Tampereen yliopisto

Lauri Kohtamäki

MONORAIL-AUTOMAATIOJÄRJESTELMÄSTÄ KERÄTYN TIEDON HYÖDYNTÄMINEN TEOLLISUUS 4.0:N NÄKÖKULMASTA

Luonnontieteiden ja tekniikan tiedekunta
Diplomityö
Marraskuu 2020

TIIVISTELMÄ

Lauri Kohtamäki: Monorail-automaatiojärjestelmästä kerätyn tiedon hyödyntäminen Teollisuus 4.0:n näkökulmasta
Diplomityö
Tampereen yliopisto
Johtamisen ja tietotekniikan diplomi-insinöörin tutkinto-ohjelma
Marraskuu 2020

Modernit automaatiojärjestelmät tuottavat valtavasti dataa, mutta samalla vaativat käyttäjiltään yhä enemmän osaamista. Teollisuus 4.0:n tuoma teknologian kehitys helpottaa merkittävästi automaatiojärjestelmien tuottaman datan käsittelyä. Tämän diplomityön tarkoituksena on tutkia datan tuomaa lisäarvoa Monorail-automaatiojärjestelmän käyttäjille.

Tässä työssä rakennetaan suunnittelutieteellistä tutkimusmenetelmää hyödyntämällä prototyyppi datan keräämisen, analysoinnin ja esittämisen avulla saatavien hyötyjen tutkimiseksi. Prototyypin rakentamisen pohjana käytetään Teollisuus 4.0:n avainteknologioita ja suunnitteluperiaatteita. Datan kerääminen toteutettiin olemassa olevaa, Common Industrial Protocol –protokollaa käyttävää, järjestelmää hyödyntäen. Kerätty data tallennettiin tiedostoina paikalliselle palvelimelle ja siirrettiin sieltä SQL-kantaan. Datan analysointi toteutettiin Pythonin avoimen lähdekoodin datatiede-kirjastojen ekosysteemiä hyödyntämällä. Datan visualisointi toteutettiin Plotly-visualisointikirjastolla ja Flask-kirjastoon perustuvalla websovelluksella.

Työn tuloksena saatiin aikaan toimiva prototyyppi datan keräämiselle, analysoinnille ja esittämiselle. Prototyypin avulla pystyttiin osoittamaan, miten järjestelmä olisi mahdollista toteuttaa ja millaista lisäarvoa sen avulla olisi mahdollista saada.

Avainsanat: Teollisuus 4.0, tiedonkeruu, analytiikka, automaatio, ohjelmoitava logiikkaohjain.

ABSTRACT

Lauri Kohtamäki: Utilizing information gathered from Monorail automation system from the perspective of Industry 4.0
Master of Science Thesis
Tampere University
Master's Degree programme in Management and Information Technology
November 2020

Modern automation systems produce vast amounts of data, but at the same time require more expertise from their users. Industry 4.0 provides tools that significantly facilitate the processing of data generated by automation systems. The purpose of this master's thesis is to study the benefits of data for Monorail automation systems users.

In this master's thesis, a prototype is built using design science research method to study the benefits of collection, analysis and visualizations of data. The key technologies and design principles of Industry 4.0 are used as the basis for building the prototype. Data collection was carried out based on the existing Common Industrial Protocol communication. The collected data was stored as files on the local server and transferred from there to the SQL database. Data analysis was done using the Python open source data science ecosystem. Data visualization was implemented with the Plotly visualization library and a web application based on the Flask library.

The result of the work was a working prototype for collection, analysis and visualization of data. The prototype was able to show how the system could be implemented and what added value would be possible to obtain.

Keywords: Industry 4.0, data collection, data analytics, automation, programmable logic controller.

ALKUSANAT

Tämä diplomityö toteutettiin Cimcorp Oy:lle. Varsinainen ohjelmisto toteutettiin osana työtehtäviäni ja kirjallinen osuus kirjoitettiin hartaasti työn ohella opiskellessa.

Haluan kiittää Jyrki Anttosta ja Jani Tuomolaa tästä mielenkiintoisesta tehtävästä ja vapaudesta toteuttaa projekti haluamallani tavalla.

Suuret kiitokset Sari Vesiluomalle ja Vesa Latva-Pukkilalle neuvoista ja kommenteista, jotka auttoivat eteenpäin, kun kirjallisen osuuden suunta ei tahtonut löytyä.

Kiitokset myös apulaisprofessori David Hästbackalle ja professori Kari Syställe.

Lopuksi haluan kiittää vaimoani ymmärryksestä ja tuesta koko opintojeni aikana.

Porissa, 12.11.2020

Lauri Kohtamäki

SISÄLLYSLUETTELO

LYHENTEET JA MERKINNÄT

API	engl. Application Programming Interface
CRISP-DM	engl. CRoss-Industry Standard Process for Data Mining
CSS	engl. Cascading Style Sheet
CSV	engl. comma-separated values
GQM	engl. Goal Question Metric, tavoite-kysymys-metriikka
HTML	engl. HyperText Markup Language
HTTP	engl. HyperText Transfer Protocol
IoE	engl. Internet of Everything, kaiken internet
IoP	engl. Internet of People, ihmisten internet
IoS	engl. Internet of Services, palveluiden internet
IoT	engl. Internet of Things, esineiden internet
NoSQL	engl. Not Only SQL
OPC UA	engl. Open Platform Communications Unified Architecture
PaaS	engl. Production as a Service, tuotanto palveluna
PLC	engl. programmable logic controller, ohjelmoitava logiikka
REST	engl. REpresentational State Transfer
SQL	engl. structured query language
WCS	engl. Warehouse control system, Cimcorp Oy:n varastonhallintajärjestelmä

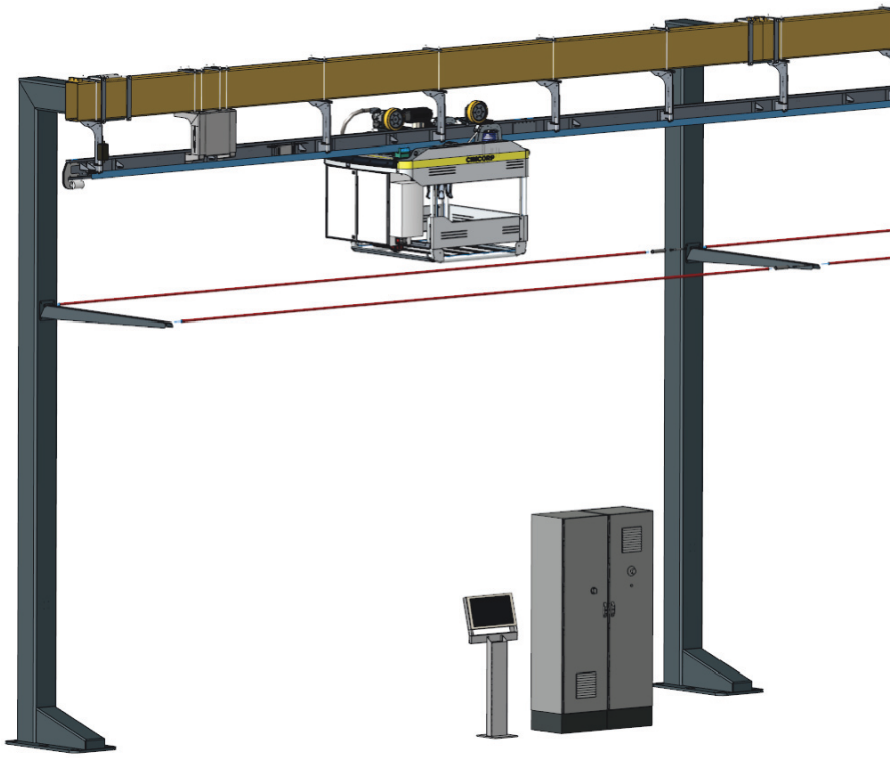
1. JOHDANTO

Tämän diplomityön tarkoituksena on tutkia, miten Cimcorp Oy:n Monorail-automaatiojärjestelmää voitaisiin kehittää datan keräämisen, analysoinnin ja esittämisen avulla, hyödyntäen Teollisuus 4.0:n suunnitteluperiaatteita. Tavoitteena on löytää keinoja Teollisuus 4.0:n kehityksen myötä havaittujen ideoiden konkreettiseen realisointiin.

Teollisuuden neljännestä vallankumouksesta on puhuttu vuodesta 2011 lähtien, kun Saksan hallitus nimesi sen yhdeksi avainaloitteekseen [1]. Lasin mukaan Teollisuus 4.0:n tärkeimpiä konsepteja ovat muun muassa älytehdas ja kyberfysikaaliset järjestelmät [2]. Molempien ajavana voimana toimii data ja kyky käsitellä sitä. Teollisuus 4.0:n konsepteilla uskotaan olevan mahdollisuus valmistuksen, suunnittelun, materiaalien käytön ja elinkaaren hallinnan perustavanlaatuiseen parannukseen [1].

Cimcorp Oy toimittaa sisälogistiikkaan keskittyviä projekteja ympäri maailmaa. Monorail-automaatiojärjestelmä on Cimcorp Oy:n sisälogistiikkaan kehittämä tuote. Järjestelmän avulla voidaan rengastehtaassa palvella erilaisia työkoneita kuljettamalla rengasaihioita tai valmiita renkaita varastojen ja työkoneiden välillä. Monorail-automaatiojärjestelmän pääkomponentit ovat Cimcorp Oy:n varastonhallintajärjestelmä (engl. Warehouse Control System, WCS), soluohjain, vaunut, raide ja operointipaneeli. Järjestelmän fyysiset komponentit on esitetty kuva 1:ssä.

Diplomityön tuloksena kehitetään prototyyppi, jonka avulla pyritään arvioimaan eri ratkaisujen toimivuutta. Prototyypin kehittämisen etuna on abstraktien ideoiden ja ominaisuuksien nopea esittely ja testaaminen [3]. Prototyypistä saatujen tulosten pohjalta pyritään luomaan määrittely tuotantokäyttöön otettavan järjestelmän rakenteesta ja vaatimuksista.



Kuva 1. Monorail-automaatiojärjestelmän raide, vaunu, soluohjain ja operointipaneeli.

