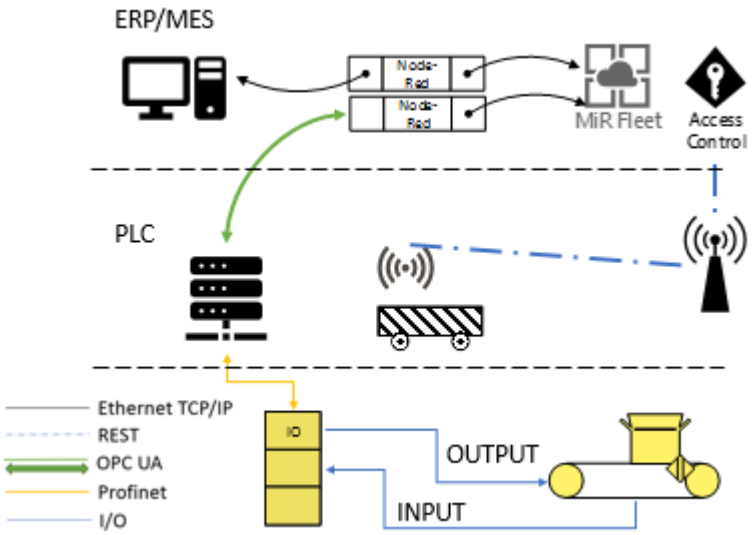
5.2 Kokonaisarkkitehtuuri

Tiedonsiirtoa järjestelmien välillä voi rakentaa useilla eri tavoilla. Järjestelmien rakentaminen on aina riippuvainen olemassa olevista laitteistoista ja niiden hyödyntämisen tasosta. Monitasoiset järjestelmät ovat käytössä pitkälle automatisoiduissa tehtaissa, joissa automaatio liittyy useimpiin toimintoihin tuotannossa. Kaksitasoinen kuvaa matalan automaation ympäristöjä tai tuotantomuotoja, joissa ihmiset vielä tekevät suurimman osan työstä manuaalisesti ja automaatio avustaa ainoastaan logistiikassa tai yksittäisissä toiminnoissa. Järjestelmissä keskeisintä ovat niin ohjauksen hierarkia kuin tuotantotiedon kuljettaminen järjestelmässä loogisesti sitä tarvitsevalle osalle.



Kuva 16. Järjestelmässä on kolme tasoa. Liittyminen tuotannonohjaukseen REST:ä käyttäen. (Ranta [1], 2020)

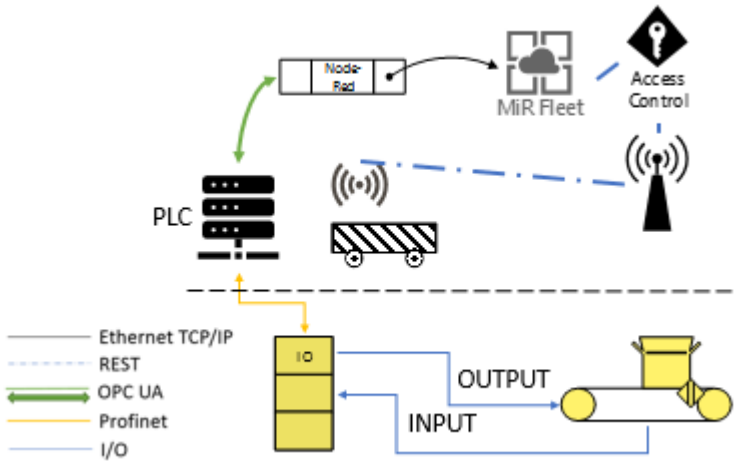
Ensimmäinen monitasoinen kuvaus esitettynä kuvassa 17 on yleisesti käytetty automatisoidun tuotannon mallia, joita on käytössä erikokoisissa tehtaissa. Ylimmällä tasolla Tuotannonohjausjärjestelmä (ERP/MES), johon MiR Fleet voidaan liittää uudemmissa järjestelmissä suoraan käyttäen REST protokollaa. Tässä tilanteessa järjestelmien linkittäminen on luonnollisinta, sillä perinteiseen operatiivisentason tiedonsiirtoon ei tarvitse kajota. Vaihtoehtoisesti vanhemmissa järjestelmissä välille on rakennettava tulkki tai tulkit, joiden avulla erilaiset tiedot ja käskyt siirtyvät järjestelmien välillä.



Kuva 17. Järjestelmä vaatii tulkin. (Ranta [1], 2020)

Yksinkertaisimmillaan liitettävät järjestelmät ovat yksittäinen PLC, johon MiR Fleet liitetään käytettävän PLC:n ehdoilla. Lähtökohtaisesti tällaiseen ratkaisuun päädytään, kun halutaan lähteä lisäämään automaatiota tehostamalla sisälogistiikkaa. Tällaisessa tapauksessa tuotantotiedot ovat kone - tai linja kohtaisia ja niiden järjestelmät poikkeavat toisistaan eikä keskitettyä tuotannonohjausta ole tai linjoja ei ole liitetty tuotannonohjaukseen. Edellä esitetty malli on kuvattuna kuvassa 18, joka on vahvasti manuaalinen toiminto tai yksittäisten automatisoitujen linjojen käytössä. Kuten korkeamman automaatiotason järjestelmissä voidaan myös käyttää PLC valmistajien omia laite ratkaisuja tulkin sijasta. Tällöin laiteratkaisu mahdollistaa REST-rajapinnan käytön suoraan.

Kuvassa 19 esitettynä ratkaisu, jossa keskitettyä tuotannon tai toiminnanohjausta ei ole ollenkaan. Tällöin yhteys rakennetaan esimerkiksi käyttäen yksittäistä PLC:tä ohjaavana elementtinä, jonka kautta työt ohjataan, joko suoraan tai tulkin kautta, MiR Fleetin keskitettyyn mobiilirobottien ohjaukseen. Periaatteessa tällaisella mallilla voidaan liittää myös useampia yksittäisiä linjoja ohjaukseen. Tällöin MiR Fleet ottaa työt vastaan saapumisjärjestyksessä ja ohjaa mobiilirobotit kulloisellekin tehtävälleen.



Kuva 18. Matalan automaation ratkaisumalli. (Ranta [1], 2020)

5.3 Valmiit alustat ohjelmisto- ja laiterajapintoina

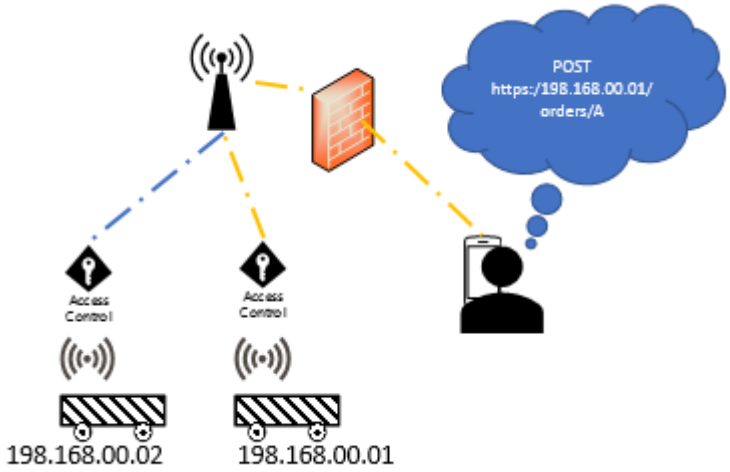
Erilaisia ohjelmistoja ja rajapintoja on kehitetty paljon, siksi on hyvä tunnistaa ennen uuden järjestelmän rakentamista, voiko olemassa olevia ratkaisuita hyödyntää uuden mallin tai toimintatavan käyttöönotossa.

Työn aikana tuli törmättyä moniin sovelluksiin kuten Postman sovellukseen, joka on yhteistyöalusta APIen kehittämiseen. Postmanilla saatiin rakennettua yksinkertainen tietoyhteys, mutta alustan käyttäminen olisi vaatinut parempaa ymmärrystä JSON ohjelmoinnista. Tästä syystä käytettäväksi valikoitui Node-Red palvelin ympäristö, jossa yksinkertaisen yhteyden rakentaminen ei vaadi syvällisempää JSON esitysmuodon ymmärrystä.

Järjestelmäkuvauksissa esitettyä teollista rajapintaa mallintamaan valittiin KepserverEX kehitysalusta, joka mahdollistaa erilaisten teollisten tiedonsiirtorajapintojen luomisen, myös REST-protokollan hyödyntämisen. MiR Fleet on mobiilirobottien oma keskitetty ohjauksensa, joka esiteltynä yleisellä tasolla.

5.3.1 MiR Fleet

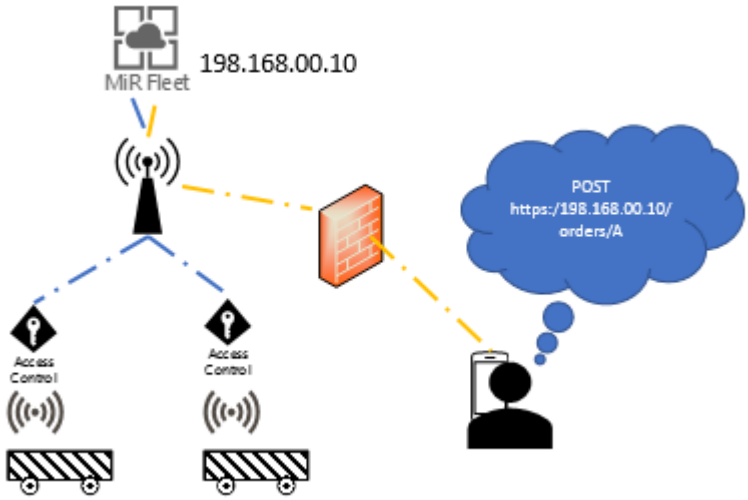
Yksittäistä robottia voidaan ohjata sen oman käyttöliittymän kautta. Käyttöliittymä on webpohjainen ja sillä on oma REST-asiakas rajapinta, joka voidaan ohjelmoida kyselemään tehtäviä aina vapautuessa serveriltä. Koska kyseessä on asiakas, se ei voi huolehtia useampien robottien saman aikaisesta toiminnasta. Tällainen järjestelmä vaatisi erillisen puskurin ja palvelimen osalta enemmän säätämistä tai robottien työalueiden ja tehtävien määrittelyn tiettyihin tuotannon osiin. (MiR koulutus kevät 2019, MIR [1] & [2] & [3], 2020)



Kuva 19. MiR vaunut ilman keskitettyä ohjausta. (Ranta [1], 2020)

Kuvassa 20 esitetään, miten robottien töiden ohjaaminen toimii ilman keskitettyä ohjausta. Jokaisella robotilla oma ohjausrajapinta, jolloin aina kun halutaan ohjata robotti suorittamaan tiettyä tehtävää pitää osata valita oikea robotti ja tietää sekä robotin IP osoite, että kirjautumistunnukset. Mikäli

yksittäisiä robotteja liitettäisiin osaksi isompaa tuotannonohjausjärjestelmää, pitäisi kaikille roboteille rakentaa omat yhteydet. (MiR koulutus kevät 2019, MIR [1] & [2] & [3])



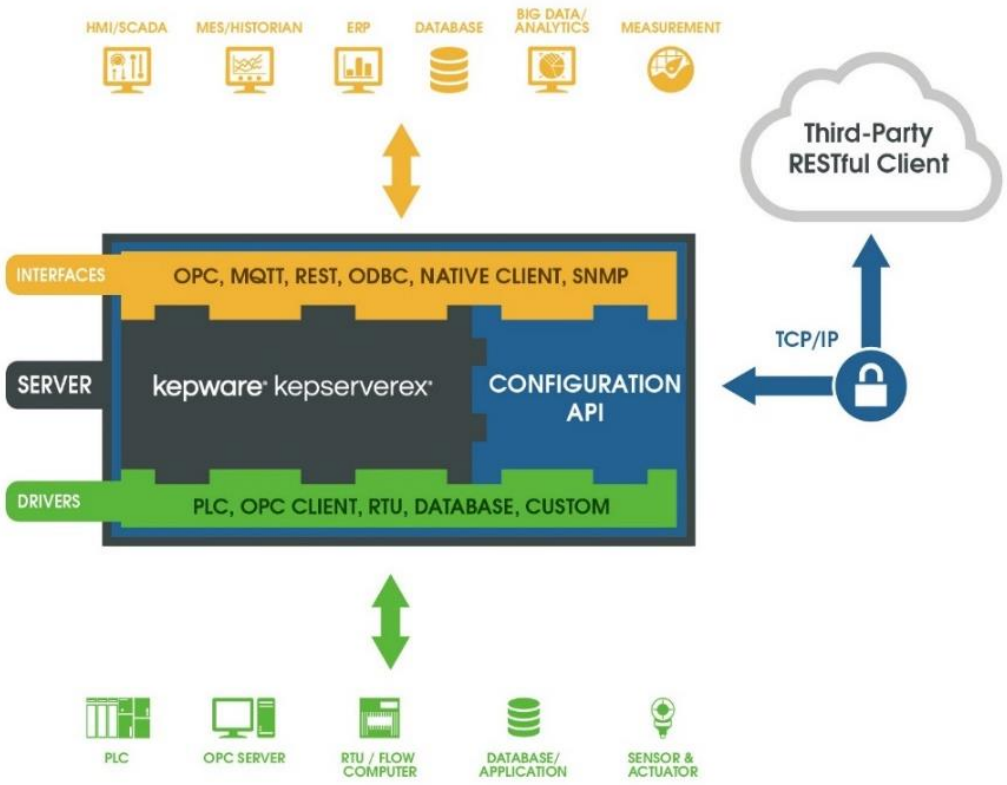
Kuva 20. MiR Fleetin avulla toteutettu keskitetty ohjaus. (Ranta [1], 2020)

MiR Fleet on mobiilirobottien oma ohjausjärjestelmä, joka hallinnoi tehtävät eri robottien välillä myös optimoiden muun muassa kuljettuja matkoja kuin akun varaustasoja. Ohjelmiston kautta voi hallita muun muassa robottien työskentely alueita. Samoin reaaliaikaisesti on mahdollista seurata mitä havaintoja robotit tekevät ympäristöstä, esimerkiksi esteistä reitiltä tulee tietoa heti ja niiden vaikutusta tuottavuuden heikkenemiseen voidaan minimoida korjaamalla havaitut poikkeamat välittömästi. Fleet toimii ikään kuin ajojärjestelijänä, joka huolehtii kaikkien tilausten täyttämisestä, joko saapumis- tai prioriteetti järjestyksessä. MiR Fleet on ulkopuoliseen ohjaukseen nähden palvelin, joten siihen liitytään asiakkaan REST -rajapintaa käyttäen. (MiR Fleet, 2020 & MiR koulutus kevät 2019, MIR [1] & [2] & [3])