1.1 Tutkimusongelma

Monorail-automaatiojärjestelmässä ja sen rajapinnoilla saattaa esiintyä vaikeasti paikannettavia, monen järjestelmän toiminnasta riippuvia, ongelmia. Näitä ongelmia joutuvat ratkomaan niin järjestelmän operaattorit, käyttöönottajat, asiakaspalvelun tuki-insinöörit kuin tuotekehityksen insinöörit.

Monorail-automaatiojärjestelmä on toteutettu hajautetulla arkkitehtuurilla eli sen komponentit on toteutettu omille alustoilleen, ja ne kommunikoivat keskenään verkon yli. Hajautetun arkkitehtuurin vuoksi järjestelmään on mahdollista suunnitella redundanttisia osia, jolloin yhden osan vikaantuminen ei pysäytä koko järjestelmän toimintaa. Esimerkiksi yhden vaunun vikaantuessa muut raiteen vaunut pystyvät hoitamaan sen tehtäviä. Hajautetun arkkitehtuurin varjopuolena on vaikeutunut ongelmien selvittäminen, sillä tarvittava tieto saattaa olla jakautuneena moneen erilliseen alijärjestelmään.

Sama ongelma toistuu myös eri laitetoimittajien laitteiden rajapinnoissa, jolloin ongelman selvittämiseksi saattaa olla tarpeellista saada tietoa toisen toimittajan laitteesta. Tällöin tiedonkulku on entistä vaikeampaa ja voi vaatia useita välikäsiä.

Globaalit toimitusprojektit aiheuttavat myös omat haasteensa, sillä käyttöönoton jälkeen laitteen toimittajan tiedot laitteen toiminnasta ovat paljolti asiakkaan tiedottamisen varassa. Toimittajalla voi olla etäyhteys laitteisiinsa, mutta yhteydet saattavat olla ominaisuuksiltaan rajattuja sekä hitaita. Lisäksi etäyhteyden avulla on vaikea saada yhtä kattavaa kokonaiskuvaa, kuin itse paikalta.

1.2 Tutkimuskysymykset

Tämän diplomityön lähtökohtana on hypoteesi, että datan keräämisen, analysoinnin ja esittämisen avulla edellisessä luvussa esiteltyjen ongelmien ratkaisemista olisi mahdollista helpottaa kaikille sidosryhmille. Tavoitteena ei kuitenkaan ole pelkästään reaktiivinen ongelmien ratkaiseminen. Teollisuus 4.0:n mukanaan tuomat teknologiat mahdollistavat siirtymisen reaktiivisesta ongelmanratkaisusta ennakoivaan ongelmanratkaisuun.

Tämän diplomityön tarkoituksena on suunnitella ja toteuttaa tietojärjestelmä, jonka avulla testataan datan keräämisen, analysoinnin ja esittämisen avulla saatavia hyötyjä Monorail-automaatiojärjestelmässä. Ongelman ratkaisua lähestytään suunnittelutieteellisesti, jolloin tutkimuksen perustehtävät ovat rakentaminen ja arviointi [4]. Rakentamisvaiheen tarkoituksena on näyttää, että artefakti on toteutettavissa [4]. Arviointivaiheessa määritellään kriteerit ja arvioidaan artefaktin toimivuus näiden perusteella [4]. Lisäksi van Aken toteaa suunnittelutieteen tehtäväksi kehittää tietoa artefaktien suunnitteluun ja toteuttamiseen [5]. Näiden tehtävien pohjalta on Thuan *et al*. [6] suunnittelutieteen tutkimuskysymyksille laatimia malleja soveltaen laadittu seuraavat tutkimuskysymykset:

1. Onko Monorail-automaatiojärjestelmän datan hyödyntäminen kannattavaa?
2. Mitkä ovat tärkeimmät komponentit Monorail-automaatiojärjestelmän dataa hyödyntävässä järjestelmässä?
3. Miten nämä komponentit kannattaisi toteuttaa?

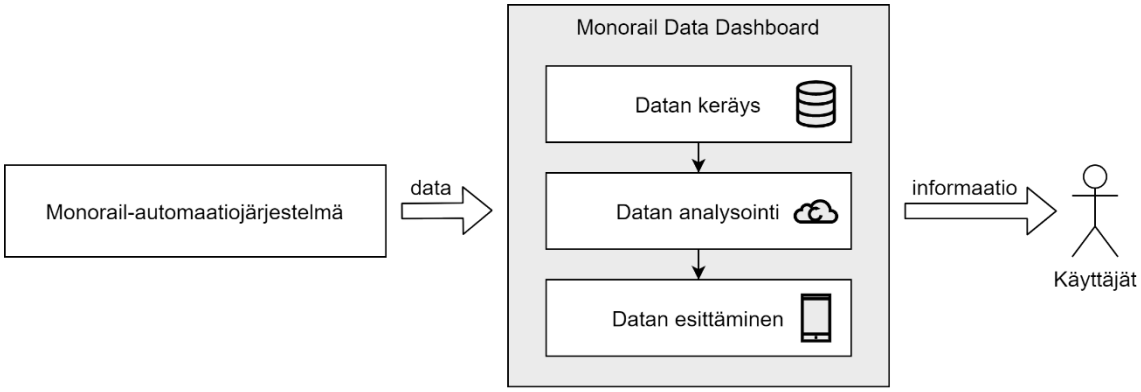
1.3 Työn rakenne

Tässä työssä pyritään noudattamaan Tampereen yliopiston Opinnäytetyön kirjoitusohje tekniikan alalla -julkaisun mukaista rakennetta [7]. Tässä johdantoluvussa pyritään esittelemään tutkimusongelma, tutkimuskysymykset ja kuvaamaan tutkimuksen suoritusvaiheet sekä työn rakenne.

Toteutettavan ohjelmiston rakenne ja toimintaympäristö on esitetty kuva 2:ssa. Järjestelmän on tarkoitus kerätä dataa Monorail-automaatiojärjestelmästä. Datan analysoinnin

ja esittämisen avulla pyritään tuottamaan käyttäjälle automaatiojärjestelmän käyttöä helpottavaa informaatiota. Tavoitteena on, että tuotetun informaation avulla automaatiojärjestelmän käyttöä olisi mahdollista tehostaa.

Luvussa 2 käsitellään työn lähtökohtana olevaa Teollisuus 4.0:llaa. Teollisuus 4.0:n myötä esiin nousseita teknologioita ja suunnitteluperiaatteita pyritään hyödyntämään konstruktiossa lisäarvon tuottamiseksi. Luvussa 3 käsitellään työssä käytettyä konstruktiivista tutkimusmenetelmää yleisesti. Luvussa 4 esitellään Monorail-automaatiojärjestelmä, työn lähtökohdat sekä varsinaisen konstruktion toteutus ja arviointi. Luvussa 4.5 arvioidaan työssä rakennettu konstruktio ja tarkastellaan sen avulla saavutettuja tuloksia. Lopuksi luvussa 5 esitetään työn johtopäätökset.

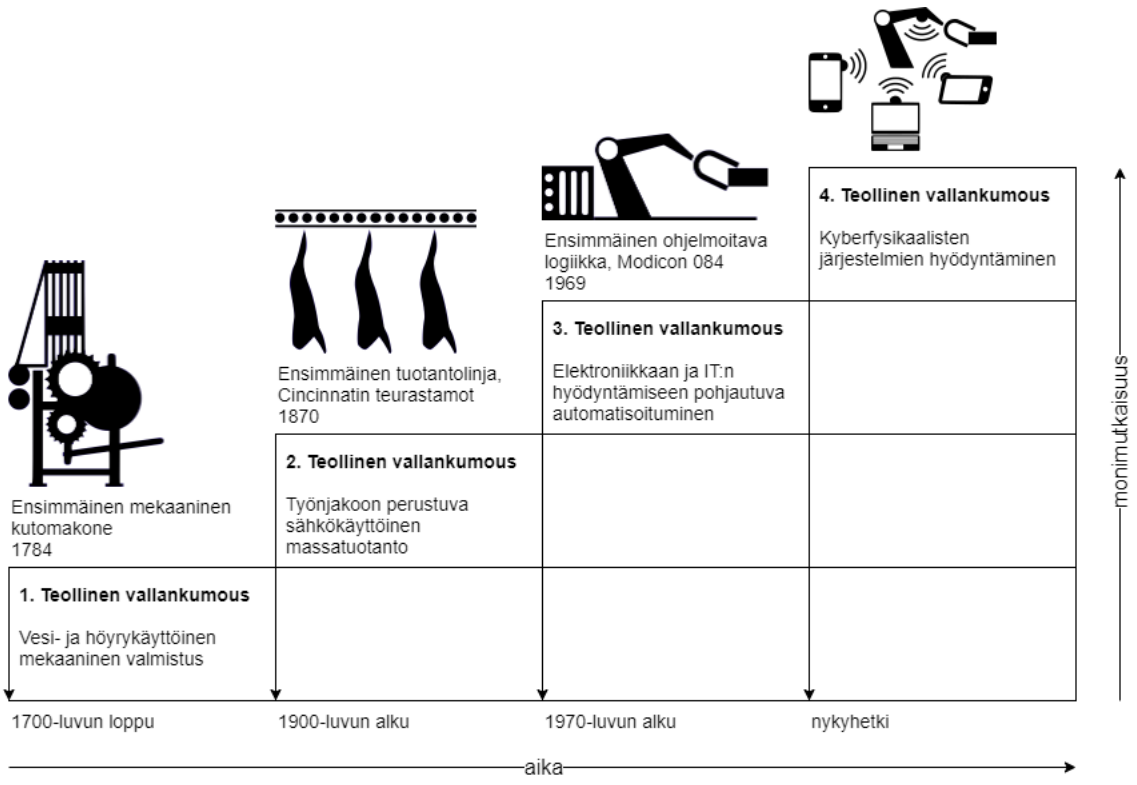


Kuva 2. Monorail Dashboardin rakenne ja toimintaympäristö

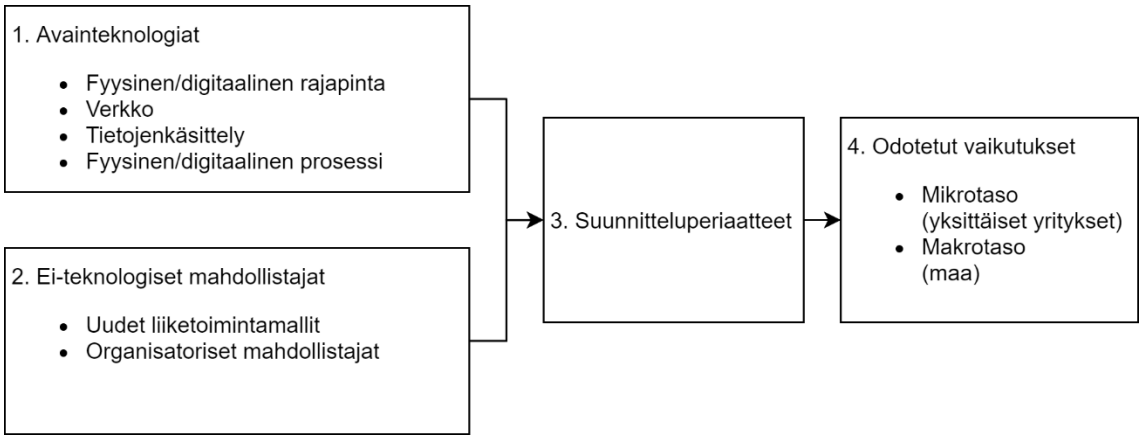
2. TEOLLISUUS 4.0

Teollisuus 4.0 -termillä tarkoitetaan ennakoitua teollisuuden neljättä vallankumousta. Kuva 3:ssa esitetään kolme aikaisempaa teollista vallankumousta, niihin johtaneet teknologiset innovaatiot ja esimerkkejä niiden varhaisista sovelluksista. Samassa kuvassa esitetään myös potentiaalinen neljäs teollinen vallankumous ja Kagermann *et al*. ennustama teknologia, kyberfysikaaliset järjestelmät, sen saavuttamiseksi [1]. Ehkäpä piankin saamme tietää, mikä on aikamme Kehruu-Jenny, kenties AlphaGo Zero?

Teollisuus 4.0 -termin voidaan katsoa syntyneen Hannoverin messuilla 2011, kun Saksan koulutus- ja tutkimusministeriön valtuuttama työryhmä esitteli konseptin [8]. Samanaikaisesti vastaavia aloitteita käynnistyi ympäri maailmaa, kuten esimerkiksi ”Advanced Manufacturing Partnership” Yhdysvalloissa [8], ”European Factories of the Future” Euroopassa [8] ja ”Made in China 2025” Kiinassa [9]. Myös monet teknologia- ja konsultointiyhtiöt julkaisivat aiheeseen liittyen [8]. Terminologia ei ollut vakiintunutta ja samoista aiheista puhuttiin muun muassa nimillä ”Industry 4.0”, ”Industrial Internet”, ”industrial revolution” ja ”smart manufacturing” [8]. Näistä termeistä ”Industry 4.0” eli suomennettuna Teollisuus 4.0 vakiintui lopulta [8].



Kuva 3. Teollisuuden neljä vallankumousta, perustuen lähteeseen [1].



Kuva 4. Teollisuus 4.0:n määrittelevät elementit, perustuen lähteeseen [8].

Käsitteenä Teollisuus 4.0 on hyvin laaja ja lisäksi sitä käsitellään *ex ante*. Tämän vuoksi sen tarkka määrittely ennakoivasti on jopa mahdotonta. Määrittelyä vaikeuttaa esimerkiksi jatkuvasti kehittyvä tekniikka. Periaatteessa on mahdollista, että jokin vielä tuntematon teknologia muodostuu varsinaiseksi uudeksi murroskohdaksi.

Tämänhetkisten näkymien perusteella Teollisuus 4.0:lle on kuitenkin annettu lukuisia määrittelyjä [8]. Culot *et al*. mukaan tärkeimpiä elementtejä määrittelyssä ovat avainteknologiat, ei-teknologiset mahdollistajat, suunnitteluperiaatteet ja odotetut vaikutukset (kuva 4) [8]. Ghobakhloo listaa lisäksi useita julkaisijoita (muun muassa Gilchrist, Liao *et al*., Santos *et al*.), joiden mukaan Teollisuus 4.0 voidaan määritellä käytettyjen teknologioiden ja suunnitteluperiaatteiden perusteella (kuva 4:ssa kohta 1 ja 3) [10].

Seuraavissa alaluvuissa käsitellään kaikki Culot *et al*. tunnistamat elementit. Näistä elementeistä keskitytään kuitenkin avainteknologioihin ja suunnitteluperiaatteisiin. Avainteknologiohin ja suunnitteluperiaatteisiin keskitytään, koska ne ovat suoraan sovellettavissa tämän työn konstruktiiviseen osuuteen. Lisäksi Teollisuus 4.0:n liittyvissä julkaisuissa juuri näihin tekijöihin on kiinnitetty suurta huomiota, jolloin on perusteltua olettaa niiden olevan merkityksellisiä. Elementit käsitellään Culot *et al*. nimeämässä järjestyksessä kuva 4:n mukaisesti.

2.1 Avainteknologiat

Teollisuus 4.0 ei perustu yhteen yksittäiseen avainteknologiaan, vaan koostuu lukuisista toisiaan tukevista teknologioista. Lisäksi teknologiat kehittyvät jatkuvasti yhä kiihtyvällä tahdilla. Tästä huolimatta teknologian kehityssuunnista ja niiden soveltamisesta voidaan hahmottaa trendejä. Näiden avulla voidaan argumentoida joidenkin teknologioiden olevan avainasemassa tulevassa teollisuuden murroksessa.

Culot *et al*. tunnistivat kirjallisuuskatsauksessaan kaksi päätrendiä: fysikaalisen ja digitaalisen maailman lähentymisen sekä paikallisen ja maailmanlaajuisen liitettävyyden. Näiden trendien pohjalta he muodostavat neljä kategoriaa Teollisuus 4.0:n liittyville teknologioille. Kategoriat muodostuvat liitettävyyden (paikallinen/globaali -akseli) ja digitaalisuuden (laite/ohjelmisto -akseli) perusteella. Ensimmäinen kategoria on fysikaalisdigitaalisten rajapintojen teknologioille (engl. physical/digital interface technologies). Tälle kategorialle ominainen teknologia on esimerkiksi esineiden internet (engl. internet of things). Toinen kategoria on verkkoteknologia (engl. network technology), jolle ominainen teknologia on esimerkiksi lohkoketju (engl. blockchain). Kolmas kategoria on tietojenkäsittelyteknologiat, joille ominainen teknologia on esimerkiksi koneoppiminen (engl. machine learning). Neljäs kategoria on fysikaalisdigitaaliset prosessiteknologiat (engl. physical/digital processing technologies), jolle ominainen teknologia on esimerkiksi 3D-tulostaminen. [8]

Culot *et al*. kokoama lista esitetään taulukko 1:ssä. Nämä neljä kategoriaa ja 13 avainteknologiaa kattoivat Culot *et al*. keräämästä materiaalista yli 94 % [8]. Tämä muodostaa riittävän kattavan viitekehyksen, jonka pohjalta on mahdollista hahmottaa Teollisuus 4.0:n liittyviä teknologioita. Tähän viitekehykseen ei kuitenkaan tule liiaksi tukeutua, sillä teknologiat kehittyvät jatkuvasti. Tämän vuoksi ei ole etukäteen mahdollista koota kattavaa listaa teknologioista.

Taulukko 1. Teollisuus 4.0:n avainteknologiat, muokattu lähteestä [8].

Kategoria	Avainteknologia
Fysikaalisdigitaaliset rajapinnat	Esineiden internet
	Kyberfysikaaliset järjestelmät
	Visualisointiteknologiat
Verkkoteknologiat	Pilvilaskenta
	Liitettävyys ja tietoturva
	Lohkoketju
Tietojenkäsittelyteknologiat	Simulointi ja mallintaminen
	Koneoppiminen ja tekoäly
	Massadata-analytiikka
Fysikaalisdigitaaliset prosessiteknologiat	3D-tulostus
	Kehittynyt robotiikka
	Uudet materiaalit
	Energianhallintaratkaisut

2.2 Ei-teknologiset mahdollistajat

Culot *et al*. nimeävät Teollisuus 4.0:n yhdeksi määrittäväksi tekijäksi muut mahdollistajat (engl. other enablers) tai ei-teknologiset mahdollistajat (engl. non-technological enablers) (kuva 4:ssa kohta 2) [8]. Näitä ei-teknologisia mahdollistajia he tunnistavat kaksi: organisatoriset mahdollistajat ja uudet liiketoimintamallit [8].

Organisatorisista mahdollistajista Culot *et al*. suorittama kirjallisuuskatsaus tunnistaa kolme pääkohtaa [8]. Ensimmäisenä organisaatiosuunnittelun tulisi tavoitella parempaa sidonnaisuutta organisaation sisällä ja organisaatioiden välillä [8]. Toisena Culot *et al*. mainitsevat lean-tyylisen, matala hierarkkisen organisaatiorakenteen, jonka tavoitteena tulisi olla hajautettu päätöksenteko [8]. Kolmannen löydöksen mukaan digitaalisia ja strategisia kykyjä tullaan tarvitsemaan organisaatioiden jokaisella tasolla [8].

Uusista liiketoimintamalleista Culot *et al*. tunnistavat kaksi näkökohtaa [8]. Ensimmäinen huomio on, että älytuotteet ja niihin liittyvät datavetoiset palvelut syrjäyttävät perinteisen tuotemyynnin [8]. Toinen liiketoimintamalleja mullistava tekijä ovat uudet tuotantomenetelmät, kuten 3D-tulostaminen [8].

2.3 Suunnitteluperiaatteet

Tämän työn konstruktiivisen osuuden kannalta suunnitteluperiaatteet ovat erityisen kiinnostavia, sillä ne vastaavat kysymykseen ”Kuinka tehdä Teollisuus 4.0:aa?” [11] [10] [8]. Kirjallisuuskatsauksissaan Culot *et al*. listaavat kahdeksan suunnitteluperiaatetta [8], Ghobakhloo tunnistaa kymmenen [10] ja Hermann *et al*. nimeävät neljä [11]. Suunnitteluperiaatteet on listattu taulukko 2:ssa. Näistä Culot *et al*. listaus on tuorein ja hyödyntää Ghobakhloo *et al*. ja Hermann *et al*. aikaisempia tuloksia.