5.3.2 KepserverEx

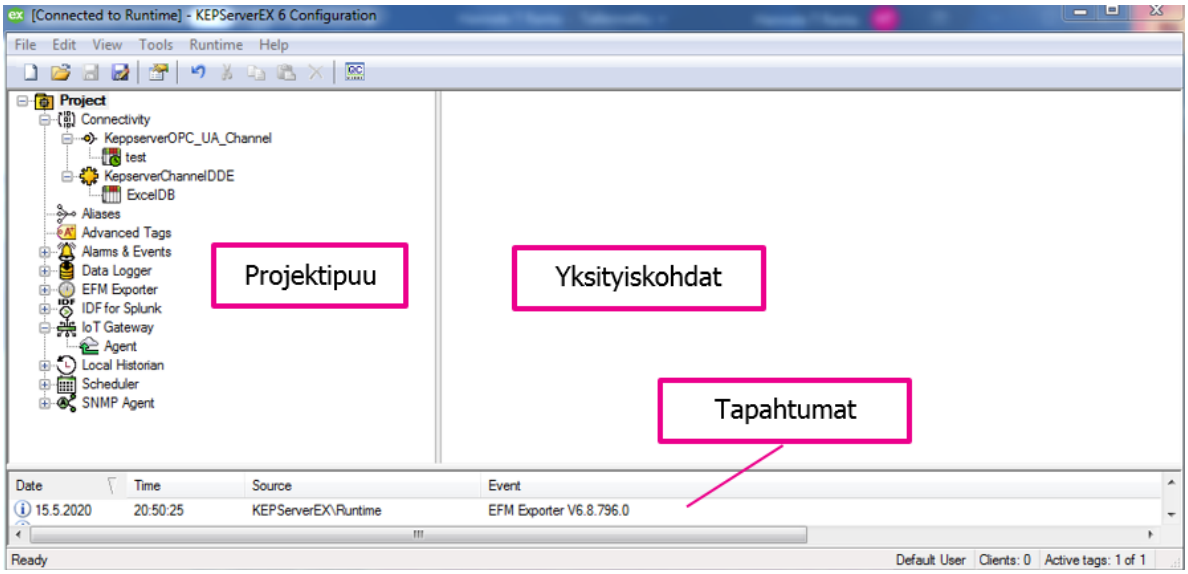
Kepware on 1995 perustettu ohjelmistoalan yritys, joka on keskittynyt erityisesti teollisuusautomaatio laitteistojen ja ohjelmien yhdistettävyyden kehittämiseen. Kepware on osa PTC yhtymää. Kepwaren ratkaisut ovat paljon teollisuudessa käytettyjä ja sertifioituja. (Kepware [1], 2020)

PTC:n Kepserverex alustaa käytetään rajapintana ja tiedon turvallisessa reitittämisessä tuotantolaitteiden ja tuotantoa ylätasolla ohjaavien laitteiden välillä. Kepserverex tarjoaa rajapinnat tuotannonsuuntaan muun muassa PLC:n ja OPC Clientin kanssa. Kolmannen osapuolen REST asiakkaita varten voidaan konfiguroida IoT portti, jolloin KepserverEx toimii serverinä. Kuvassa 22 Kepwaren esitys omasta rajapinta todellisuudestaan. (Kepware, 2017)



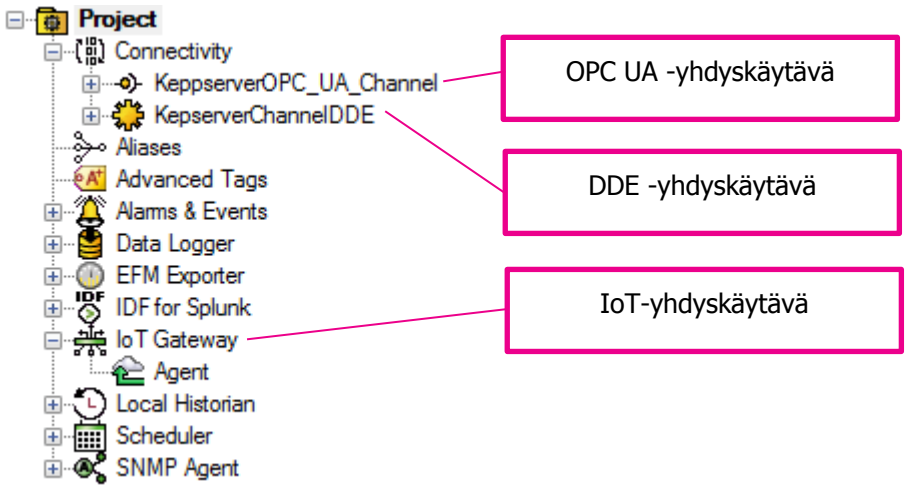
Kuva 21. KepserverEx:n rajapinnat. (Kepware [3], 2020)

Kepwaren KepserverEX on käytettävissä demoversiona, jolloin se vaatii aina uudelleen käynnistyksen toimiakseen. KepserverEX käyttöliittymässä tehdään kaikki asettelut yhteyksille erilaisiin rajapintoihin. Kuvassa 23 on esiteltynä projektipuu, yksityiskohtainen näkymä sekä tapahtumat. Liitynnät määritellään projektin alle. Projektille voidaan tehdä rajatusti koko projektia koskevia suojausasetuksia sekä käytettäville liityntä muodoille on omat asettelunsa. (Kepware [1] 2019, s.30-35)



Kuva 22. KepserverEX konfiguraattorin käyttöliittymä. (Ranta [2], 2020)

Projektipuuhun muodostuu kaikkien yhteys- ja datamuotojen valikoima, joiden alle luodaan eri laitteiden vaatimat yhteysratkaisut ja datan hallinnointi. Kuvassa 24 projektipuuhun on luotu OPC UA -, DDE –, ja IoT -yhdyskäytävät. (KEPWARE [2], 2020)

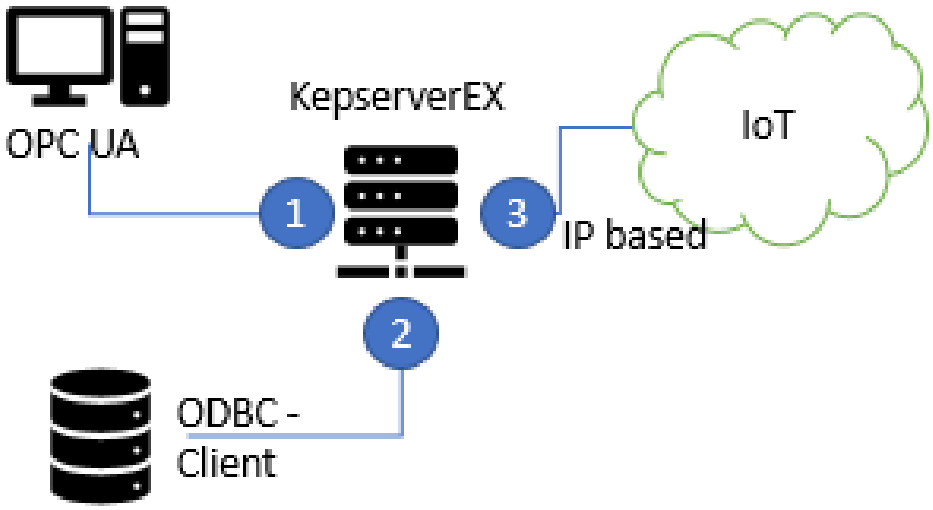


Kuva 23. Projektipuun yksityiskohtaisempia tietoja. (Ranta [2], 2020)

KepserverEX toimii välittäjänä määritellyille yhteyksille ja tallentaa tietoa palvelimelle tai esimerkiksi sen omaan dataloggeriin, josta tieto on nopeasti käytettävissä. Kuvassa 24 esiintyvä DDE-yhdyskäytävä on perinteinen Excelin tiedonsiirtoyhteys, joka korvautuu nyttemmin ODBC-yhdyskäytävällä. ODBC on yhteensopiva uudempien Excel järjestelmien kanssa, kun DDE toimii vanhemmilla Excelin 97-00 -tiedostomuodoilla ja ohjelmilla. (KEPWARE [2], 2020)

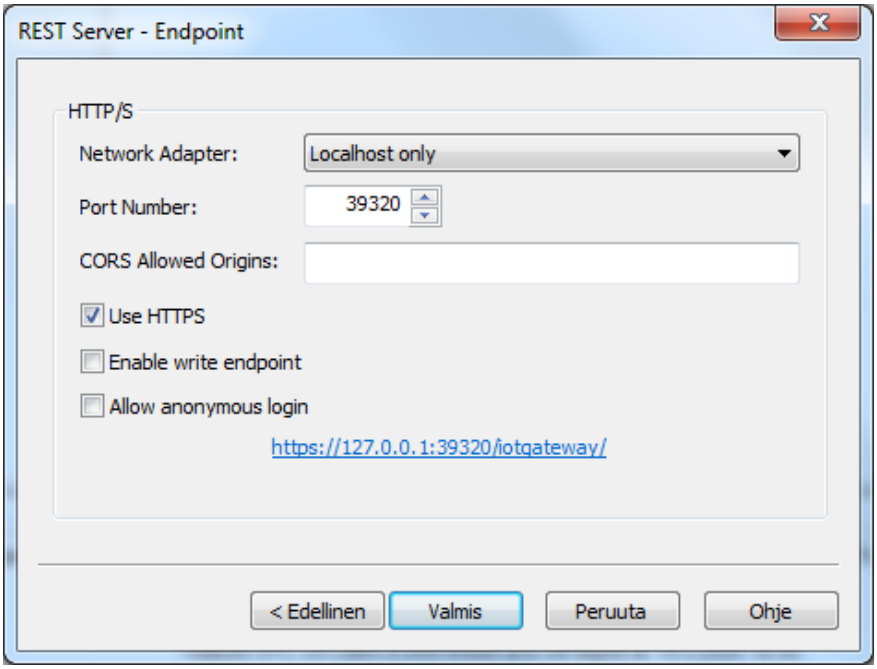
Kuvassa 25 on kuvattuna yksinkertaistettu topologia:

1. OPC UA -yhdyskäytävä.
2. ODBC -yhdyskäytävä.
3. IoT Agent -yhdyskäytävä.

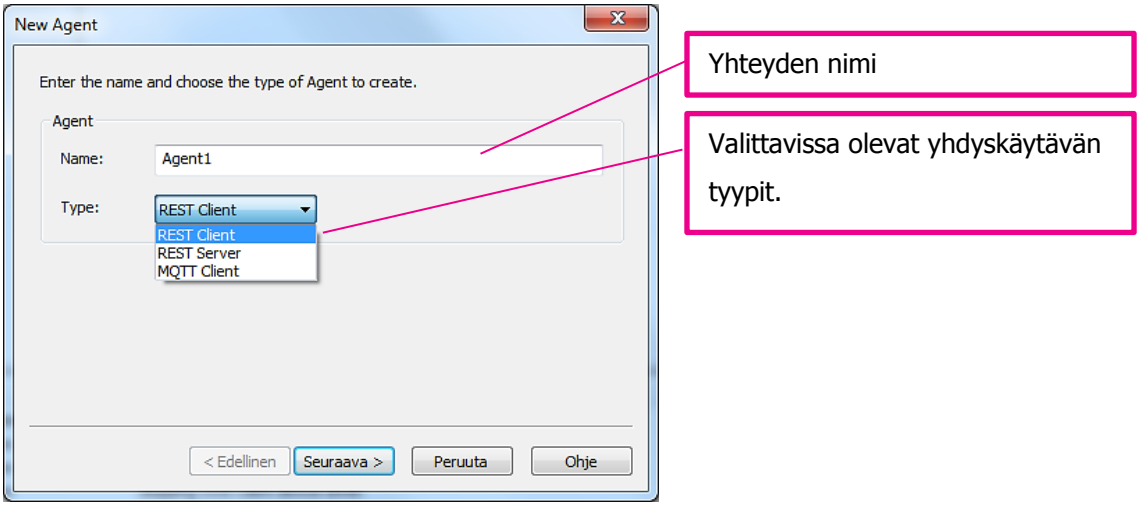


Kuva 24. Rajapintojen topologia. (Ranta [1], 2020)

IoT- yhteysväylään luodaan agentit, joiden avulla välitetään tai vastaanotetaan tietoa kolmannen osapuolen palveluntarjoajilta. Tässä osiossa Kepserver käyttää IP pohjaisia ratkaisuja. Valittavissa on REST pohjaisesti asiakas (Client) ja palvelin (Server), näiden lisäksi voidaan valita MQTT. Mikäli Kepserverillä käytetään REST-palvelinagenttia, tulee jokaisella erilliselle yhteydelle määrittää omat porttinumerot, mikäli ne johtavat samaan verkkosovittimeen. (Kepware [2] 2019, s. 6-10, 28) Kuvassa 26 on esitetty REST -palvelimen luominen KepserverEx ympäristöön.



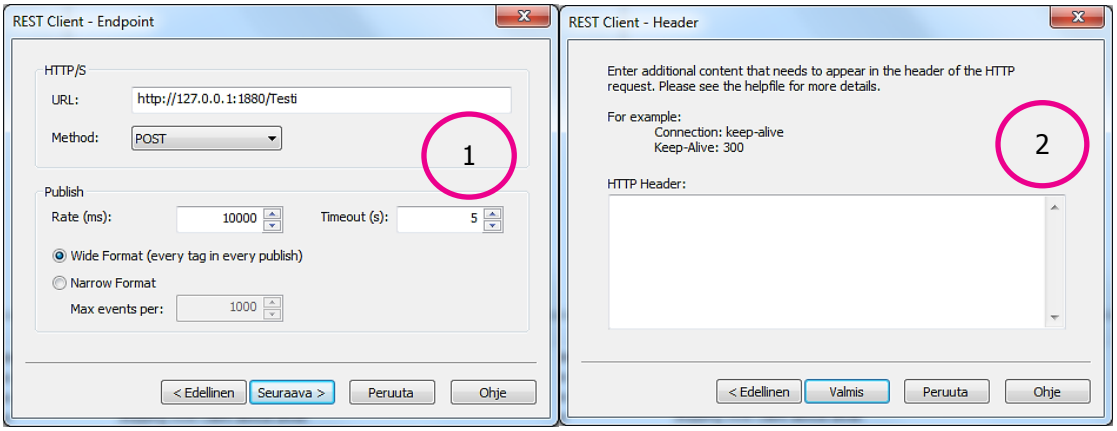
Kuva 25. REST-palvelinagentin asetukset luodessa palvelinpäätettä. (Ranta [2], 2020)



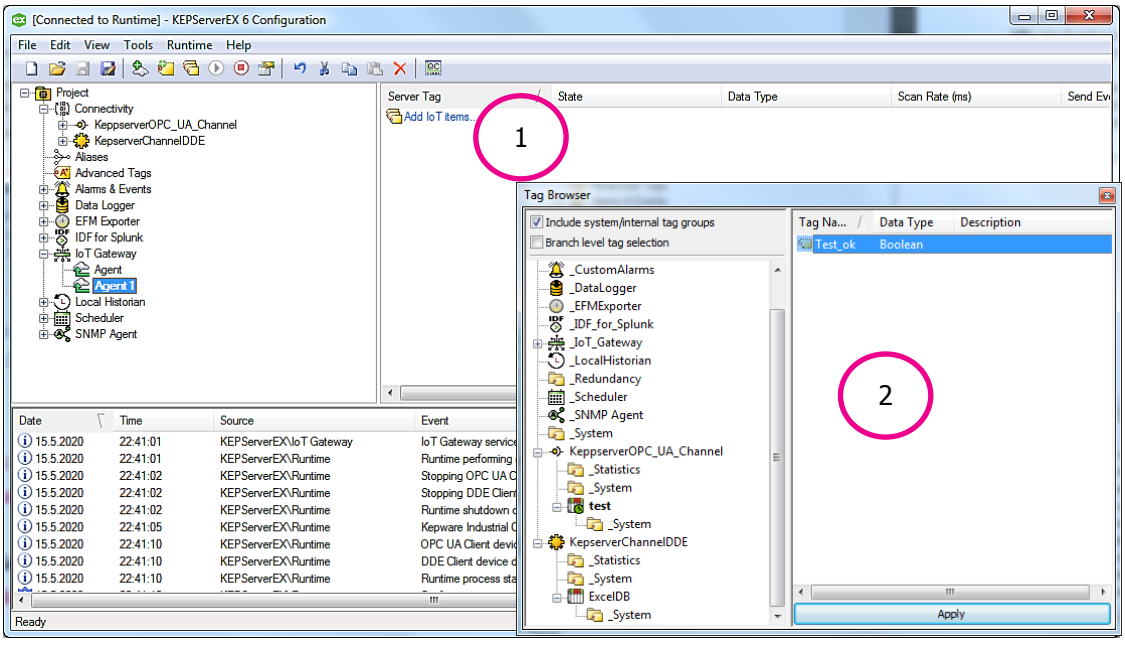
Kuva 26. Uuden yhteyskäytävän luominen. (Ranta [2], 2020)

REST -asiakas (Client) yhdyskäytävän avulla voidaan välittää tietoa REST päätepisteeksi määritetyn URI:n ja portin avulla. REST-asiakkaan luominen Kepserverillä on esitettynä kuvassa 27. Asetuksien loppuun saattaminen on esitetty kuvassa 28. URL määrittää kommunikoinnille päätepisteen ja julkaisu asetukset taas määrittävät kuinka usein tietoa päivitetään määritettyyn päätepisteeseen.

HTTP-otsikko -kentällä (HTTP header) voidaan välittää lisätietoja vastausten ja pyyntöjen mukana. HTTP-otsikot voidaan jakaa neljään ryhmään: Yleiset otsikot (general headers), pyyntö otsikot (request headers), vastaus otsikot ja entiteetti otsikot. Pyynnöt sisältävät tietoa haettavasta kohteesta tai pyynnön lähettävästä kohteesta. Vastaus otsikot sisältävät tarkentavia tietoja vastauksen lähettäjästä. Entiteetti otsikot sisältävät tarkempia tietoja kohteen rungosta. Yleiset otsikot sisältävät lähinnä pyyntöjen ja vastausten yleistietoja, joilla ei ole yhteyttä varsinaiseen lähetettyyn runkoon. (MDN web docs, 2020)

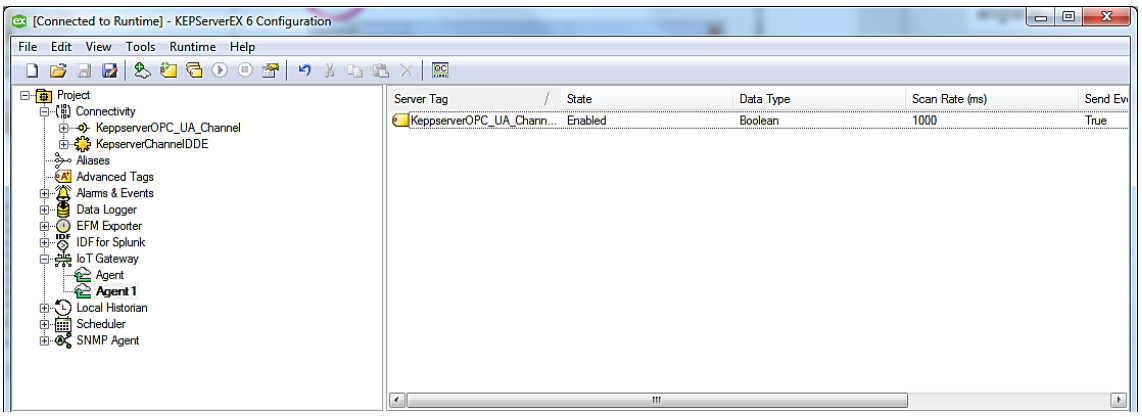


Kuva 27. Yhteyden ja päivitysasetusten luominen asiakkaalle. (Ranta [2], 2020)

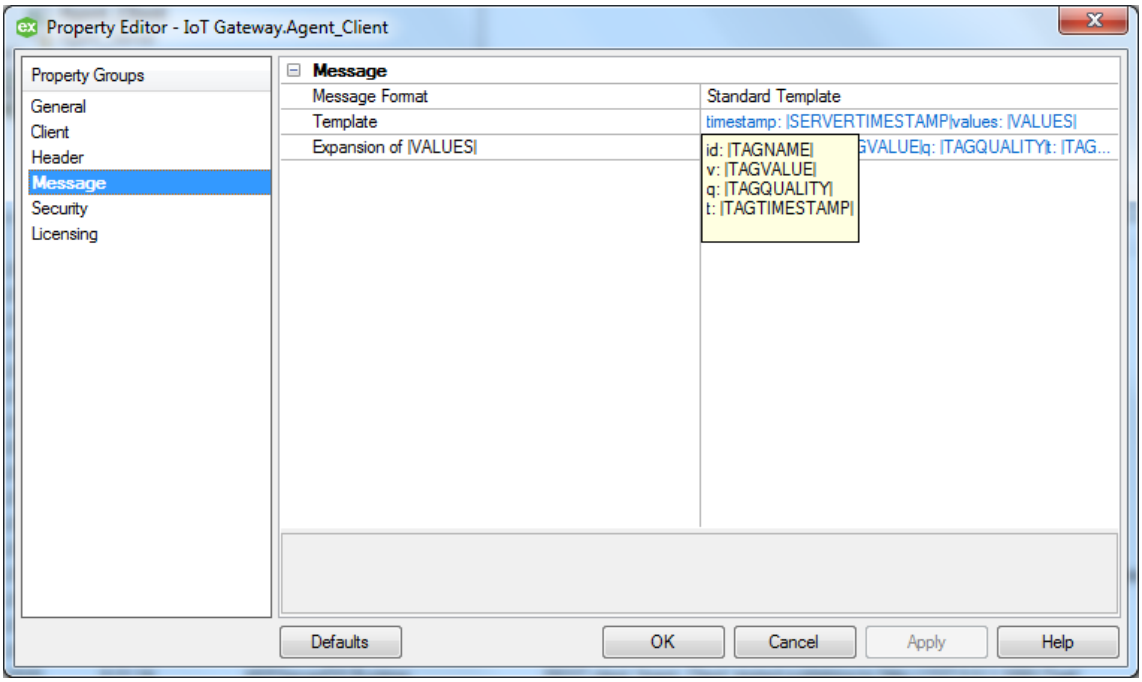


Kuva 28. Yhteyden luominen eri datalähteiden välillä. (Ranta [2], 2020)

Jotta tietoa saadaan välitettyä erilaisten yhteyspalveluiden välillä, tulee tiedot linkittää palveluiden käytettäviksi. Valitulle agentille lisätään nimikkeet (Add IoT items) kuvassa 29 kohdassa yksi (1). Projektille luotujen yhteyksien ja niiden sisälle rakennettujen tunnisteiden (tag) joukosta valitaan halutut datalähteet yhdistettäviksi kuvan 29 kohdassa kaksi (2). Kuvassa 30 nimike on lisätty agentille. Kuvassa 31 voidaan tarkastella asiakas agentin lähettämän viestin muotoa.



Kuva 29. Nimike luotu agentille ja yhdistetty alkuperäiseen nimikkeeseen. (Ranta, 2020)

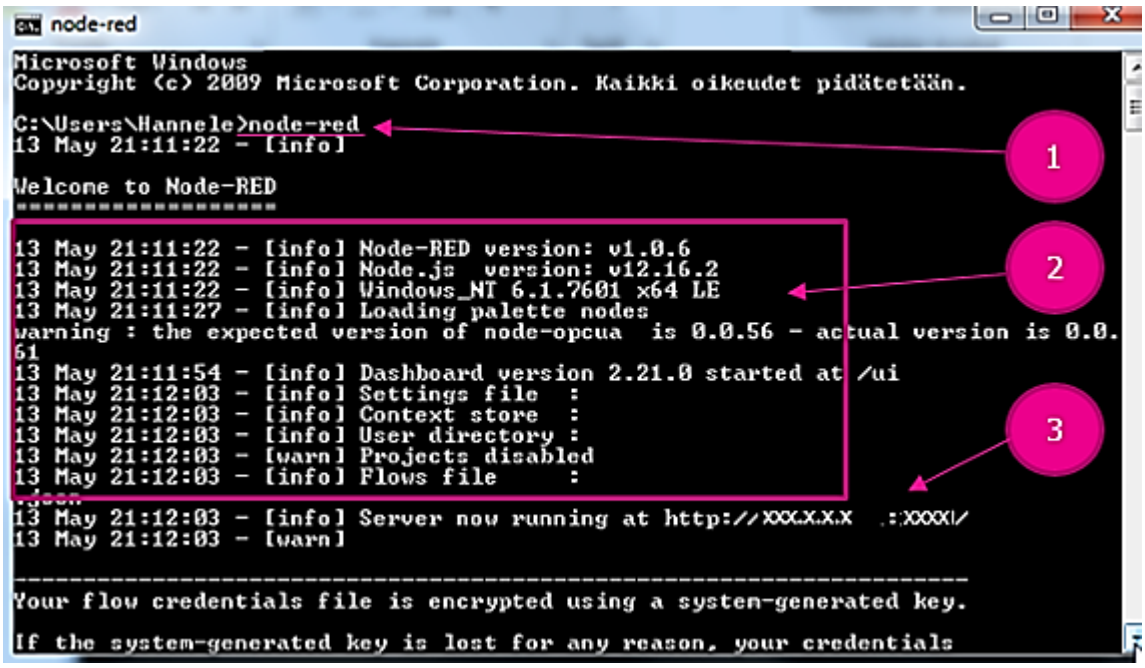


Kuva 30. Viestimuoto, jonka asiakas yhdyskäytävä generoi. (Ranta [2], 2020)

5.3.3 Node-Red

Node-RED on IBM:n tutkimus organisaation kehittämä vuokaaviopohjainen ohjelmointialusta. Nykyisellään Node-RED on osa JS Foundation kehitystyötä. Vuokaavio ohjelmointi perustuu Morrisonin 1970-luvulla tekemään kuvaukseen. Kuvauksen mukaan toiminnot verkossa on kuvattu mustilla laatikoilla tai solmuilla, joihin tuodaan tietoa ja jotka välittävät tiedon eteenpäin. Noodeilla on tarkasti määritelty tehtävä, jonka mukaan se käsittelee saamansa datan ennen eteenpäin lähettämistä. (Node-RED, 2020)

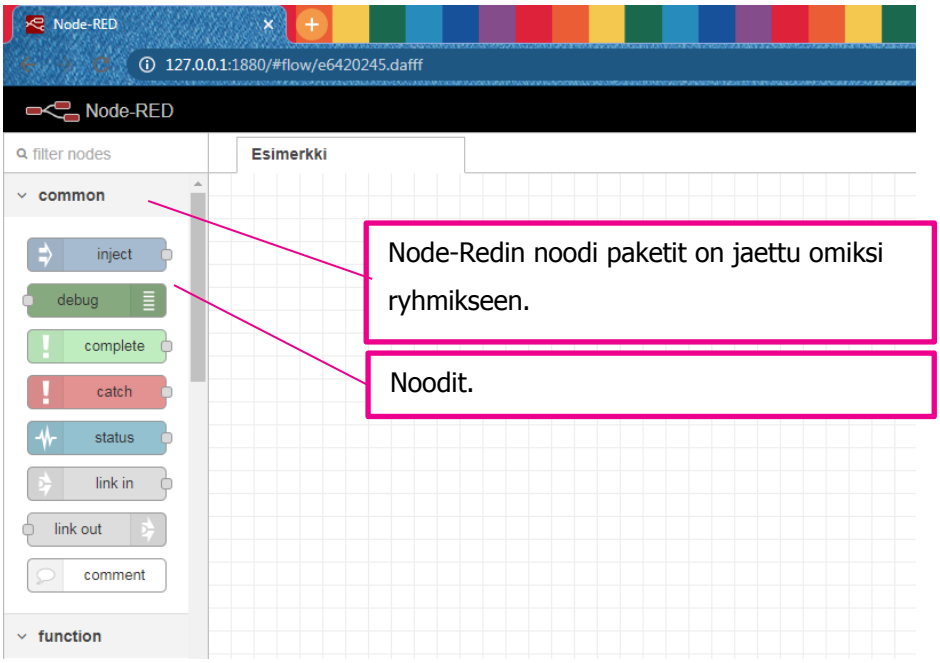
Node-RED on rakennettu käyttämään Node.js pohjaista suorituspalvelinta, joka mahdollistaa editorin avaamisen nettiselaimeen. Ohjelma asennetaan koneelle komentotulkin avulla. Samoin tausta-ajon käynnistäminen (yhteyden avaaminen) suoritetaan komentotulkin kautta. Editori voidaan avata komentotulkin osoittavan osoitteen kautta selaimessa. Kuvassa 32 on esitettynä, kuinka tausta-ajo käynnistetään ja minkä polun kautta editori saadaan avattua. Kuvassa 33 editori on avattuna selaimen kautta.