Valitettavasti yksikään edellä mainituista julkaisuista ei tarjoa korkeamman tason luokittelua tunnistamilleen suunnitteluperiaatteille. Lisäksi suunnitteluperiaatteita on tunnistettu eri abstraktiotasoilla. Esimerkiksi Hermann *et al*. nimeävät yhteenliitettävyyden (engl. interconnectivity) yhdeksi suunnitteluperiaatteeksi ja älytehtaat sen osatekijäksi [11]. Ghobakhloo puolestaan mainitsee älytehtaat omana suunnitteluperiaatteenaan [10]. Culot *et al*. taas pitävät älytehdasta Teollisuus 4.0:n verrattavana, koko ilmiötä kuvaavana nimenä [8]. Edellä mainittujen syiden vuoksi suunnitteluperiaatteita on vaikea esitellä yhtenäisenä kokonaisuutena. Tästä huolimatta seuraavissa kappaleissa esitetään lyhyesti taulukko 2:ssa luetellut suunnitteluperiaatteet. Suunnitteluperiaatteita on pyritty ryhmittelemään samankaltaisuuden perusteella. Muutoin ne käydään läpi niiden esitysjärjestyksessä, aloittaen Hermann *et al*. julkaisusta.

Hermann *et al*. nimeävät ensimmäiseksi suunnitteluperiaatteeksi yhteenliitettävyyden. Tällä he tarkoittavat koneiden, antureiden ja ihmisten liittämistä yhteen tilanteessa, jossa IoT (engl. Internet of Things, esineiden internet) ja IoP (engl. Internet of People, ihmisten internet) muodostavat IoE:n (engl. Internet of Everything, kaiken internetin). Heidän mukaansa yhteenliitettävyys tulee mahdollistamaan eri valmistajien modulaaristen laitteiden joustavan liittämisen toisiinsa. Tämä modulaarisuus mahdollistaa Teollisuus 4.0:n älytehtaiden sopeutumisen vaihtelevaan kysyntään ja yksilöityihin tuotteisiin. [11]

Ghobakhloon neljäs suunnitteluperiaate on yhteentoimivuus (engl. interoperability), joka pääsääntöisesti vastaa yhteenliitettävyyttä. Ghonakhloo tunnistaa yhteentoimivuudesta neljä tasoa: toiminnallisen, semanttisen, systemaattisen ja teknisen yhteentoimivuuden. Toimijat, kuten älytuotteet, -tehtaat tai operaattorit ovat yhteentoimivia, jos ne pystyvät yhdistymään, kommunikoimaan ja toimimaan yhdessä. [10]

Ghobakhloon viides ja Culot *et al*. kahdeksas suunnitteluperiaate on modulaarisuus. Se on ajatuksena hyvin lähellä yhteenliitettävyyttä ja yhteentoimivuutta. Modulaarisuudessa Ghobakhloo korostaa erityisesti ketteryyttä kaikilla tuotantoketjun tasoilla. [10] [8]

Ghobakhloo jatkaa saman aihepiirin parissa yhdeksännellä suunnitteluperiaatteellaan, järjestelmäintegraatiolla. Culot *et al*. ensimmäinen suunnitteluperiaate on prosessien integrointi ja se vastaa hyvin järjestelmäintegraation ajatusta. Järjestelmäintegraatio sitoo yhteenliitettävyyden, yhteentoimivuuden ja modulaarisuuden yhdeksi kokonaisuudeksi. Järjestelmäintegraatio eroaa ajatuksena edellä mainituista lähinnä laajuudeltaan. Puhuessaan järjestelmäintegraatiosta Ghobakhloo tarkoittaa sekä horisontaalista että vertikaalista integraatiota koko arvoketjun laajuudelta. [10] [8]

Hermann *et al*. tunnistama toinen suunnitteluperiaate on tiedon läpinäkyvyys. Myös Culot *et al*. tunnistavat reaaliaikaisen tiedon läpinäkyvyyden toiseksi ja Herman *et al*. näkemykseen kuuluvan ennustettavuuden seitsemänneksi suunnitteluperiaatteekseen. Yhteenliitettyjen tahojen kasvava määrä ja digitaalisen ja fyysisen maailman lähentyminen mahdollistavat uudenlaisen tiedon läpinäkyvyyden. Anturidatan liittäminen digitaalisiin malleihin mahdollistaa virtuaalisen kopion luomisen fyysisestä maailmasta. Kontekstitietoisen informaation tarjoaminen IoE:n tahoille on tärkeää oikeiden päätösten tekemiseksi. Kontekstitietoiset järjestelmät suorittavat tehtävänsä yhdistelemällä tietoja fyysisestä ja digitaalisesta maailmasta. Fyysisen maailman analysoimiseksi raaka anturidata pitää koota korkeammalle tasolle tulkittavaksi. Läpinäkyvyyden luomiseksi dataanalyysin tulokset pitää upottaa avustusjärjestelmiin kaikkien IoE:n tahojen saataville. [11] [8]

Ghobakhloon kahdeksas ja Culot *et al*. neljäs tunnistama suunnitteluperiaate on virtualisointi. Se on hyvin lähellä Hermann *et al*. tunnistamaa tiedon läpinäkyvyyden periaatetta. Ghobakhloo kertoo virtualisoinnin mahdollistavan digitaalisen kaksosen luomisen koko arvoketjusta. Digitaalinen kaksonen syntyy yhdistämällä fyysisen maailman anturidata virtuaalisiin malleihin. Digitaalisen kaksosen avulla olisi esimerkiksi mahdollista optimoida prosesseja häiritsemättä tuotannossa olevia laitteita. Digitaalinen kaksonen liittyy myös vahvasti Ghobakhloon esille nostamaan älytuotteeseen ja -tehtaaseen, joita käsitellään myöhemmin. [10] [8]

Kolmas Hermann *et al*., seitsemäs Ghobakhloon ja neljäs Culot *et al*. (nimellä autonomisuus) tunnistama suunnitteluperiaate on hajautettu päätöksenteko. Hajautetut päätökset perustuvat yhteenliitettyihin laitteisiin ja ihmisiin sekä informaation läpinäkyvyyteen. Yhteenliitettyjen ja hajautettujen päätöksentekijöiden yhdistelmä mahdollistaa paikallisen ja maailmanlaajuisen tiedon hyödyntämisen kokonaistuottavuuden parantamiseksi. IoE:n tahojen tulisi suorittaa tehtävänsä mahdollisimman itsenäisesti ja ylätason tulisi osallistua vain poikkeustapauksissa. Hermann *et al*. mukaan kyberfysikaaliset järjestelmät ovat avainasemassa hajautetun päätöksenteon mahdollistajina. [11] [10] [8]

Viimeinen Hermann *et al*. tunnistama suunnitteluperiaate on tekninen avustaminen. Teollisuus 4.0:ssa ihmisen päärooli siirtyy operaattorista strategiseksi päätöksentekijäksi ja joustavaksi ongelmanratkojaksi. Hajautettujen kyberfysikaalisten järjestelmien muodostamat tuotantojärjestelmät kasvattavat monimutkaisuuden tasolle, jossa ihmisiä on tuettava avustusjärjestelmillä. Näiden järjestelmien on kerättävä ja visualisoitava tietoa ymmärrettävään muotoon varmistaakseen, että ihmiset pystyvät tekemään perusteltuja päätöksiä ja ratkaisemaan kiireellisiä ongelmia lyhyellä varoitusajalla. Kehittyvän robotiikan odotetaan tulevaisuudessa mahdollistavan ihmisten fyysisen tukemisen epämiellyttävissä, uuvuttavissa tai vaarallisissa tehtävissä. Robottien tulee olla sujuvasti ja intuitiivisesti vuorovaikutuksessa ihmisten kanssa tehokkaan tuen aikaansaamiseksi. [11]

Ghobakhloon tunnistamista suunnitteluperiaatteista palvelulähtöisyys on ensimmäinen. Culot *et al*. nimeävät palvelulähtöisyyden (engl. product servitization) viidentenä ja tuotannon palvelullistamisen kuudentena (engl. servitization of manufacturing capabilities) suunnitteluperiaatteenaan. Teollisuus 4.0 -näkökulmasta nämä tarkoittavat verkottuneen tuotantoinfrastruktuurin yhteiskäyttöä tuotteiden valmistamiseksi PaaS-mallin mukaisesti (engl. Production as a Service, tuotanto palveluna). Valmistajien yhteenliittyvyys, IoT ja pilvilaskenta mahdollistavat uuden tuotantoympäristön, jossa yritykset voivat ilmoittaa tuotantotarpeensa ja -kapasiteettinsa automaattisesti. Tässä tuotantoympäristössä monimutkaiset valmistustehtävät voidaan suorittaa monen eri valmistajan yhteistyönä. Tämä kehitys muuntaisi valmistajien päätuotteen fyysisistä tuotteista valmistuskapasiteettiin. [10] [8]

Toisena suunnitteluperiaatteena Ghobakhloo nostaa esiin älytuotteet. Älytuote hyödyntää antureitaan keräämällä, tallentamalla ja siirtämällä dataa ympäristöstään elinkaarensa eri vaiheissa. Esimerkiksi tuotantovaiheessa älytuote pystyy kertomaan, missä se on valmistettu, mikä seuraava tuotantovaihe on ja kuinka se pitäisi suorittaa. Kulutusvaiheessa älytuotteet ja IoS-infrastruktuuri (engl. Internet of Services, palveluiden internet) edesauttavat PaaS-liiketoimintamallin leviämistä. Ghobakhloon kolmas suunnitteluperiaate on älytehdas. Älytehdas vastaa ajatukseltaan vahvasti älytuotetta. Älytehtaalla Ghobakhloo tarkoittaa erittäin tuottavaa valmistusympäristöä, jossa yhteenliitetyillä älykkäillä koneilla minimoidaan hävikki, virheet ja seisokit. [10]

Kuudes Ghobakhloon tunnistama suunnitteluperiaate on räätälöinti. Teknologiat, kuten kyberfysikaalisetjärjestelmät, IoT, avoin tuotearkkitehtuuri ja 3D-tulostus, mahdollistavat tuotteiden uudelleenkonfiguroinnin jatkuvasti muuttuvien asiakkaiden mieltymysten mukaisiksi. Mieltymyksiä on mahdollista tunnistaa käyttäytymisen arvioinnin ja ennakoimisen avulla. Tällöin valmistajille ei riitä pelkästään asiakkaiden olemassa olevien vaati-

musten täyttäminen, vaan heidän tulee myös ennakoida asiakkaidensa tulevat vaatimukset ja mieltymykset. Tämä on mahdollista hyödyntämällä simulaatioita ja massadata-analytiikkaa markkinatrendien ja asiakastarpeiden ennustamiseen. [10]

Viimeinen Ghobakhloon mainitsema suunnitteluperiaate on yhteiskuntavastuu. Muiden suunnitteluperiaatteiden kohdalla julkaisijoilla oli huomattavasti samankaltaisia ajatuksia, mutta Ghobakhloo on ainoa, joka nosti yhteiskuntavastuun esille. Yhteiskuntavastuulla Ghobakhloo tarkoittaa yritysten itsesääntelyä, joka integroituu liiketoimintamalleihin. Teollisuus 4.0 odotetaan tuovan mullistuksia heikosti koulutetuille työntekijöille, joten Teollisuus 4.0:aan valmistautuvien yritysten kannattaisi panostaa työntekijöiden kouluttamiseen. Yhteiskuntavastuuseen liittyy myös vastuu ympäristöstä. Teollisuus 4.0 tarjoaa valtavia mahdollisuuksia kestävään kehitykseen tuotteiden, materiaalien ja energian tehokkaan koordinoinnin avulla koko tuotteen elinkaaren ajan. [10]

Taulukko 2. Suunnitteluperiaatteet julkaisijoittain, perustuen lähteisiin [11] [10] [8].

Hermann *et al*.	**Ghobakhloo**	**Culot *et al*.**
Yhteenliitettävyys	Palvelulähtöisyys	Prosessien integrointi
Tiedon läpinäkyvyys	Älytuotteet	Reaaliaikainen tiedonläpinäkyvyys
Hajautettu päätöksenteko	Älytehdas	Virtualisointi
Tekninen avustaminen	Yhteentoimivuus	Autonomisuus
	Modulaarisuus	Palvelulähtöisyys
	Räätälöinti	Tuotannon palvelullistaminen
	Hajauttaminen	Ennustettavuus
	Virtualisointi	Modulaarisuus
	Järjestelmäintegraatio	
	Yhteiskuntavastuu	

2.4 Odotetut vaikutukset

Culot *et al*. mukaan yksi Teollisuus 4.0:aa määrittävä tekijä on sen soveltamisesta seuraavat vaikutukset (kuva 4:ssa kohta 4) [8]. Culot *et al*. kirjallisuuskatsauksen perusteella Teollisuus 4.0:n odotetut lopputulokset jaotellaan mikro- ja makrotasoon vaikutusten laajuuden perusteella [8]. Mikrotason vaikutuksilla tarkoitetaan yksittäisen yrityksen tasolle ulottuvia vaikutuksia. Makrotasosta puhutaan, kun vaikutukset näkyvät maatasolla tai laajemmin.

Mikrotason vaikutukseksi Culot *et al*. lähteissä nousivat yleisesti kaikki perinteiset suorituskyvyt [8]. Kaikkein yleisimmät odotetut vaikutukset olivat tuottavuuden ja joustavuuden parantuminen massaräätälöinnin tasolle [8]. Culot *et al*. mukaan nämä odotukset ovat linjassa sen tulevaisuuden skenaarion kanssa, jossa kysyntä muuttuu hajanaisemmaksi ja epävakaammaksi sekä matala- ja korkeatulotason maiden välisten työvoimakustannusten erot pienentyvät [8]. Muita yleisiä mainintoja olivat ympäristöystävällisyys sekä markkinoilletuontiaika ja -kustannus [8]. Yllättävää Culot *et al*. mielestä oli suhteellisen vähäinen huomio laatuun ja läpivientiaikaan, vaikka näihin liittyy useita avainteknologioita [8].

Makrotasoon liittyviä julkaisuja ei Culot *et al*. käsittelemässä materiaalissa ole läheskään mikrotasoon verrattavaa määrää [8]. Makrotason vaikutuksena Teollisuus 4.0:n odotetaan tuovan talouskasvua, laajentavan automaation soveltamisalaa ja nopeuttavan sen käytön lisääntymistä [8]. Teollisuus 4.0:n uskotaan myös vaikuttavan työllisyyteen. Culot *et al*. mukaan monet julkaisijat uskovat työllisyyden kehityksen olevan myönteistä perustuen aikaisempien innovaatioiden vaikutuksiin [8].

3. SUUNNITTELUTIETEELLINEN TUTKIMUSMENETELMÄ

Tässä luvussa käsitellään yleisesti innovaation toteuttamista suunnittelutieteellisen tutkimusmenetelmän avulla. Tämän diplomityön tarkoituksena on suunnitella ja toteuttaa tietojärjestelmä, jonka avulla testataan datan keräämisen, analysoinnin ja esittämisen avulla saatavia hyötyjä Monorail-automaatiojärjestelmässä. Koska ei ole olemassa aikaisempaa järjestelmää, joka toteuttaisi juuri tätä rajattua toiminnallisuutta, on kyseessä uuden innovaation toteuttaminen. Van Akenin mukaan tavoitteena on siis konstruktio-ongelman ratkaiseminen, jota suunnittelutiede käsittelee [5].

Pentti ja Annikki Järvinen ovat koonneet yhteen lukuisia näkemyksiä suunnittelutieteestä ja muodostaneet niistä kehyksen suunnittelutieteellisen tutkimuksen tekemiselle [12]. Järvisten esittämä kehys jakautuu kahteen päävaiheeseen: innovaation toteuttamiseen ja sen arviointiin [12]. Tämä kehys vastaa Marchin ja Smithin tunnistamia suunnittelutieteen perustehtäviä eli artefaktin suunnittelua ja arviointia [4]. Järvisten kehystä hyödyntämällä on tarkoitus saada vastaukset johdannossa määriteltyihin tutkimuskysymyksiin ja saada aikaa toimiva prototyyppi.

Järvisten kehyksen tukena käytetään taulukko 3:een koottuja Hevnerin *et al*. laatimia suunnitteluperiaatteita [13]. Järvisten kehystä ja Hevnerin *et al*. suunnitteluperiaatteita hyödynnetään diplomityössä, jotta se täyttäisi paremmin taulukko 3:n viidennen suunnitteluperiaatteen vaatimuksen tutkimuksen perusteellisuudelle.

Valmiin kehyksen tarkoituksena on tuoda työlle luotettavuutta ja uskottavuutta sekä helpottaa sen toteuttamista. Työn toteuttaminen helpottuu, koska kehys tarjoaa selkeän vaihejaon ja vaatimuksia vaiheille. Kehystä ja suunnitteluperiaatteita käyttämällä on mahdollista hyödyntää usean ansioituneen tutkijan kokemusta ja siten säästyä rakenteen uudelleenkeksimisen vaivalta.

Taulukko 3. Suunnittelututkimuksen periaatteet, muokattu lähteestä [13].

Suunnitteluperiaate	Kuvaus
Periaate 1: Suunnittelu artefaktina	Suunnittelututkimuksen tulee tuottaa toimintakykyinen artefakti.
Periaate 2: Ongelman merkitys	Suunnittelututkimuksen tavoitteena on kehittää teknologiaan perustuva ratkaisu merkitykselliseen liiketoimintaongelmaan.
Periaate 3: Suunnittelun arviointi	Artefaktin hyödyllisyys, laatu ja tehokkuus tulee osoittaa hyvillä arviointimenetelmillä.
Periaate 4: Tutkimustulokset	Tehokkaan suunnittelututkimuksen on tuotettava selkeitä ja todennettavissa olevia tuloksia artefaktin suunnitteluun tai toteutukseen liittyen.
Periaate 5: Tutkimuksen perusteellisuus	Suunnittelututkimus pohjautuu perusteellisten menetelmien soveltamiseen artefaktin suunnittelussa ja arvioinnissa.
Periaate 6: Suunnittelu hakuprosessina	Tehokkaan ratkaisun etsiminen vaatii käytössä olevien resurssien hyödyntämistä tavoitteiden saavuttamiseksi ongelmakentän vaatimuksen huomioon ottaen.
Periaate 7: Tutkimuksen viestintä	Suunnittelututkimus on esitettävä sekä teknisille että johdon sidosryhmille.

3.1 Innovaation toteuttaminen

Innovaation toteuttaminen on suunnittelututkimuksen perustehtävä [4]. Taulukko 3:sta löytyvä Hevnerin ensimmäinen suunnitteluperiaate painottaa artefaktin toteuttamisen merkitystä [13]. Konkreettinen toteutus helpottaa luontaisesti abstraktin tietojärjestelmän hahmottamista [3]. Konkreettinen hahmottaminen on tärkeää, sillä se yhdistää ongelman ratkaisua eri taustoista ja näkökulmista lähestyviä sidosryhmiä.

Innovaation toteuttamiseen on lukuisia tapoja. Innovaation toteuttamisesta Järviset ovat tunnistaneet seuraavat vaiheet: lähtötila, spesifiointiprosessi, implementointiprosessi,

valmisosan hankinta, saavutettu lopputila ja tavoitetila [12]. Kuva 5:ssa on esitetty Järvisten kokoamia tapoja pyrkiä lähtötilasta tavoitetilaan näiden vaiheiden kautta.

Innovaation toteuttaminen lähtee ongelman tunnistamisesta ja ideasta sen ratkaisemiseksi. Järviset määrittelevät tämän idean tarkemmin. Heidän mukaansa ”kyse on jonkin resurssin, teknisen, inhimillisen tai tiedollisen resurssin taikka niiden yhdistelmän, hyödyntämisestä.” [12] He myös korostavat tämän idean raportoinnin merkitystä [12]. Näiden tietojen pohjalta muodostuu lähtötilan määrittely. Lähtötila on Järvisten mukaan lähes aina tunnistettavissa [12].