Kuva 31. Node-RED:n käynnistäminen Windowsin komentotulkilla. (Ranta [2], 2020)

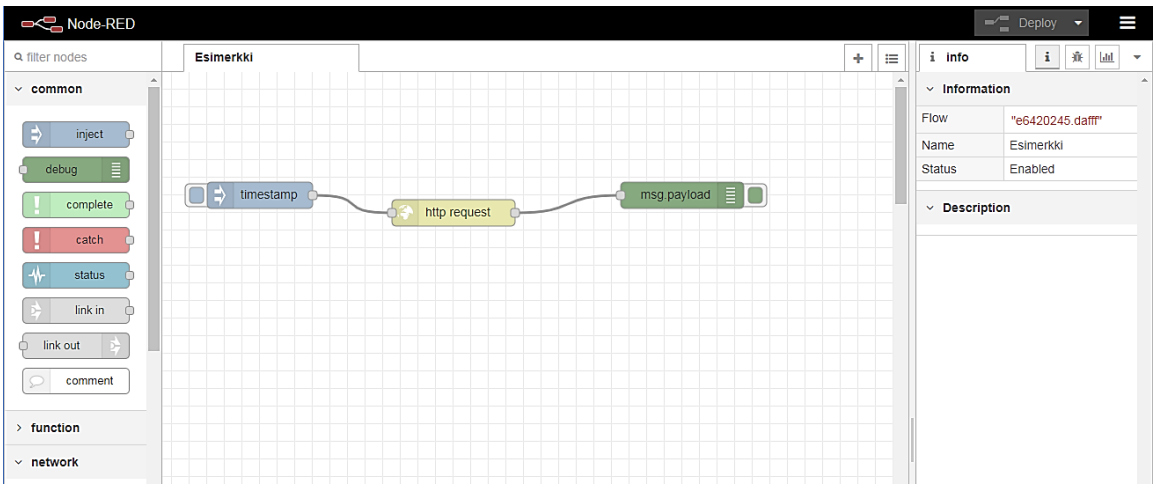
Kuvassa 32 on esitettynä komentotulkilla Node-Red palvelun käynnistäminen:

1. Ohjelmointi alusta käynnistäminen.
2. Käynnistys ja versio tiedot.
3. Käyttöliittymän osoite (xxx.x.x.x) ja portti (xxxx).

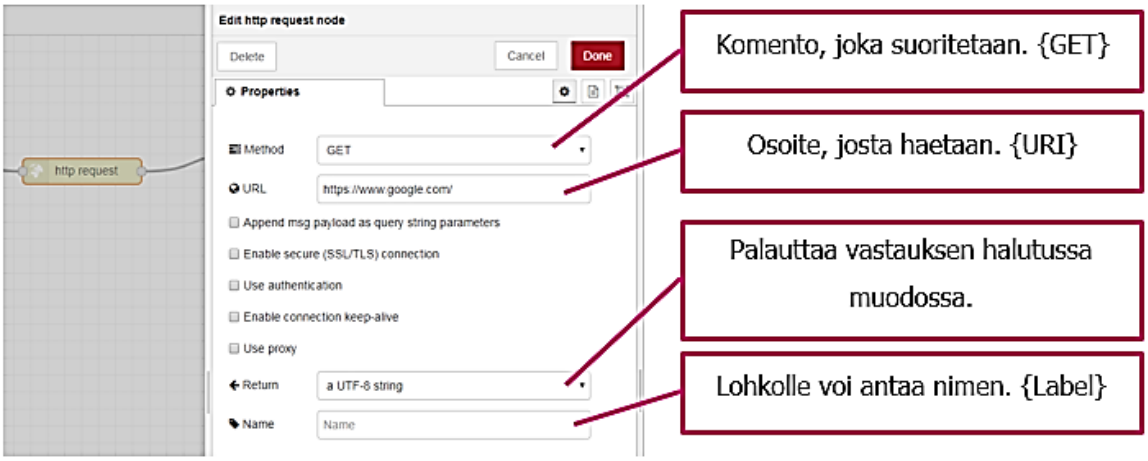


Kuva 32. Node-RED editori avattuna. (Ranta [2], 2020)

Vuokaavio ohjelmointia havainnollistamaan tehtiin yksinkertainen esimerkki, jossa ohjelman suorittaminen tulostaa sivun testaus alustalle. Esimerkki ohjelma hakee Googlen perussivun koodi sisällön testialusta eli debug osioon. Kuvassa 34 on ohjelman rakennettuna.

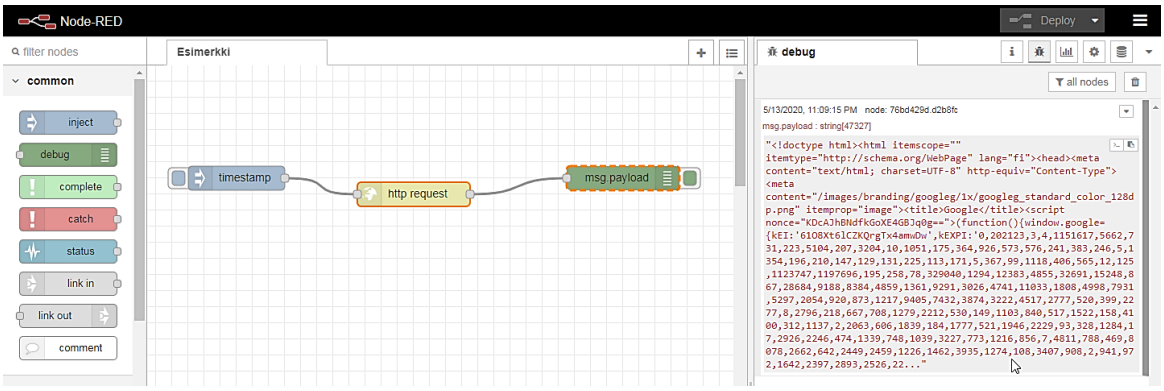


Kuva 33. Ohjelma Googlen sivun tietojen hakemiseen. (Ranta, 2020)



Kuva 34. Http request -solmun asetukset. (Ranta, 2020)

Ohjelma koostuu kolmesta solmusta: *inject, http request* ja *msg payload*. *Inject* toimii triggerinä ja *http request* suorittaa varsinaisen tiedon hakemisen. *Msg payload* taas tulostaa sisällön debug osioon. Kuvassa 35 on haku toiminnon asettelut http request solmulle. Kahdelle muulle solmulle ei tarvitse erillistä asettelua tässä tapauksessa. Kuvassa 36 on tulostettuna Googlen sivutiedot.



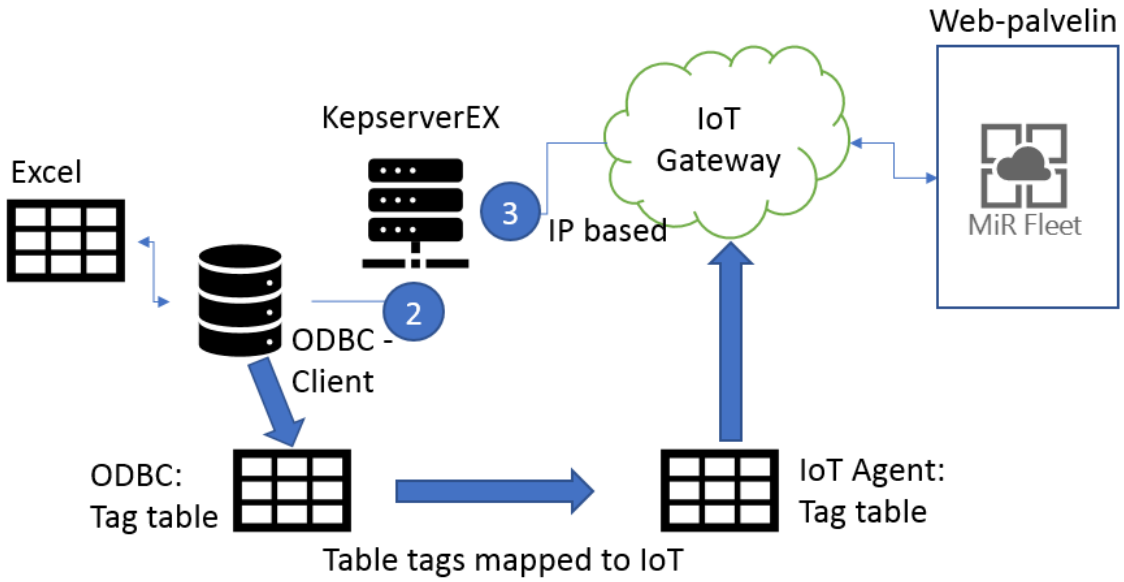
Kuva 35. Ohjelman hakemat sivun tiedot. (Ranta, 2020)

5.4 REST-rajapinta esimerkki

REST-palvelua ja Node-Redin hyödyntämistä kuvaamaan rakensin KepserveEx ympäristöön yksinkertaisen IoT Gateway asiakasrajapinnan ulospäin ohjattavaa tietoa varten sekä ODBC-asiakkaan, joka lukee Testi Exceliä. Exceliin määrittelin sarakkeet ”Name” tagien nimeämistä varten sekä ”Value” tagien arvoja varten. KepserverEx palveluun asiakasrajapinnan osoitteeksi (URL) asetettiin Node-Redin polku ja siellä määritetty http-noodin osoite.

Esimerkin on tarkoitus ilmentää REST-protokollan toimintaa ja tiedon siirtymistä erilaisten alustojen ja palveluiden välillä. Jokaisella palvelulla ja alustalla on omat esitysmuotonsa tiedolle, siksi on hyvä ymmärtää myös palveluiden esitysmuotojen eroja. Eroja on myös eri ohjelmistojen toiminnalla, jolloin yksinkertaisenkin toiminnan rakentaminen vaatii ymmärrystä ohjelmiston käyttämisestä.

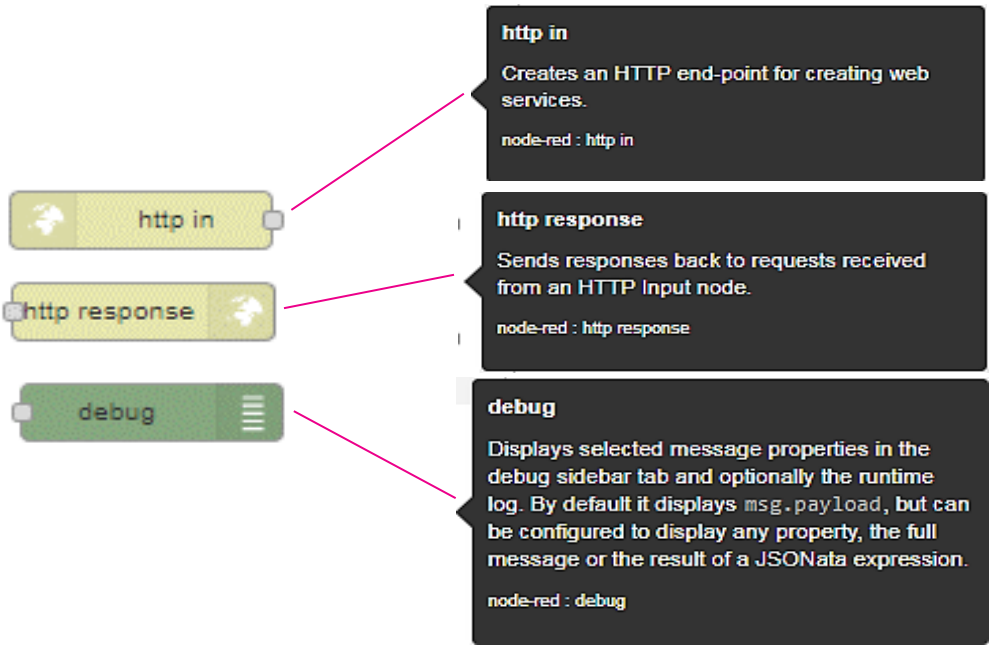
Seuraavissa kappaleissa on kuvattu palvelun osat, joita voidaan verrata kuvassa 37 esitettyyn rakennemalliin. Web-palvelin kuvaa MiR Fleet palvelua, Excel ja ODBC-asiakas kuvaavat PLC:ltä tuotannonohjausjärjestelmään liittymistä. KepserverEx on tässä tapauksessa tuotannon ohjauksen palvelin tai yhdyskäytävä.



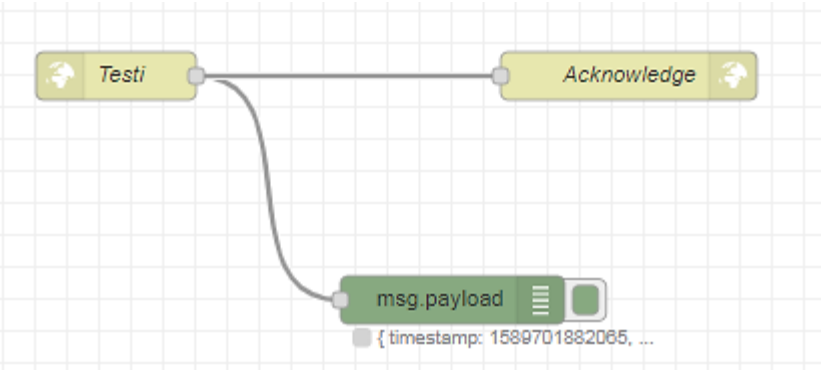
Kuva 36. REST-rajapinta esimerkin rakenne. (Ranta [2], 2020)

5.4.1 Web-palvelin

Node-Rediin luodaan kolmella noodilla web-palvelu, joka lukee KepserverEx IoT Gatewayn asiakas agentin yhteysväylää. Työkulku koostuu kolmesta noodista, jotka on esitetty kuvassa 38 Node-Red palvelun selvennykset noodeille: *http in* luo web-päätteen, jonne yhteys KepserverEx:stä osoitetaan, *http response* lähettää kuittauksen vastaanotetusta tiedosta ja *debug* tulostaa työkulun testaus näkymään. Kaikki halutut noodit raahataan vasemmasta valikosta ruudukolle ja noodien päissä olevista pisteistä voidaan yhdistää noodit toisiinsa klikkaamalla ja viemällä yhdistettävän noodin vastaavalle pisteelle. Kuvassa 39 on esitettynä tämän esimerkin työnkulku.