Spesifiointiprosessin tarkoituksena on tuottaa tavoitetilan kuvaus [12]. Tavoitteena voi olla joko vanhan innovaation parantaminen tai täysin uuden suunnittelu [12]. Järviset suosittelevat kirjallisuustutkimusta innovaation uutuuden varmistamiseksi [12]. Tarkoituksena on välttää pyörän uudelleen keksimistä. Tämä on tärkeää sekä tutkimuksen että taloudellisuuden näkökulmasta. Spesifioinnissa pitää ottaa huomioon käytettävissä olevien resurssien asettamat rajoitteet [12]. Lisäksi eri sidosryhmillä saattaa olla toisistaan poikkeavia tai jopa vastakkaisia pyrkimyksiä, jotka pitää ottaa huomioon spesifioinnissa [12].

Implementointiprosessissa pyritään spesifiointiprosessista saadut vaatimukset muuttamaan käytettävissä olevilla resursseilla innovaatioksi. Innovaation toteuttaminen myös osoittaa, että innovaatio oli toteutettavissa, joka ei ole lähtötilanteessa itsestään selvää [12].

Implementointia voidaan lähestyä jollakin ongelmaratkaisun heuristiikalla, kuten esimerkiksi ongelmareduktion heuristiikalla [12]. Ongelmareduktiossa pääongelma jaetaan pienempiin osaongelmiin, jotka ratkaisemalla myös pääongelma ratkeaa. Jos osaongelma ei ratkea, voidaan siihen edelleen rekursiivisesti soveltaa ongelmareduktiota, kunnes ongelma on tarpeeksi pieni ratkaistavaksi.

Ongelmaa voidaan lähestyä myös syvyys ensin -periaatteen avulla tai leveys ensin -periaatteen avulla [14]. Syvyys ensin -periaatteella pyritään ratkaisemaan osaongelmia ja rakentamaan kokonaisuutta näiden ratkaisujen päälle. Esimerkiksi riskiohjattu prosessi toimii syvyys ensin -periaatteella, sillä siinä pyritään ratkaisemaan suuri riskisin osaongelma ensin. Leveys ensin periaatteella pyritään hahmottamaan kokonaisuus ja viivyttämään ratkaisuihin sitoutumista mahdollisimman pitkään, jolloin muutoksista aiheutuvat kustannukset ovat mahdollisimman pieniä [14]. Huomattavaa on, että kumpikin periaate vaatii sovellusalan osaamista [14]. Syvyys ensin -periaatteessa tulee keskittyä oikeaan osaongelmaan ja leveys ensin -periaate vaatii kokonaisuuden hahmottamista [14].

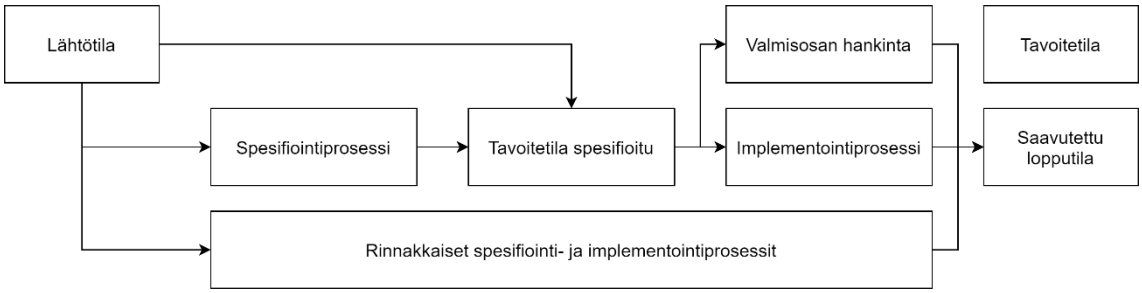
Valmisosan hankinnan ideana on välttää pyörän uudelleen keksimistä [12]. Ohjelmistoalalla moneen sovellukseen on saatavilla kaupallinen ratkaisu, joka voidaan sovittaa innovaation vaatimuksiin [3]. Valmiin tuotteen sovittaminen voi olla halvempaa ja nopeampaa, kuin oman ratkaisun kehittäminen [3]. Myös avoimen lähdekoodin ratkaisut voivat säästää aikaa ja rahaa, jos sovellusalalle löytyy laadukkaita julkaisuja [3]. Tuotteiden sijaan myös palveluja tai aineettomia hyödykkeitä on mahdollista hankkia ulkopuolisilta toimittajilta [12]. Hankintaprosessi muodostuu kandidaattien selvittämisestä, niiden vertailusta, hankintapäätöksestä sekä toimituksesta ja hyväksymisestä [12].

Spesifiointia ja implementointia on mahdollista toteuttaa myös rinnakkain. Järvisten mukaan innovaation asteittainen kehittäminen johtuu usein siitä, että ihmisten on vaikea kuvitella sellaista, mitä koskaan ei ole ollut olemassa [12]. Tämän takia tavoitetilan määrittely on vaikeaa [12].

Innovaation iteratiivinen ja inkrementaalinen kehitys auttaa paremmin käsittämään teknologian tuomat mahdollisuudet. Uusien käsitysten myötä myös tavoitetilan määrittely helposti muuttuu, kun uusia mahdollisuuksia tunnistetaan. Määrittelyjen muuttuminen johtaa usein projektin laajuuden muuttumiseen, jolloin myös aikataulut ja resurssitarpeet herkästi muuttuvat.

Ohjelmistojärjestelmien joustavuuden vuoksi ohjelmistoprojektien vaatimusten muuttuminen on yleisempää, kuin muissa projekteissa. Tämän vuoksi vaatimuksien muutoksista johtuvien ongelmien ratkaisemiseen on ohjelmistoalalla kiinnitetty erityistä huomiota. Yksi käytetyimmistä tavoista selvitä muuttuvien vaatimuksien kanssa on hyödyntää ketteriä ohjelmistokehityksen menetelmiä (engl. agile methods). Ketteriä menetelmiä alettiin kehittämään 1990-luvulla vastineena vallalla olleisiin raskaampiin menetelmiin [3]. Agile manifeston julkaisu vuonna 2001 vakiinnutti ketterien menetelmien termin ja filosofian.

Innovaation toteuttamisen ja käyttöönoton jälkeen on saavutettu lopputila. Lopputilaa verrataan tavoitetilaan ja tarvittaessa prosessi aloitetaan alusta, jotta tavoitetila saavutettaisiin [12]. Lopputila saattaa poiketa tavoitetilasta olemalla joko arvioitua huonompi tai parempi [12]. Lopputilan saavuttaminen joka tapauksessa osoittaa, että innovaatio oli toteutettavissa [12]. Innovaatio on hyödyllinen, jos sen avulla saavutettu lopputila on pysyvä parannus lähtötilaan [12]. Jotta mahdollinen parannus ja sitä kautta innovaation hyödyllisyys voitaisiin osoittaa, pitää innovaatio arvioida. Innovaation arviointia käsitellään tarkemmin alaluvussa 3.2.



Kuva 5. Vaihtoehtoisia tapoja toteuttaa innovaatio perustuen Järvisten kuvaukseen [12].

3.2 Innovaation arviointi

Innovaation systemaattinen arvioiminen on ainoa tapa saada luotettavaa tietoa sen toimivuudesta. Marchin ja Smithin mukaan arviointi on, rakentamisen lisäksi, suunnittelutieteen toinen perustoiminto [4]. Innovaation arviointia vaikeuttaa se, että innovaatiota voidaan soveltaa monessa eri tilanteessa, vaihtelevalla menestyksellä [4].

Innovaation arvioinnissa on kaksi vaihetta: innovaation käyttöympäristöön soveltuvien mittarien määrittely ja varsinainen arviointi näillä mittareilla [4]. March ja Smith esittävät joukon valmiita mittareita erityyppisille innovaatioille (käsitteistöille, malleille, metodeille ja realisaatioille). Lisäksi Järviset ovat koonneet useista eri julkaisuista Marchia ja Smithiä täydentäviä mittareita erityyppisille innovaatioille [12]. Tässä työssä toteutetaan realisaatio, joten kiinnostuksen kohteena ovat realisaation arviointiin soveltuvat mittarit. Järvisten kokoamia mittareita innovaation realisaatiolle ovat:

1. Tehokkuus
2. Vaikuttavuus
3. Vaikutukset ympäristölle ja käyttäjille
4. Sosiaaliset vaikutukset
5. Poliittiset vaikutukset
6. Historialliset vaikutukset
7. Investointien arviointi
8. Huolto- ja kokonaiskustannukset [12].

Marchin ja Smithin mukaan realisaatioiden arvioinnissa tulisi huomioida niiden tehokkuus (engl. efficiency) ja vaikuttavuus (engl. effectiveness). Tehokkuus ja vaikuttavuus ovat Marchin ja Smithin [4] käyttämiä termejä, joista käytetään Järvisten suomennosta [12]. Termit itsessään ovat hyvin laajoja ja niiden käännökset eivät täysin vastaa toisiaan. Lähtökohtana tässä työssä käytetään lähdekielisiä määritelmiä. Karkeasti ottaen tässä

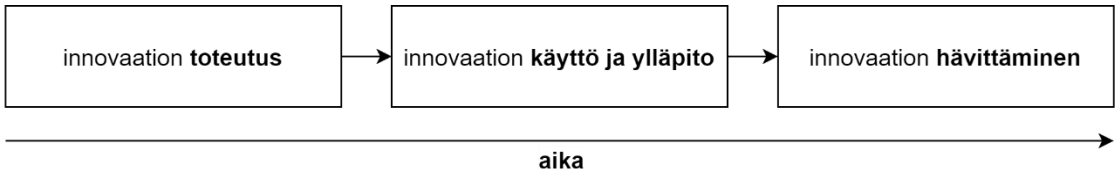
kontekstissa tehokkuudella tarkoitetaan tavoitteen saavuttamiseen käytettyjen panosten ja siitä saatujen hyötyjen suhdetta. Vastaavasti vaikuttavuudella tarkoitetaan kykyä saavuttaa tavoite.

Järvisten mukaan Marchin ja Smithin vaikuttavuuden arviointi [4] keskittyy suunniteltujen vaikutusten arviointiin ja niiden lisäksi tulisi huomioida myös Orlikowskin [15] painottamat, positiiviset tai negatiiviset, odottamattomat vaikutukset [12]. Näiden lisäksi Kling kiinnitti huomiota informaatiosysteemien vaikutusten arvion tiukkaan rajaamisen teknologiaan ja sen välittömään käyttöön [16]. Hänen mukaansa informaatiosysteemejä tulisi arvioida myös sosiaalisissa, poliittisissa ja historiallisissa konteksteissa [16].

Tämän lisäksi Dalhbom ja Mathiassen erottavat innovaatioon, kulttuuriin ja valtaan keskittyvät näkökulmat [17]. Myös innovaation vaikutus sen ympäristöön ja käyttäjiin tulisi huomioida [4]. Mittarien lisäksi on tärkeää arvioida missä määrin asetetut tavoitteet on saavutettu [12].

Tehokkuutta ja vaikuttavuutta arvioidaan usein taloudellisesta näkökulmasta. Tämän vuoksi mittareita suunniteltaessa on hyvä tiedostaa laskentatoimen perusongelmat: laajuusongelma (mitkä hyödyt ja kustannukset lasketaan mukaan), mittaamisongelma (miten hyödyt ja kustannukset mitataan), arvostusongelma (miten hyötyjä ja kustannuksia arvostetaan) ja jakamisongelma (miten hyödyt ja kustannukset jaetaan tuotteille ja palveluille) [12].

Innovaatioiden arvioinnissa tulisi ottaa huomioon niiden koko elinkaari. Elinkaari voidaan jakaa toteutukseen, käyttöön ja ylläpitoon sekä hävittämiseen kuva 6:n mukaisesti [12]. Varsinaista lisäarvoa innovaatio tuottaa vasta, kun se on otettu käyttöön, mutta luonnollisesti uuden luominen vaatii resursseja ja tuottaa siten kuluja. Tämän vuoksi toteutusvaihe on tärkeä huomioida arvioinnissa. Naiivi arvio voisi olla, että innovaation kulut syntyvät sen toteutuksessa ja käyttövaiheessa olisi mahdollista vain nauttia sen hyödyistä. Sommerville kuitenkin arvioi, että ohjelmistotuotteissa suurin osa kuluista, karkeasti arvioiden noin kaksi kolmasosaa, syntyy käyttövaiheessa ja vain yksi kolmasosa toteutusvaiheessa [3]. Käyttövaiheen kulut voidaan jakaa virheenkorjauksesta, ympäristöön adaptoitumisesta sekä toiminnallisuuden muokkaamisesta ja lisäämisestä syntyviin kuluihin [3].



Kuva 6. Innovaation elinkaari perustuen Järvisten kuvaukseen [12].

Arvioinnin kannalta on huomattava, että vasta innovaation elinkaaren loputtua on mahdollista tarkasti laskea sen tuottamat hyödyt ja kustannukset kokonaisuudessaan [12]. Innovaation käytön ja ylläpidon lopettamisen jälkeen se hävitetään. Innovaatioon käytetyt resurssit pitää joko siirtää uudelleenkäytettäväksi tai niiden käyttö pitää lopettaa [12]. Jokaisella resurssilla on silloin tietty jäännösarvo tai hävityskustannus [12].

Konkreettisten sovellusten hävittäminen vaatii niiden sisältämien materiaalien hävittämisen tai kierrättämisen, mutta myös tietoteknisten ja inhimillisten resurssien poistaminen pitää huomioida. Inhimilliset resurssit ovat yksilöiden ja ryhmien hiljaista tietämystä, jonka hankkimiseen on mennyt pitkä aika [12]. Suurin osa tästä saattaa olla käytettävissä uudenkin innovaation yhteydessä [12]. Tämän vuoksi innovaatio olisi hyvä suunnitella niin, että sen käyttämisestä opitut taidot olisi mahdollista siirtää myös muihin sovelluksiin. Tämä onnistuu esimerkiksi suosimalla vakiintuneita käytäntöjä käytettävyyden suunnittelussa. Myös tarpeettoman inhimillisen tietämyksen poisoppiminen vaatii ponnistuksia [12]. Esimerkiksi jos vanha käyttöliittymä on toiminut erikoisella tavalla, vanhan työtekijän siirtyminen uuteen järjestelmään saattaa olla jopa vaikeampaa uuteen työntekijään verrattuna.

Inhimillisten ja konkreettisten resurssien lisäksi myös tietoteknisistä resursseista tulee huolehtia. Vanhat tietokannat luovat pohjan uusille sovelluksille [12] ja niihin liittyvät tarpeettomat tiedot pitää tuhota, sillä väärinkäytettynä ne saattavat aiheuttaa muun muassa henkilötietoihin liittyviä ongelmia [12]. Samoin myös tarpeettomat ohjelmakomponentit pitää poistaa, sillä ne kuluttavat resursseja ja saattava aiheuttaa epätoivottuja sivuvaikutuksia, jos ne jäävät unohdettuina ajoon. Laadukkaita tietoteknisiä resursseja, kuten lähtötietoja, vaatimusmäärittelyjä tai ohjelmistokomponentteja on myös mahdollista käyttää uudelleen [12].

Käyttäjän näkökulmasta Monorail Dashboardin tarkoituksena on ensisijaisesti helpottaa heidän työtänsä. Tämän arvioinnissa Dahlbomin ja Mathiassenin tunnistamat roolit auttavat [17]. Roolien avulla sovellusta voidaan arvioida insinöörin, fasilitoijan ja emansipoijan näkökulmista.

Insinöörin näkökulmasta käyttäjää avustetaan kehittämällä parempia sovelluksia [17]. Insinöörin näkökulmasta sovellus on käyttäjälle hyvä, jos sen avulla haluttu tulos on nopeasti saatavissa. Toisaalta, jos käyttäjä joutuu panostamaan huomattavasti aikaansa saavuttaakseen halutun lopputuloksen, ei sovellus ole hyvä. Esimerkiksi tilanne on huono, jos käyttäjä joutuu odottamaan tietokantakyselyiden tai analyysien tuloksia pitkään.

Fasilitoijan tavoitteena on kehittää käyttäjän pätevyyttä [17]. Fasilitoija pyrkii kehittämään teknologiaa palvelemaan ihmistä mahdollisimman hyvin [17]. Onnistunut sovellus auttaa käyttäjää ratkaisemaan ongelmia aikaisempaa helpommin ja lisää ymmärrystä järjestelmän tilasta ja toiminnasta. Epäonnistunut sovellus puolestaan tuo käyttäjälle lisähaasteita. Sovellus on epäonnistunut esimerkiksi tilanteessa, jossa käyttäjä joutuu syöttämään tietoa järjestelmään, muttei saa panokselleen vastinetta. Sovellus on myös epäonnistunut, jos sen käyttäminen on vaikeaa tai vaatii huomattavasti opiskelua.

Emansipoijan tarkoituksena on käyttää teknologiaa yhteiskunnan ja organisaatioiden edistämiseen [17]. Emansipoijan näkökulmasta on tärkeää suojella ihmisiä tietotekniikan haittapuolilta [17]. Esimerkiksi työntekijän toimien liian tarkka ja automatisoitu tarkkailu saattaa eriarvoistaa eri rooleissa toimivat työntekijät. Toisaalta informaation läpinäkyvyyden lisääntyminen demokratisoi informaatiota. Parhaimmillaan älykkäät sovellukset helpottavat työntekijöiden työtehtäviä ja vapauttavat heidät tylsistä ja toistuvista tehtävistä.

Järvisten listaamien realisaatioita koskevien mittarien viimeinen kohta on ohjelmiston huolto- ja kokonaiskustannukset [12]. Lano ja Haughton jakavat huoltokustannukset korjaavaan, sopeuttavaan, parantavaan ja ehkäisevään huoltoon [18]. Sommerville puolestaan tunnistaa korjaavan, ympäristöön sopeuttavan ja toiminnallisuutta lisäävän huollon [3]. Sommervillen mukaan huoltokustannukset vastaavat noin kahta kolmasosaa ohjelmiston kustannuksista ja ohjelmiston kehittäminen noin yhtä kolmasosaa [3]. Huoltotyypeittäin kustannukset ovat noin 17 % korjaavalle, 18 % ympäristöön sopeuttavalle ja 65 % toiminnallisuutta lisäävälle huollolle [3]. Sommervillen luvut perustuvat dataan vuosien 1980–2005 väliltä ja hänen mukaansa ne ovat pysyneet ajan suhteen yllättävän muuttumattomina [3]. Lukujen tarkat arvot eivät ole tässä tärkeitä, mutta ne on hyvä tiedostaa mittareita suunniteltaessa. Huomattavaa on, että sovellusalalla on merkittävä vaikutus kustannusten jakautumiseen [3].

Analytiikkasovelluksissa tarve uusille toiminnoille saattaa olla erityisen korkea, sillä nopeasti muuttuvissa tilanteissa myös analyysitarpeet muuttuvat. Esimerkiksi, jos tämän hetken suurin ongelma pystytään jonkin tuotetun analyysin avulla selvittämään, nousee seuraavaksi suurin ongelma puolestaan suurimmaksi. Tällöin tämä uusi ongelma vaatii

luultavasti oman analyysinsä kehittämisen. Analyysien tuottaminen on perinteisesti tapahtunut jonkun muun, kuin analyysiä hyödyntävän tahon puolesta. Tämä aiheuttaa viiveitä tarpeen tunnistamisesta analyysin hyödyntämiseen. Liiketoimintatiedon hyödyntämisen puolella käyttäjien tarpeet ovat johtaneet itsepalvelumallin lisääntymiseen [19]. Itsepalvelumallissa tiedon hyödyntäjille tarjotaan mahdollisimman hyvät edellytykset analyysien itsenäiseen tekemiseen, jolloin on mahdollista saavuttaa aikaisempaa parempi tehokkuus [19]. Tämä kehitys on huomattava indikaattori analytiikkaan liittyvien sovellusten jatkuvaan muutostarpeeseen.

4. MONORAIL DASHBOARDIN TOTEUTTAMINEN

Tämän luvun alussa esitellään Monorail-automaatiojärjestelmä ja lähtötilanteen tavat hyödyntää dataa. Seuraavaksi esitellään toteutustapa ja sen valintaan liittyvä pohdinta. Lopulta esitellään konstruktiivisen tutkimuksen ydin eli varsinainen toteutus ja sen arviointi.