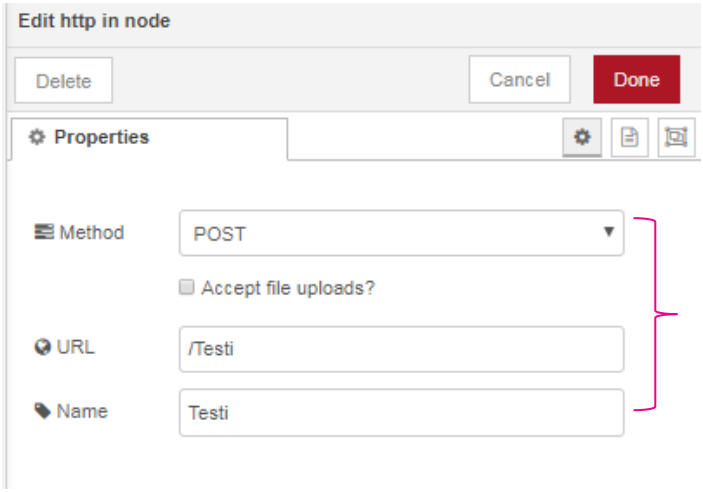


Kuva 37. Web-palvelimella käytettävät noodit. (Ranta [2], 2020)



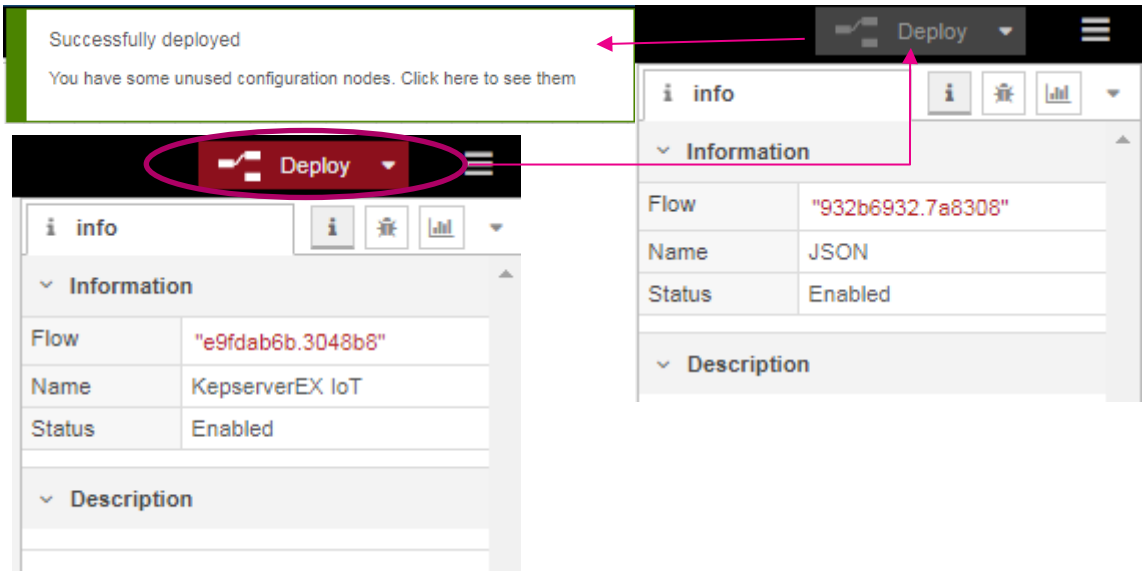
Kuva 38. Node-Rediin tehty web-palvelinpääte. (Ranta [2], 2020)

Web-palvelu noodille annetaan metodiksi ”POST” ja osoitteeksi (URL) ”/Testi”. Osoitus tähän noodiin tapahtuu käytännössä Node-Red palvelun paikallisosoitteella, jonka perään noodin osoite lisätään: http://127.0.0.1:1880/Testi. Noodille voidaan antaa myös nimi, joka näkyy työnkulussa noodin tunnuksena. Kun kaikki on aseteltu, vahvistetaan asetukset ”Done” painikkeella. Asetukset noodille esitettynä kuvassa 40.



Kuva 39. Web-palvelu noodin asettelut. (Ranta, 2020)

Kun noodit on aseteltu ja yhdistetty, voidaan työkulku ottaa käyttöön *Deploy* -painikkeella. Käyttöönoton myötä *Deploy* -painike muuttuu harmaaksi ja palautuu käytettäväksi aina kun työkulkuihin tehdään muutoksia. Tehdyt muutokset astuvat voimaan vasta kun painiketta on painettu, jonka jälkeen ilmoitus banneri kertoo käyttöönoton onnistumisesta tai epäonnistumisesta. Node-Red ohjaa käyttäjää pitkin matkaa ja ilmoittaa selkeästi virheistä työnkuluissa. Työkulun käyttöönotto on esitetty kuvassa 41.

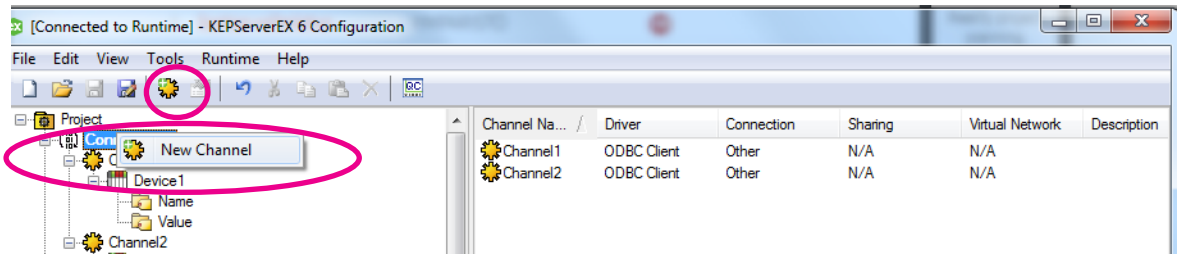


Kuva 40. Työkulun käyttöön ottaminen. (Ranta, 2020)

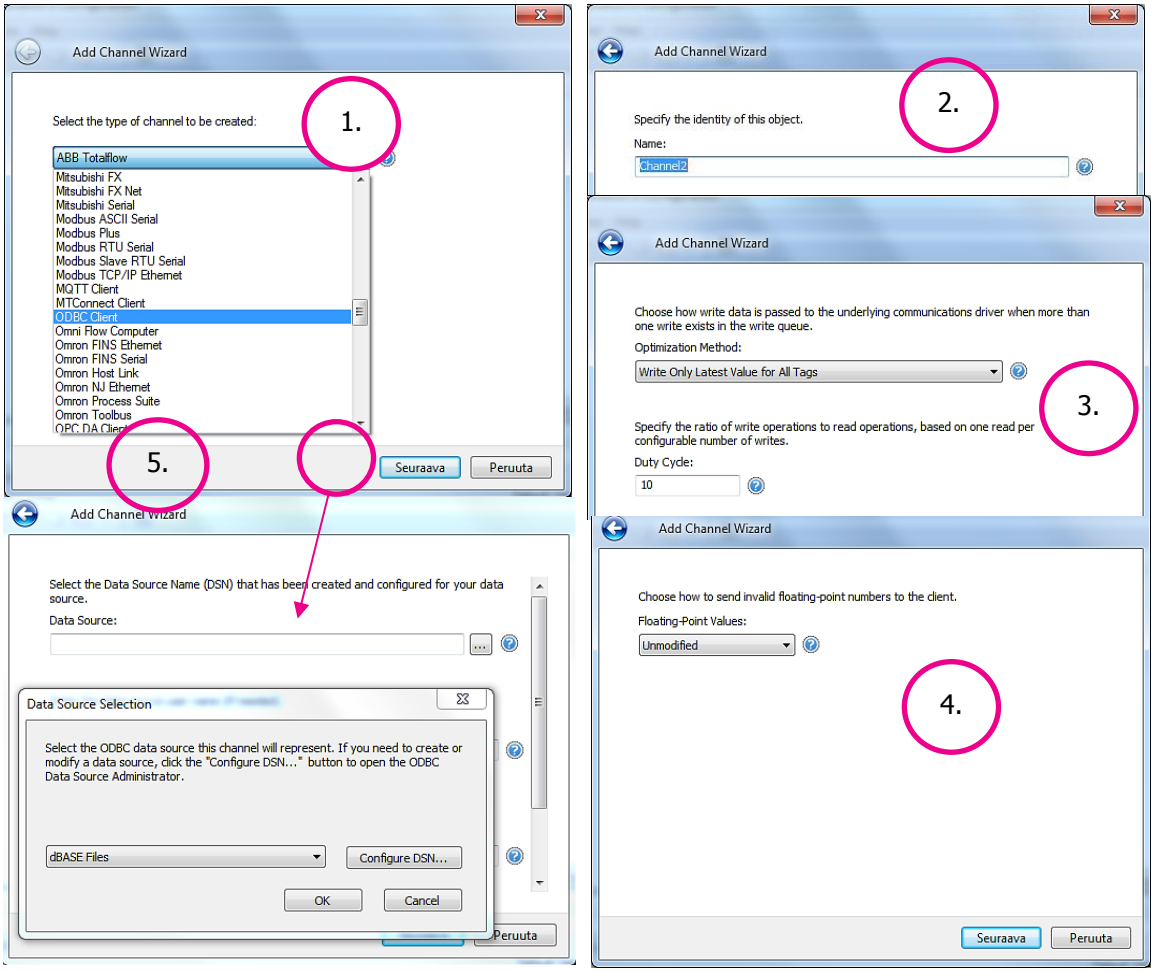
5.4.2 ODBC kanavan määrittely ja Excel työkirjan yhdistäminen

KepserverEx:n käyttöliittymä ohjaa tietoyhteyskanavan luomisessa ja jokaisen valinnan kohdalla on suorat linkit ohjekirjaan, jossa kerrotaan kunkin asetuksen vaikutuksesta luotavaan kanavaa. Kanava luodaan projektipuun *New Channel* kohdasta tai työkalupalkin vastaavasta kuvakkeesta, jotka on esitetty kuvassa 42. Kuvassa 43 on esitettynä vaiheita 1-5, joiden läpi apuohjelma ohjaa asettelua.

Oleellista on kohdassa viisi (5) valita *"Configure DSN",* kun halutaan luoda uusi tietolähde tai muokata olemassa olevaa.



Kuva 41. Uuden kanavan luominen tietoyhteydelle. (Ranta [2], 2020)



Kuva 42. ODBC kanavan määrittely. (Ranta [2], 2020)

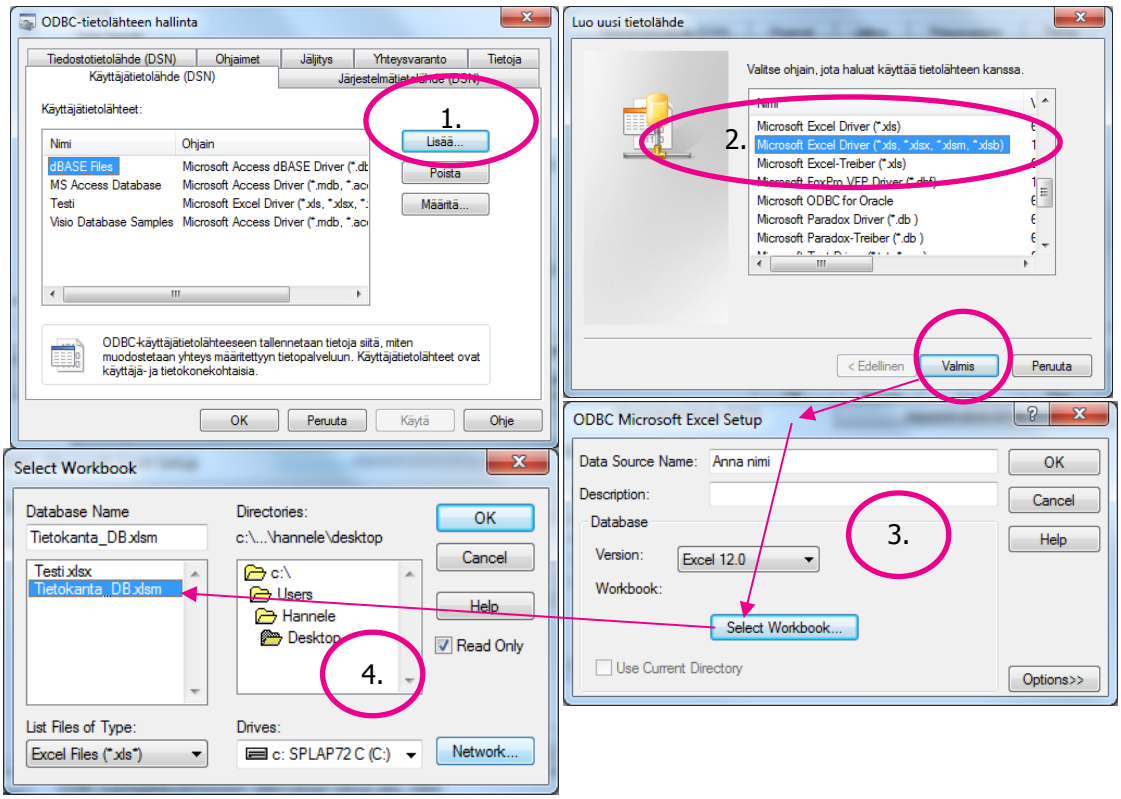
Uusi tietolähde lisätään kuvan 44 mukaisesti:

1. Lisätään uusi tietolähde *Lisää*.

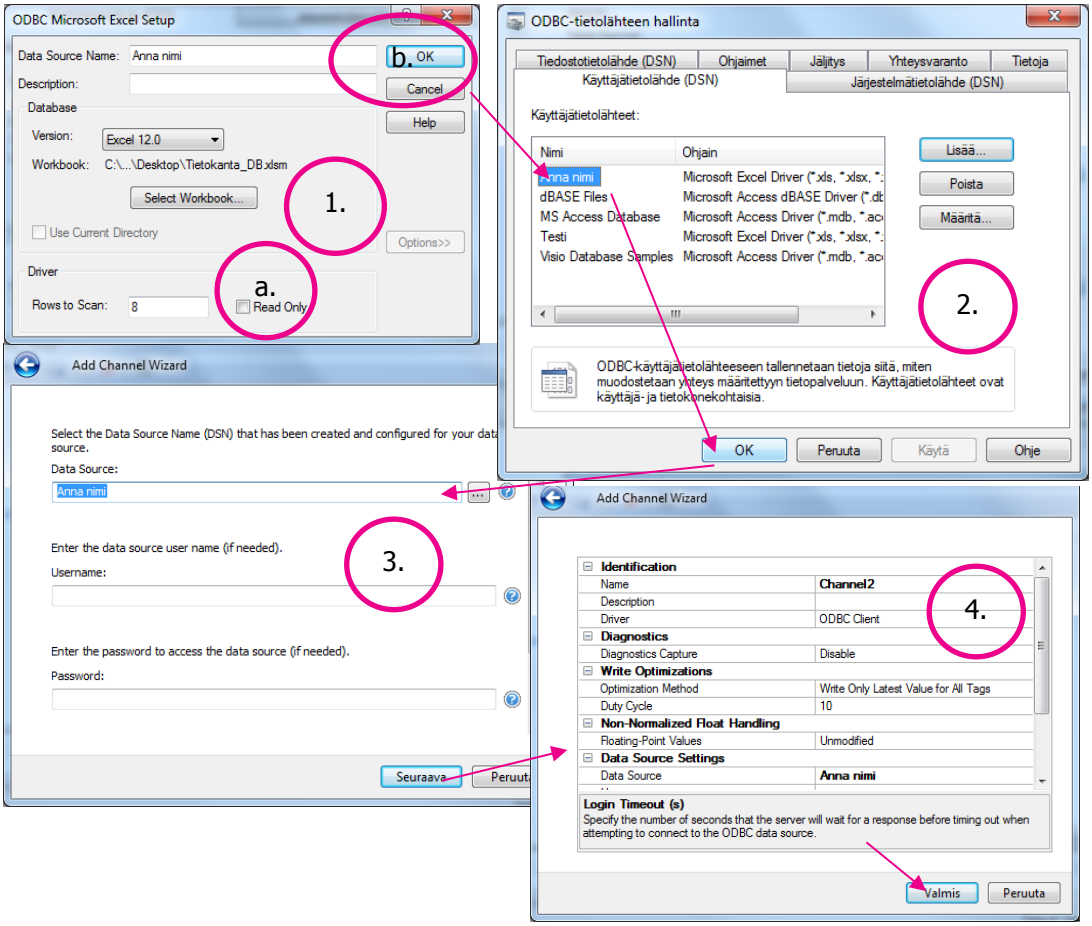
2. Valitaan halutunlainen tiedostotyypin muoto tietolähteelle ja klikataan *Valmis*.

3. Annetaan uudelle tietolähteelle nimi kohdassa *Data Source Name* ja siirrytään valitsemaan haluttu työkirja.

4. Tässä valitaan normaalisti tiedostoselaimesta valmiiksi muokkailtu työkirja. Lopuksi vahvistetaan valinta *OK* -painikkeella.



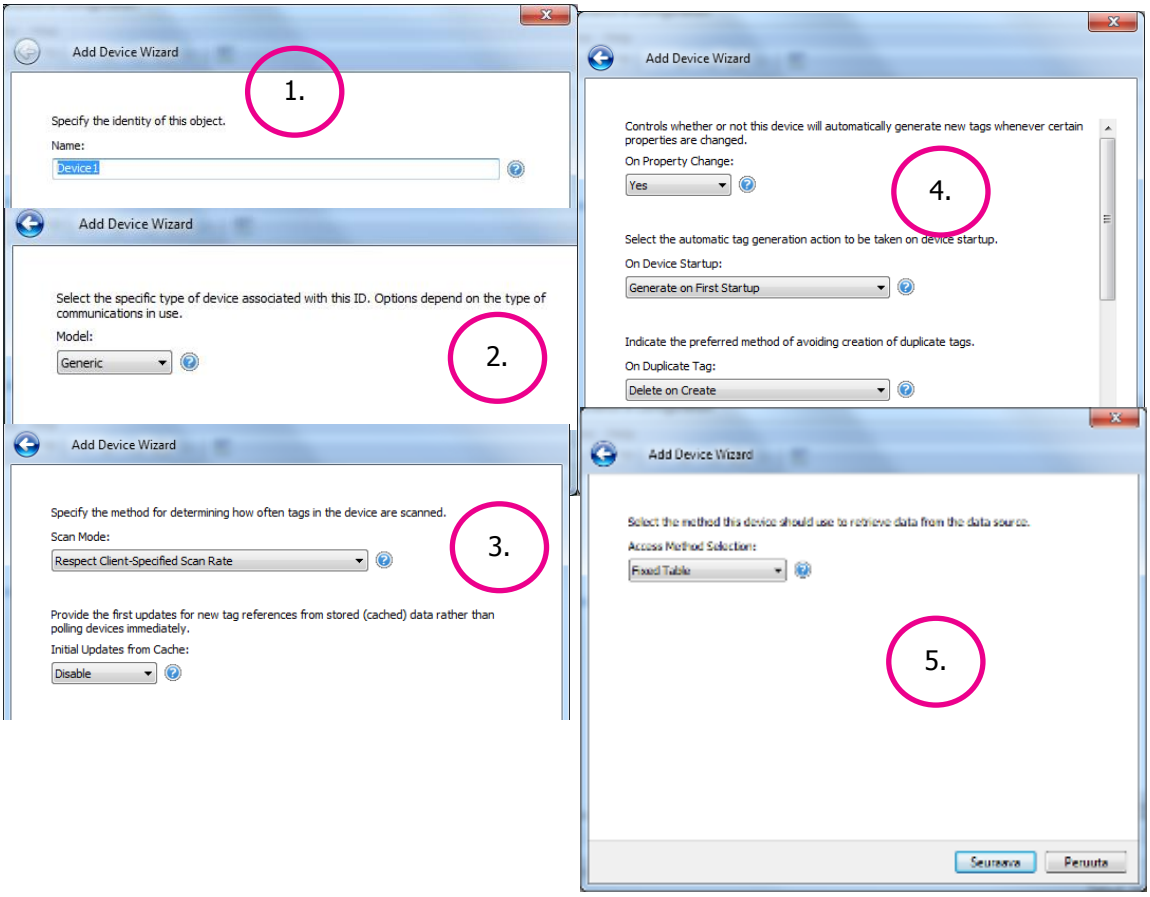
Kuva 43. Excel -työkirjan määrittäminen yhteydelle. (Ranta [2], 2020)



Kuva 44. Excel työkirjan lisääminen ja kanavan viimeistely. (Ranta [2], 2020)

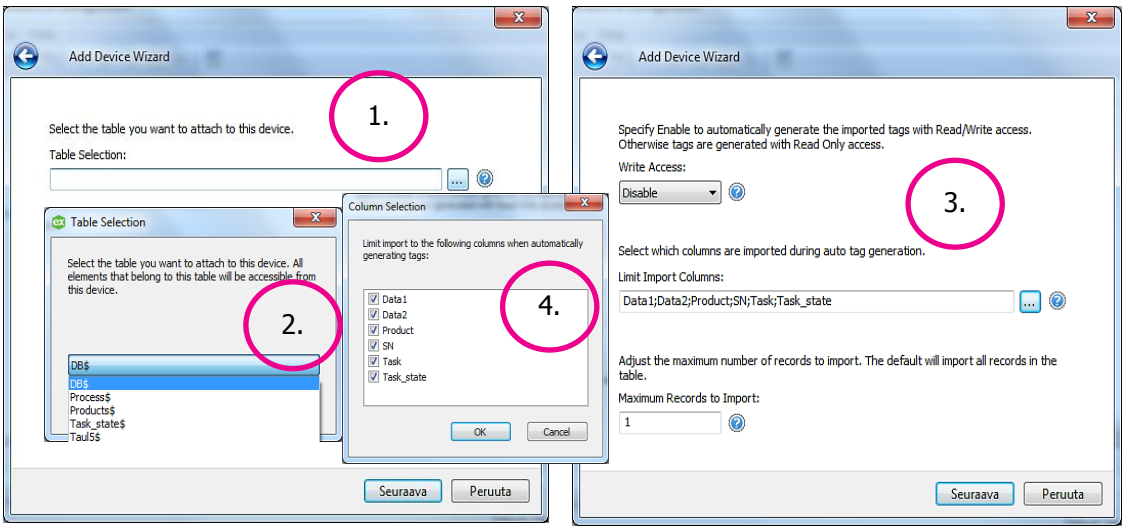
Työkirjan käyttötavat täytyy vielä määrittää ennen tietolähteen ja kanavan viimeistelyä. Kuvassa 45 oleellisinta on määrittää kohdan yksi (1) mukaisesti, onko tietolähde vain luettavissa vai myös muokattavissa KepserverEx:n kautta. Esimerkissä määritettiin tietolähde vapaaksi myös kirjoittamiselle. Uusi tietolähde on määrittelyjen jälkeen kuvan 45 kohdassa kaksi (2) valittavissa kanavan määrityksiin ja lopulta kohdassa neljä (4) tietoyhteyskanava on luotu.

Kanavalle luodaan uusi laiteyhteys, joka lukee määritetystä yhteydestä haluttua taulukkoa. Perusasetukset voi mennä perusasetuksilla kuvan 46 kohtien 1-5 mukaisesti.

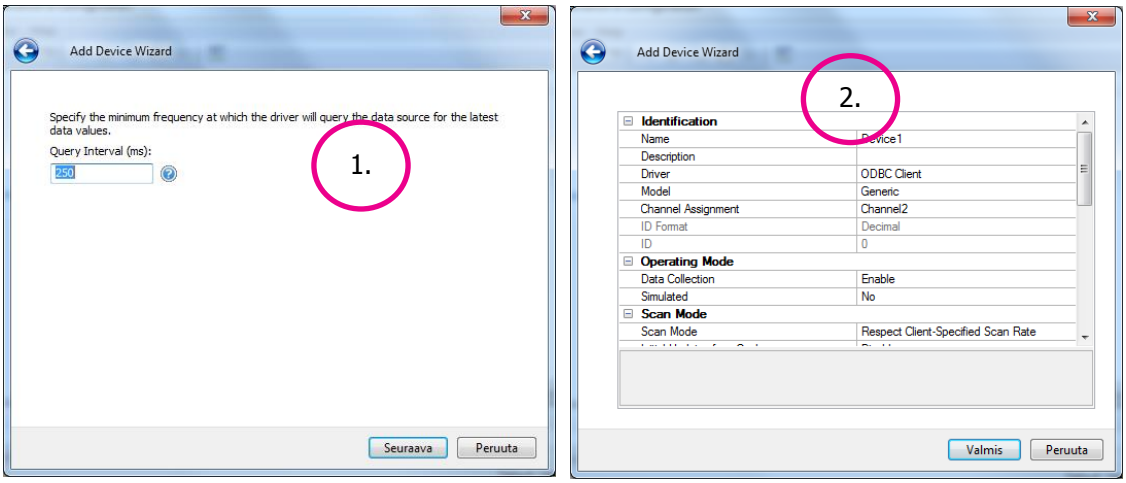


Kuva 45. Laiteasetukset kanavalle. (Ranta [2], 2020)

Kuvassa 47 näytetään kuinka Excel -tietoyhteyden taulukosta haluttu taulukko sivu määritetään laitteelle. Taulukko valitaan alasvetovalikosta kuvan 47 kohtien 1-2 mukaisesti. Kohdissa 3-4 määritetään mitkä saraketiedot otetaan käyttöön tälle laitteelle. Lopuksi määritetään taulukolle päivitystaajuus ja tarkastetaan kaikki tiedot kuvan 48 mukaisessa järjestyksessä.

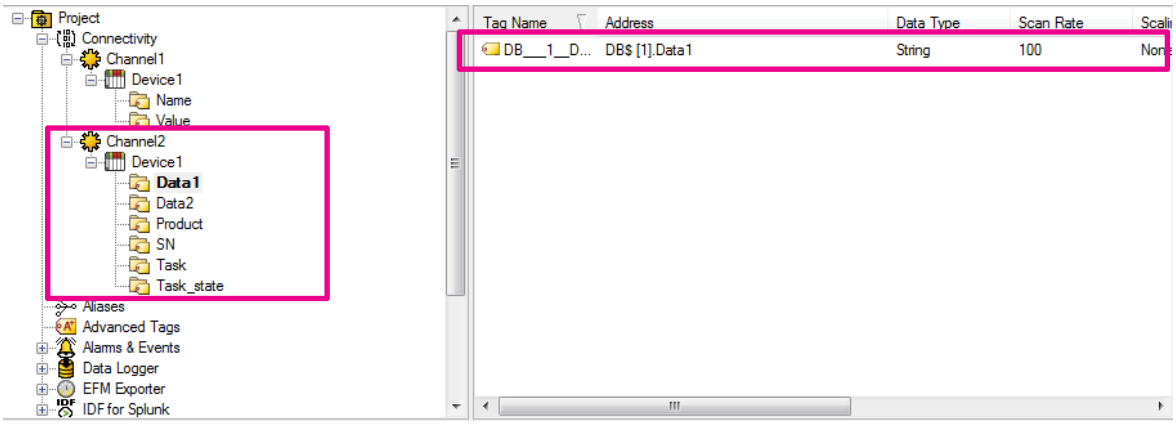


Kuva 46. Taulukon ja sarakkeiden valitseminen laitteelle. (Ranta [2], 2020)



Kuva 47. Päivitystaajuuden asettelu ja asetusten tarkastus. (Ranta [2], 2020)

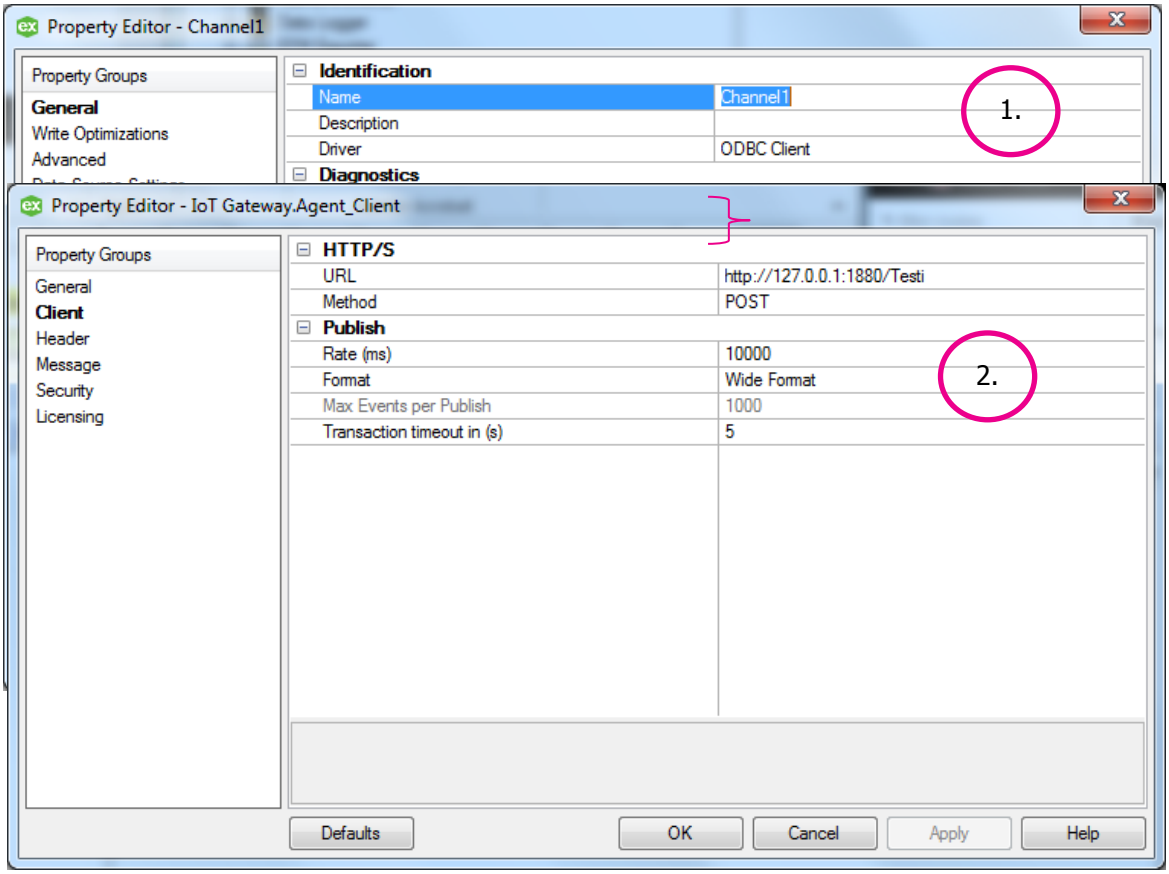
Kun yhteys ja laite asetukset on tehty, luo ohjelmisto projektipuuhun laitteen alle kansiot jokaisesta sarakkeesta ja kansioissa on esitettynä jokaisen sarakkeen kaikki määritellyt rivit esitettyinä tageina. Kuvassa 49 on esitettynä äsken luodun yhteyden ja laitteen alle luotu puu ja yhden kansion tagi.



Kuva 48. Käyttövalmis tietoyhteys ja laite tageineen. (Ranta [2], 2020)

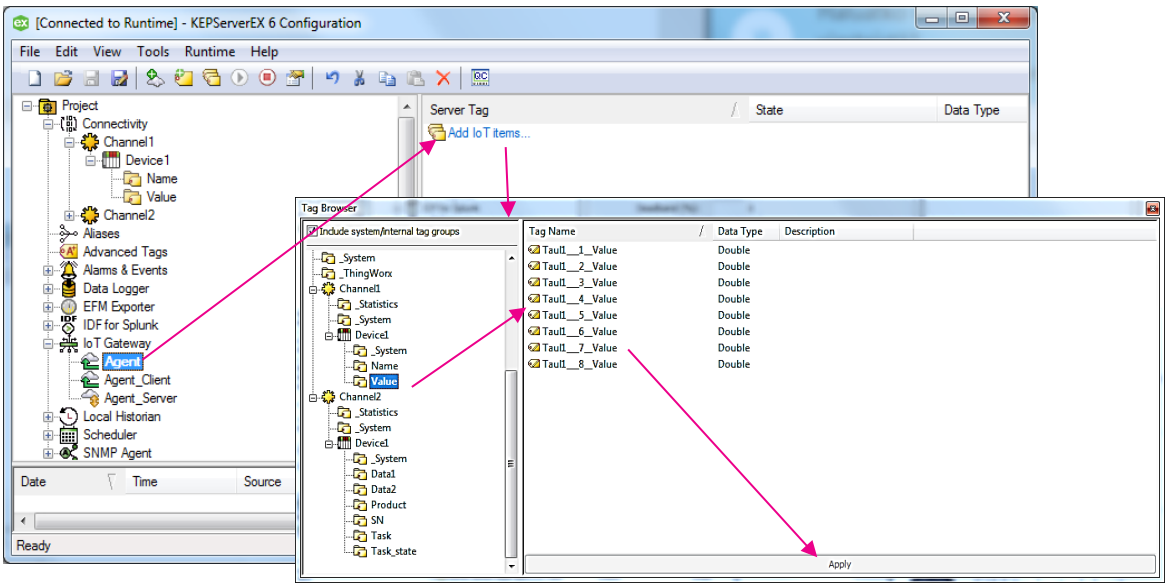
5.4.3 IoT -asiakas yhdyskäytävän määrittely ja tagi yhteydet

Kappaleessa 3.4.1 esitetyn mukaisesti luodaan IoT tietoyhteyskäytävä asiakas tyyppisenä. Kuvasta 50 on tarkasteltavissa asetukset, joita yhteydelle on annettu. Kuvassa kohdassa kaksi on esitettynä osoite, jonka mukaan asiakas agentti osaa lähettää (POST) tietoa oikealla web-palvelimelle.



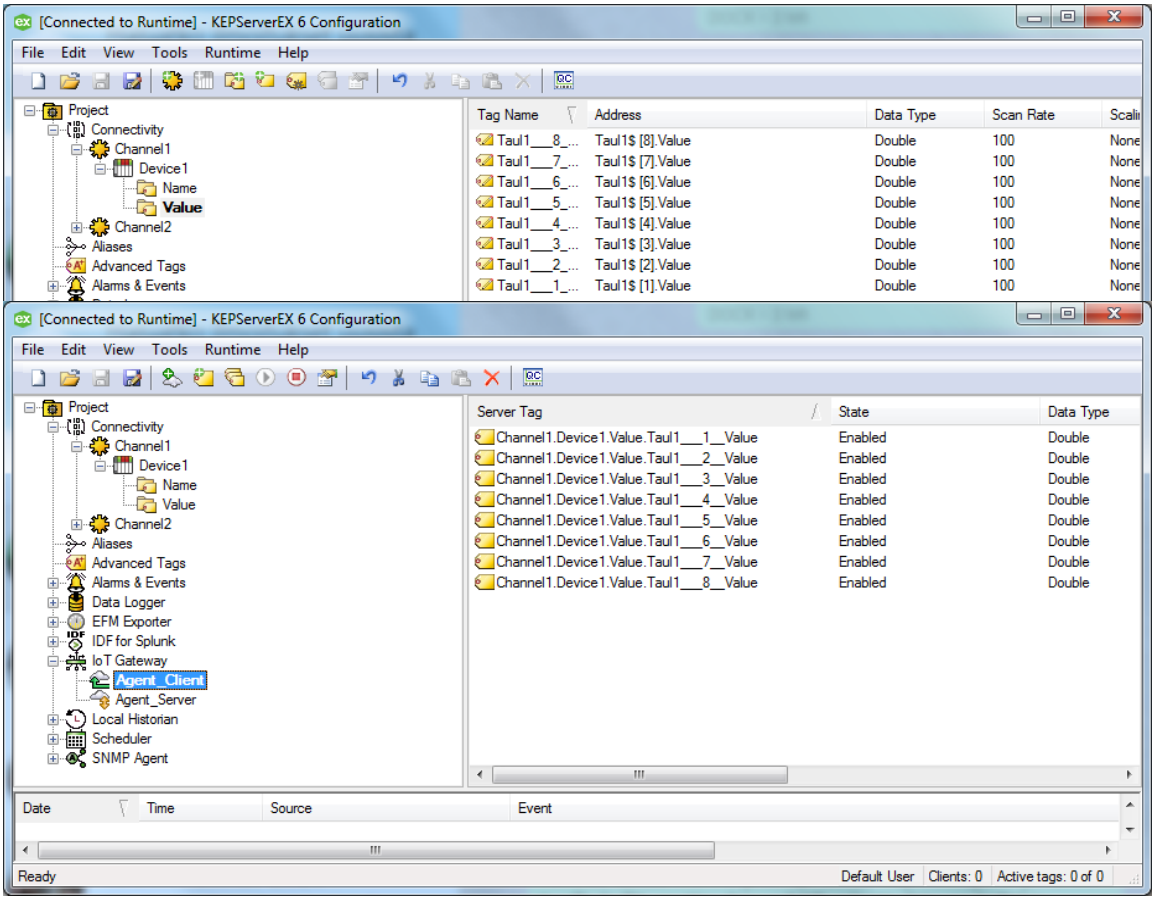
Kuva 49. Asiakasrajapinnan julkaisutavan osoittaminen. (Ranta [2], 2020)

Kuvassa 51 on esitettynä, kuinka asiakas agentille haetaan liityntä rajapinta toisesta yhdyskäytävästä tulevaan tietoon. Agentille lisätään parit *"Add IoT items"*kohdasta, joka aukaisee näkymän saatavilla oleviin laitteisiin ja tietoihin, jotka voidaan linkittää. Valitaan haluttu laite ja yhdistettävät tiedot, lopuksi vahvistetaan valitut *"Apply"* -painikkeella avoinna olevasta ikkunasta.



Kuva 50. Tagien yhdistäminen ja reitittäminen. (Ranta [2], 2020)

Lopuksi voidaan vertailla tagien esitystapaa, riippuen katsotaanko tietoja laitteen listauksesta vai IoT agentin listauksesta. Kuvassa 52 on esitettynä vertailu näkymä tageista.

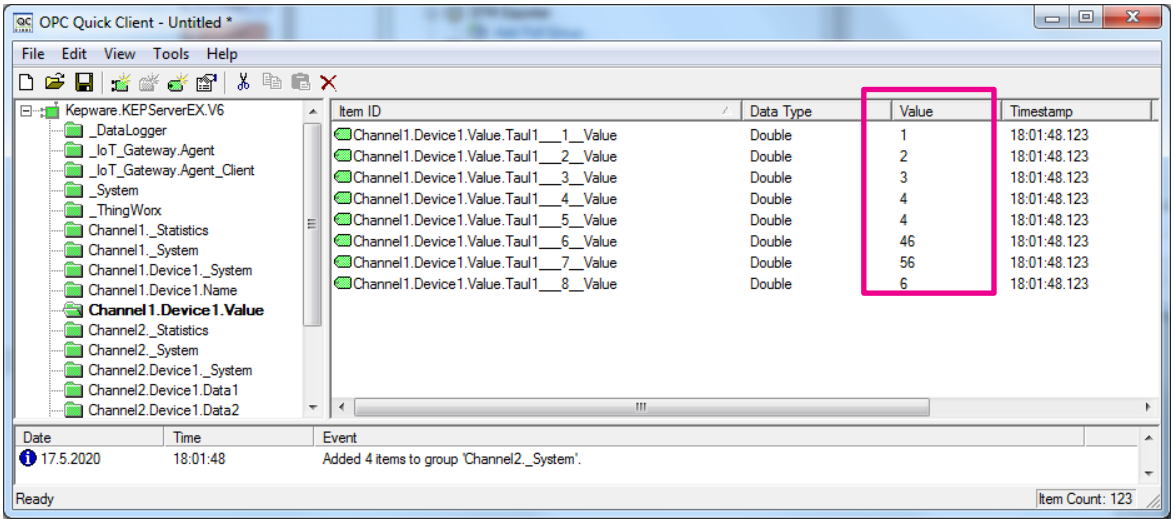


Kuva 51. IoT yhdyskäytävä ja sille määritetyt tagit. (Ranta [2], 2020)

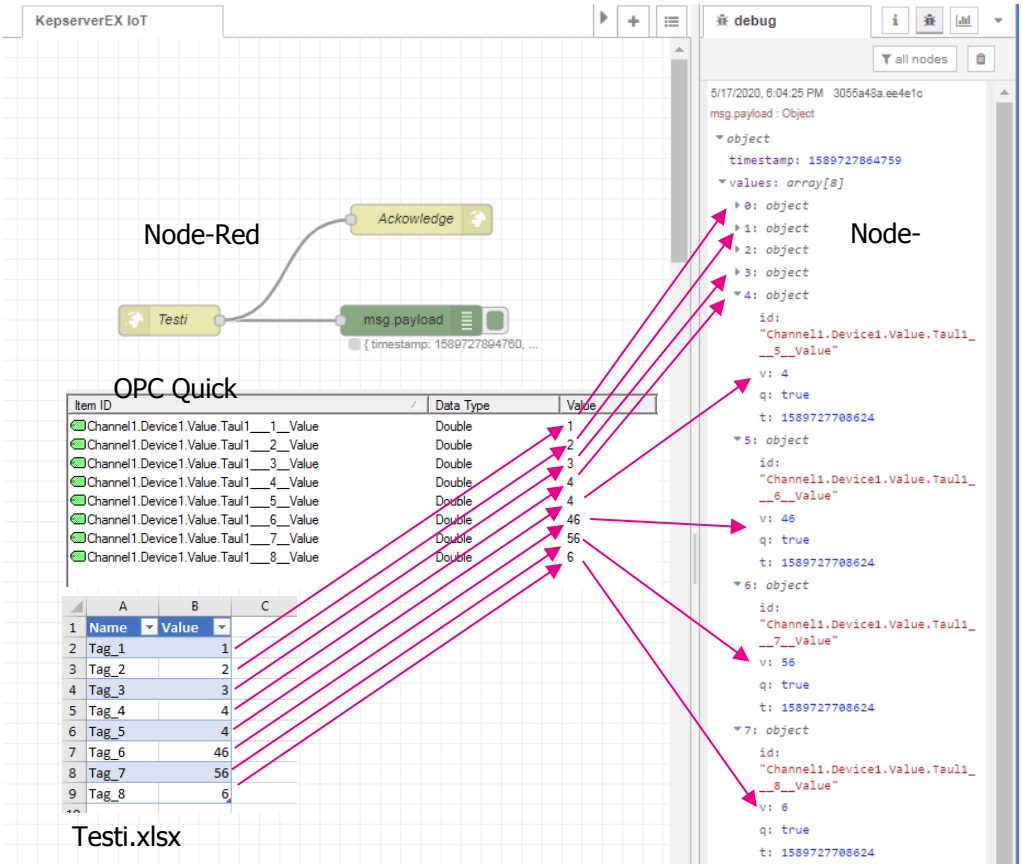
5.4.4 Toiminnan testaaminen

Yhteyksien osalta on nyt rakennettu KepserverEx palvelua hyödyntäen liityntä Excelin tietoihin ja REST palvelua käyttäen yhteys Web-palvelinpäätteelle, joka rakennettiin Node-Rediä hyödyntäen.

ODBC avulla muodostetulla yhteydellä voidaan lukea Excelin solutietoja OPC Quick Client ohjelmalla, joka kuuluu KepserverEx:n työkaluihin. Tällä todennetaan, että Excelin ja Kepserverin välille on saatu muodostettua yhteys. Exceli kuvaa tässä tapauksessa laitteistoa, joka lähettää tilatietoja. ODBC -lukee *Value* -sarakkeen arvot Quick clientin avulla. Koska ODBC-yhteyden data on linkitetty IoT-yhteyden kanssa, siirtyvät sarakkeen arvot sellaisenaan IoT-väylän tagien kautta Node-Rediin rakennetulle web-palvelimelle, REST-protokollaa hyödyntäen.



Kuva 52. OPC Quick Clientilla tarkasteltuna haetut taulukko arvot. (Ranta [2], 2020)



Kuva 53. Vertailuna Excel, Node-Red ja OPC Quick Clientin tiedot, jotka vastaavat toisiaan. (Ranta [2], 2020)

Kuvassa 54 rakennetun tietoyhteyden kautta saadut tiedot web-palvelimelle alkuperäisestä Excelistä. Tagien 5-8 osalta avatuista viesteistä voidaan todeta, että tiedot on luettu oikein aina web-palvelimelle asti. Objektien numerointi alkaa nollasta, joten objektit web-palvelimella ovat 0-7, kun taas Excelissä ja Kepserverillä nimeäminen on 1-8 välillä.

Esimerkin tarkoitus on olla havainnollistava esimerkki, kuinka työhön valikoituneita alustoja voi käyttää. Perusperiaatteiltaan REST-protokolla toimii esimerkissä kuvatuilla tavoilla, joskin kokonaisarkkitehtuurin käyttöönotto vaatii kaikkien eri yhteyksien ja viestien rakentamista tuotannonohjauksen vaatimalle tasolle. Tällöin kuvaan tulee myös MiR Fleetin tarkempien ohjausparametrien asettelu, sekä koko tuotannonohjauksen ja kuormituksen suunnittelu arkkitehtuurin sisälle.

6 TYÖN TOTEUTUS JA ARVIOINTI

Työn suunnittelu ja aiheen ideointi alkoi keväällä 2019 työnohjaajan Manu Pernun johdolla. Toimeksiantajan puolelta tarjottiin kahta mahdollista aihetta opinnäytetyöksi, joista toinen oli vahvasti dokumentointi painotteinen sekä toteutukseen valikoitunut Mobiilirobottien liittäminen tuotannonohjaukseen. Hankkeistamissopimus allekirjoitettiin toukokuussa 2019.

6.1 Työn suunnittelu

Työnsuunnittelu ja hahmottaminen alkoivat jo ennen aihekuvauksia ja lopullisen aiheen lukitsemista. Aiheen ympärillä käytiin keskusteluja muutamaan eri otteeseen yhdessä toimeksiantajan työnohjaaja Manu Pernun kanssa. Toimeksiantajan tarjoama MiR-mobiilirobotteihin syventävä koulutus toimi perehdytyksenä aiheeseen. Kouluttajana toimi Mobile Industry Robots:n kenttäinsinööri.

Hankkeistamissopimuksen allekirjoittamisen jälkeen tehtiin työtä varten tarkempi kuvaus ja taustoitettiin työn merkitystä. Työsuunnitelman ja siihen liittyviä analyysejä työstettiin kesän 2019 aikana. Työsuunnitelma valmistui elokuussa 2019.

Tavoitteeksi työlle asettui yksinkertaisen ja toistettavan ohjausmallin löytäminen automaatioympäristöön, jotka parantavat tuotannollisen työn kannattavuutta. Resurssien mukaan myös mahdollisesti havainnollistavan käytännönsovelluksen rakentaminen ohjausmallista, katsottiin tukevan työn tarkoitusta. Työsuunnitelman tutkimuskysymyksillä pyrittiin ohjaamaan työn fokusta läpi prosessin.

6.2 Työn toteutus ja aikataulu

Alkuperäinen aikataulu oli toteuttaa työ 2019 kesän ja syksyn kuluessa. Aikataulu kuitenkin venyi aina keväälle 2020 asti, resurssien ollessa rajalliset. Kesän ja syksyn 2019 aikana kerättiin pohjatietoa aiheesta. Varsinaiseen mallinrakentaminen ja käytännönsovellusten testaaminen alkoivat vasta talven ja kevään 2020 aikana. Työ vaati lopulta paljon vaivaa ja aikaa, jotta se voitiin esittää tässä muodossaan.

Työn aikana tutustuttiin ainakin tusinaan erilaisia ohjelmistoja, joista työssä on esitelty käytetyt ohjelmistoratkaisut. Ratkaisun löytäminen lähti avautumaan ympäristön ISA95-mallintamisen jälkeen, jonka myötä ratkaisuja oli helpompi hahmottaa työtä rakentaessa. Erilaisten työvaiheiden ja kokeilujen jälkeen työssä löydettiin esittämiskelpoinen malliratkaisu, jonka visuaalinen kuvaus on tarkentunut aivan viime metreille asti.

Kaikkien työn johdannossa ja teoriatautoissa esitettyjen näkökulmien yhdistäminen oli lopulta loogista ja ne syntyivät teknistenkuvausten pohjalta. Prosessin myötä kaikki asiat löysivät oman järjestyksensä opinnäytetyön esitetyssä muodossa.

6.3 Lopputuloksen arviointi

Tavoitteena oli löytää keskitetyn tuotannonohjauksen tarpeisiin sopiva ratkaisu liittää mobiilirobotit osaksi sisälogistista järjestelmää ja nostaa siten tuottavuutta. Käytetyt ratkaisumallit ovat kaikki olemassa olevia eikä niiden hyödyntämisen esteenä ole juurikaan taloudellisia tai järkiperusteita. Työssä esitetyt ja koostetut mallit antavat selkeän kuvauksen järjestelmille ja niiden sijoittumiselle topologisesti suhteessa toisiinsa. Kaikki kuvaukset on rakennettu olemassa olevien mallien pohjalta.

Työ on toisaalta laaja ja toisaalta melko rajattu kokonaisuus. Tällaisen työn rajaaminen selkeäksi kokonaisuudeksi on haastavaa, sillä itsessään tuotannonohjaus, tiedonsiirtorajapinnat sekä lean ovat isoja kokonaisuuksia itsenäisinäkin aiheina. Työssä kuvatut tiedonsiirron ominaisuudet ovat odotusten mukaisesti pilkottu yksinkertaisiksi kokonaisuuksiksi ja niitä voi yhdistellä monin eritavoin. Esimerkki tietoyhteys antaa käytännöntasolla osviittaa millaisia asioita tietoyhteyksien rakentamiseen liittyy valituilla alustoilla, edelleen sen ollessa vain yksi mahdollinen ratkaisu.

Prosessin läpivieminen ja resurssien riittävyys olivat haasteellisia, mutta antoisia. Lopputulos kuvastaa silti automaation tiedonsiirtoratkaisujen arkkitehtuuria ja antaa vastauksen työn otsikoinnille: ”Mobiilirobottien liittäminen tuotannonohjaukseen”. Tuotannonohjauksen ja REST-protokollan käsittelyssä on huomioitu erilaisia tasoja lean-filosofian ja tiedonsiirtoarkkitehtuurin näkökulmista kuin myös niiden välisten riippuvuuksien ja yhtäläisyyksiä huomioiden.

Työstä tuli eheä kokonaisuus, jonka pohjalta riittää aiheita ja tutkimista myös seuraaviin opinnäytetöihin.

LÄHTEET JA TUOTETUT AINEISTOT

ALA-MUTKA Kirsti, RINTALA Matti, SAVIKKO Vespe, PALVIAINEN Jarmo 1996-2002 [Viitattu 2020-03-25.] Saatavilla: http://www.cs.tut.fi/etaopetus/titepk/luku19/OSI.html

ASCOLAB. [1] OPC UA Data Model [Viitattu 2020-04-06.] Saatavilla: http://www.asco-lab.com/en/technology-unified-architecture/meta-model.html

ASCOLAB. [2] OPC UA Data Model [Viitattu 2020-04-06.] Saatavilla: http://www.asco-lab.com/en/technology-opc-classic.html

BRUSH, Kate. 8.2019. IoT Agenda TechTarget. DEFINITION mobile robot (mobile robotics) [Viitattu 2020-03-14.] Saatavilla: https://internetofthingsagenda.techtarget.com/definition/mobile-robot-mobile-robotics

CERIFFI Oy Kahdeksan hukan muotoa. [Viitattu 2020-02-19.] Saatavilla: http://www.ceriffi.fi/palvelut/kahdeksan-hukan-muotoa

DUMMIES, 2020. Network Basics: Clients and Servers. [Viitattu 2020-05-10.] Saatavilla: https://www.dummies.com/programming/networking/network-basics-clients-and-servers/

FIELDING, Roy 2000. Principled Design of the ModernWeb Architecture. [Viitattu 2020-05-11.] Saatavilla: https://www.ics.uci.edu/~fielding/pubs/webarch_icse2000.pdf

FORCEPOINT. CYBER EDU: What is the OSI Model? [Viitattu 2020-03-25.] Saatavilla: https://www.forcepoint.com/cyber-edu/osi-model

HEIKKILÄ, Kari. 4.2.2019. Ymmärrä ekosysteemi hankittavan ohjelmiston ympärillä [Viitattu 2020-03-15.] Saatavilla: https://www.visma.fi/blog/ymmarra-ekosysteemi-hankittavan-ohjelmiston-ym-parilla/

JYVÄSKYLÄN YLIOPISTO. Mitä prosessit ovat? [Viitattu 2020-02-19.] Saatavissa: https://www.jyu.fi/laatua/ohjaus/prosessien-mallintaminen/mitaprosessitovat

KEPWARE, 10.5.2017. IIoT Use Cases and Applications. [Viitattu 2020-05-12.] Saatavilla: https://fast.wistia.net/embed/iframe/xuabl4m0oh?popover=true

KEPWARE [1], PTC Inc 2019. KEPServerEX. [Viitattu 2020-05-12.] Saatavilla: https://www.kepware.com/getattachment/5759d980-7641-42e8-b4fb-7293c835a2f9/kepserverex-manual.pdf

KEPWARE [2], PTC Inc 2019. IoT Gateway. [Viitattu 2020-05-12.] Saatavilla: https://www.kepware.com/getattachment/96bdb7bb-4f9a-4cfe-be30-ef048d16dd83/iot-gateway-manual.pdf

KEPWARE [1], PTC Inc 2020. [Viitattu 2020-05-10.] Saatavilla: https://www.kepware.com/en-us/about/overview/

KEPWARE [2], 2020. Connectivity Guide KEPServerEX®, DDE, and Excel [Viitattu 2020-05-10.] Saatavilla: https://www.kepware.com/getattachment/5ab60e28-3bb9-4573-bcd0-4556f6ccaa33/kepserverex-dde-excel.pdf

KIRAN D. R. 18.6.2019. Elsevier Science & Technology. Production Planning and Control. Elsevier Inc. MPS Limited, Chennai Intia.

LAHOTI, Sugandha 11.7.2019. Defining REST and its various architectural styles. [Viitattu 2020-05-12.] Saatavilla: https://hub.packtpub.com/defining-rest-and-its-various-architectural-styles/

LAHOTI, Sugandha 2.10.2018. What are REST verbs and status codes [Tutorial] [Viitattu 2020-05-12.] Saatavilla: https://hub.packtpub.com/what-are-rest-verbs-and-status-codes-tutorial/

LEWIS, Bryon [1] 12.3.2018. Video 8 - Control Systems Review - Industrial Networking Part 1 of 2 [Viitattu 2020-05-10.] Saatavilla: https://www.youtube.com/watch?v=4c61paFk1YE

LEWIS, Bryon [2] 11.3.2018. Video 8A - Control Systems Review - Industrial Networking Part 2 of 2 [Viitattu 2020-05-10.] Saatavilla: https://www.youtube.com/watch?v=_cqlhXGToqk

MDN WED DOCS. HTTP headers [Viitattu 2020-03-15.] Saatavilla: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers

MIKKONEN, Juuso. 27.5.2017 Tivi, Uutiset, Rajapinnat. Rest on nettipalveluiden yhteinen kieli. [Viitattu 2020-02-20.] Saatavilla: https://www.tivi.fi/uutiset/rest-on-nettipalveluiden-yhteinen-kieli/23703ab5-dd19-383e-a422-ebfc3d910583

MIR, 18.1.2017. Webinar: MiR Fleet 3 - Remote control. [Viitattu 2020-05-11.] Saatavilla: https://www.youtube.com/watch?time_continue=36&v=RrT45QVHFUg&feature=emb_logo

MIR [1]. MiR FLEET REST API 2.8.2.2 [Viitattu 2020-04-13.] Saatavilla: https://www.mobile-industrial-robots.com/media/9036/mir_fleet_rest_api_2822.pdf

MIR [2]. About MiR. [Viitattu 2020-05-12.] Saatavilla: https://www.mobile-industrial-robots.com/en/about-mir/

MIR [3], 2020. MiR Fleet. [Viitattu 2020-05.12.] Saatavilla: https://www.mobile-industrial-robots.com/en/solutions/robots/mir-accessories/mir-fleet/

MODIG Niklas, ÅHLSTRÖM Pär. 2017. Tätä on lean. Ratkaisu tehokkuusparadoksiin. Rheologica Publishing. Bull Graphics AB, Halmstad Ruotsi.

NODE-RED. [Viitattu 2020-05-12.] Saatavilla: https://nodered.org/about/

OPC FOUNDATION. [1] What is OPC? [Viitattu 2020-03-31.] Saatavilla: https://opcfoundation.org/about/what-is-opc/

OPC FOUNDATION. [2] Unified Architecture [Viitattu 2020-03-31.] Saatavilla: https://opcfoundation.org/about/opc-technologies/opc-ua/

OPC FOUNDATION. [3] OPC Unified Architecture. Interoperability for Industrie 4.0 and the Internet of Things [Viitattu 2020-03-31.] Saatavilla: https://opcfoundation.org/wp-content/uploads/2016/05/OPC-UA-Interoperability-For-Industrie4-and-IoT-EN-v5.pdf

QUALITY KNOWHOW KARJALAINEN OY. Six Sigma: Yleistä Leanistä. Mitä Lean on? [Viitattu 2020-03-22.] Saatavilla: http://www.sixsigma.fi/index.php/fi/lean/yleinen/

RADA, Michael 21.1.2018. INDUSTRY 5.0 definition. [Viitattu 2020-03-22.] Saatavilla: https://medium.com/@michael.rada/industry-5-0-definition-6a2f9922dc48

SAMARA, Tarek. 19.10.1015 John Wiley & Sons, Incorporated. ERP and Information Systems: Integration or Disintegration. John Wiley & Sons, Incorporated. Verkkoaineiston ISBN: 9781119232742.

SCHIEKOFER, Rainer, SHOLZ, Andreas, WEYRICH Michael 2018. Siemens AG, University of Stuttgart. REST based OPC UA for the IIoT. [Viitattu 2020-03-31.] Saatavilla: https://www.ias.uni-stuttgart.de/dokumente/publikationen/2018_rest_based_opc_ua_for_the_iiot.pdf

SHAW, Keith 22.10.2018. NetworkWorld: The OSI model explained: How to understand (and remember) the 7 layer network model. [Viitattu 2020-03-24.] Saatavilla: https://www.networkworld.com/article/3239677/the-osi-model-explained-how-to-understand-and-remember-the-7-layer-network-model.html

SIEMENS, 2020. ISA 95 Framework & Layers. [Viitattu 2020-05-10.] Saatavilla: https://www.plm.automation.siemens.com/global/en/our-story/glossary/isa-95-framework-and-layers/53244

SOLVING. Vihivaunut (AGV). [Viitattu 2020-03-15] Saatavilla: https://www.solving.com/tuotteet/vihivaunut-agv/

WIKIPEDIA [1]. TCP/IP [Viitattu 2020-03-30.] Saatavilla: https://fi.wikipedia.org/wiki/TCP/IP

WIKIPEDIA [2]. Transmission Control Protocol [Viitattu 2020-05-10.] Saatavilla: https://en.wikipedia.org/wiki/Transmission_Control_Protocol

KUVAT

KEPWARE [3], PTC Inc 2020. [Viitattu 2020-05-10.] Saatavilla: https://www.kepware.com/en-us/configuration-api-example-code/

MIR. MOBILE INDUSTRIAL ROBOTS. MiR Hook 200. [Viitattu 2020-03-15.] Saatavilla: https://www.mobile-industrial-robots.com/media/3998/mir_200_hook_bur_01_small.png?anchor=center&mode=crop&width=560&height=420&rnd=131798519860000000

TUOTETUT AINEISTOT

RANTA, Hannele 2019. Opinnäytetyön *Työsuunnitelma V1.0 2019-08-11.* [Viitattu 22.3.2020]

RANTA, Hannele [1] 2020. Topologia kuvat ja kuvaukset itse rakennettuja kuvia.

RANTA, Hannele [2] 2020. Kuvakaappaukset ohjelmista ja käyttöliittymistä itse otettuja ja muokattuja.