4.1 Monorail-automaatiojärjestelmän kuvaus

Cimcorp Oy toimittaa sisälogistiikkaan keskittyviä projekteja ympäri maailmaa. Logistiikkasovelluksille ominaista on liittyminen useisiin muihin järjestelmiin. Monorail-automaatiojärjestelmä on Cimcorp Oy:n sisälogistiikkaan kehittämä tuote. Järjestelmän avulla voidaan rengastehtaassa palvella erilaisia työkoneita kuljettamalla rengasaihioita tai valmiita renkaita varastojen ja työkoneiden välillä. Yleisimpiä työkoneita ovat renkaan kokoonpanokoneet ja paistopuristimet. Varastoina käytetään yleensä Cimcorp Oy:n portaalirobottisolua tai korkeavarastoa. Monorail-automaatiojärjestelmä toimii usein tuotantokoneiden rajapintana Cimcorp Oy:n järjestelmiin.

Monorail-automaatiojärjestelmän pääkomponentit ovat Cimcorp Oy:n varastonhallintajärjestelmä (engl. Warehouse Control System, WCS), soluohjain, vaunut, raide ja operointipaneeli. Kuva 1 esittää järjestelmän komponentit WCS:ää lukuun ottamatta. WCS on Cimcorp Oy:n tuotannonohjausjärjestelmä. Sen tehtävänä on huolehtia varastoista, tuotteiden seurannasta, tuotetietojen hallinnasta ja tuotantokoneiden materiaalitarpeista. WCS vastaanottaa tiedot tuotetuista rengasaihioista kokoonpanokoneilta ja materiaalipyynnöt paistopuristimilta. Näiden pohjalta materiaali reititetään kuljetinjärjestelmän avulla poimintapaikalle. Poimintapaikalla olevasta materiaalista WCS lähettää Monorailin soluohjaimelle poiminta- ja jättötehtävät.

Soluohjaimen tehtävänä on liittää WCS ja automaatiolaitteet yhteen. Soluohjain on toteutettu ohjelmoitavalla logiikalla (engl. Programmable Logic Controller, PLC). Fyysisesti laitteet toimivat raiteen ympärillä. Raiteella voi olla useita vaunuja ja siihen voi liittyä useita tuotantokoneita poiminta- ja jättöpaikkojen avulla. Yksi soluohjain voi ohjata usean raiteen laitteita. Soluohjain välittää WCS:n lähettämät tehtävät vaunuille ja vaunujen palauttamat kuittaukset sekä työkoneiden informaation WCS:lle. Soluohjain välittää radalla liikkuville vaunuille tiedot niiden naapurien liikkeistä. Vaunut käyttävät tätä informaatiota törmäyksenestoalgoritmeissään. Soluohjain toimii myös multiplekserinä (sekä demulti-

plekserina) vaunujen ja poiminta- ja jättöpaikkojen välisille kättelysignaaleille. Multipleksausta tarvitaan, koska se vähentää tarvittavien linkkien määrää merkittävästi. Multipleksaamalla on mahdollista keskittää paikkariippuvaisten signaalien välitys yhdelle laitteelle. Ilman multipleksausta jokaiselta poiminta- ja laskupaikalta pitäisi tuoda omat signaalit jokaiselle vaunulle. Soluohjaimen vastuulle kuuluvat lisäksi koneturvallisuuteen liittyvät ohjausjärjestelmät. Turvajärjestelmän avulla voidaan pysäyttää koko raide tai estää tarttujan liikkeet tietyllä alueella.

Monorail-automaatiojärjestelmän tehtävänä on kuljettaa renkaat WCS:n tehtävien mukaisesti poimintapaikoilta jättöpaikoille. Vaunu on soluohjaimen tapaan toteutettu ohjelmoitavalla logiikalla. Vaunu saa absoluuttisen paikkatiedon viivakoodinauhalta ja pystyy liikkumaan radalla molempiin suuntiin. Vaunun tarttuja pystyy liikkumaan vertikaalisesti ja tarttumaan eri halkaisijalla oleviin renkaisiin. Lisäksi vaunussa on avattava luukku, joka estää renkaita tippumasta vaunusta, vaikka ne vahingossa irtoaisivat tarttujasta. Lisäksi turvajärjestelmä estää luukun avaamisen, ellei turva-alue ole aktiivisena. Vaunun liikkeet on toteutettu paikkaohjatuilla servoakseleilla. Vaunut kommunikoivat soluohjaimen kanssa langattomasti, joten niiden tulee pystyä toimimaan myös lyhyiden yhteyskatkosten aikana. Ennen renkaan poimintaa tai jättöä vaunu ja poiminta- tai jättöpaikka kättelee siirron. Kättelyssä varmistetaan muun muassa, onko positioon sallittua ajaa ja ovatko tuotetiedot oikein (esimerkiksi varmistetaan, onko poimintapaikalla anturitiedon mukaan rengasta).

Solun käyttöliittymänä toimii operointipaneeli. Operointipaneelilta on mahdollista hallita vaunuja ja nähdä niihin liittyviä tilatietoja. Operointipaneelilta vaunut on mahdollista asettaa automaattitilaan, operoida niitä manuaalisesti tai pysäyttää ne. Vaunuja on mahdollista operoida yksittäin tai raidekohtaisesti. Operointipaneelin lisäksi manuaalisia toimintoja on mahdollista suorittaa myös kauko-ohjaimen avulla. Vaunun tilatiedoista paneelille tuodaan muun muassa akseleiden positiot, anturien tilat, aktiivisena oleva tehtävä ja virhetiedot. Virhetiedot jäävät myös historiakantaan, josta niitä on mahdollisuus selailla erilaisten suodattimien avulla.

4.2 Datan hyödyntäminen lähtötilanteessa

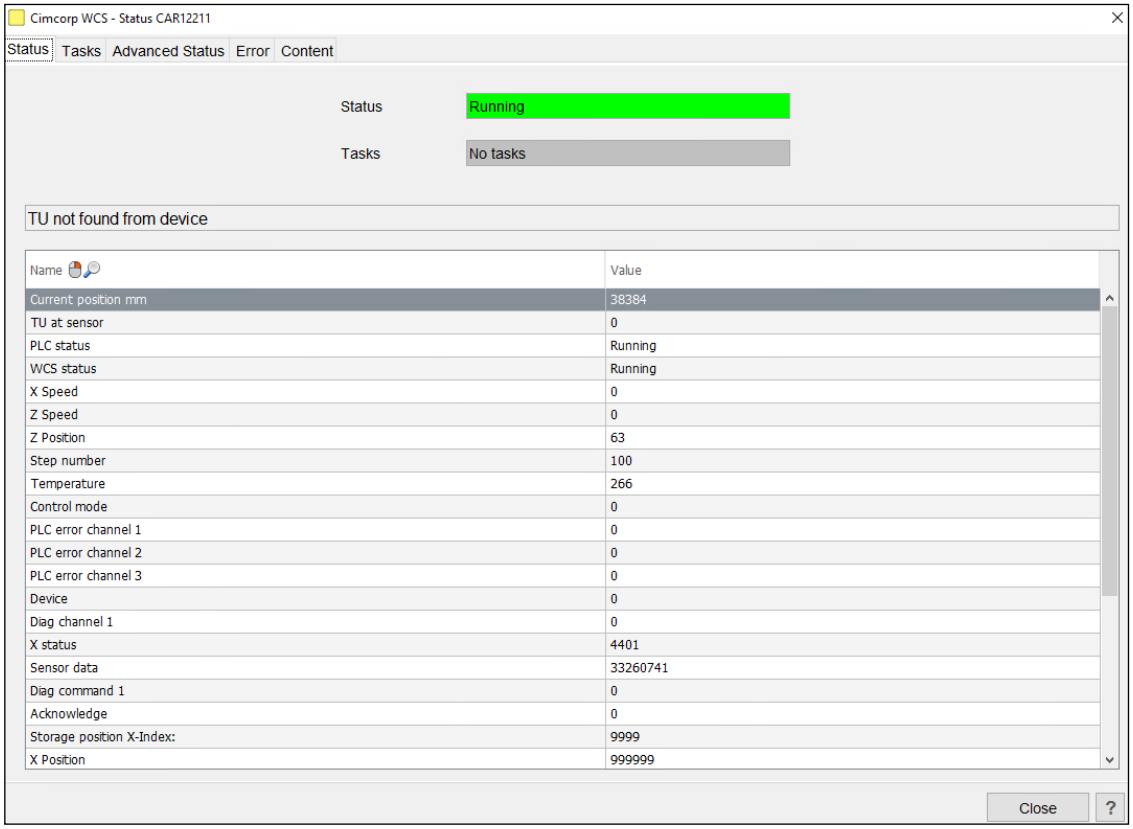
Cimcorp Oy:n Monorail-automaatiojärjestelmän datan kerääminen, analysointi ja esittäminen perustuivat lähtötilanteessa pääasiassa kahteen eri järjestelmään. Ensimmäinen näistä järjestelmistä on WCS, joka ylemmän tason ohjausjärjestelmänä vastaa tuotevirtojen ja –tietojen hallinnasta. Toinen järjestelmä on operointipaneeli, jonka avulla operaattorit käyttävät laitetason järjestelmiä, kuten vaunuja.

WCS:n graafisella käyttöliittymällä esitetään tehtaan pohjapiirros. Pohjapiirroksesta on mahdollista saada yleiskuva tehtaan toiminnasta. Esimerkiksi Monorail vaunujen liikkeet on havainnollistettu spatiaalisesti 2D-tasossa ja tilatieto värikoodaamalla. Pohjapiirroksesta on mahdollista valita yksittäinen vaunu, jolloin aukeaa kuva 7:n havainnollistama ikkuna. Ikkunasta on mahdollista seurata useita vaunun tilatietoja reaaliaikaisesti. Näistä tärkeimpiä ovat tila (engl. status), joka kertoo vaunun käyttötilan, ja tehtävä (engl. task), joka kertoo vaunulle annetun tehtävän. Kuva 7:ssä esimerkiksi vaunu on automaattitilassa ja sillä ei ole aktiivista tehtävää. Reaaliaikaisten tilatietojen lisäksi graafiselta käyttöliittymältä on nähtävissä päiväkohtaisesti vaunun virhehistoria.

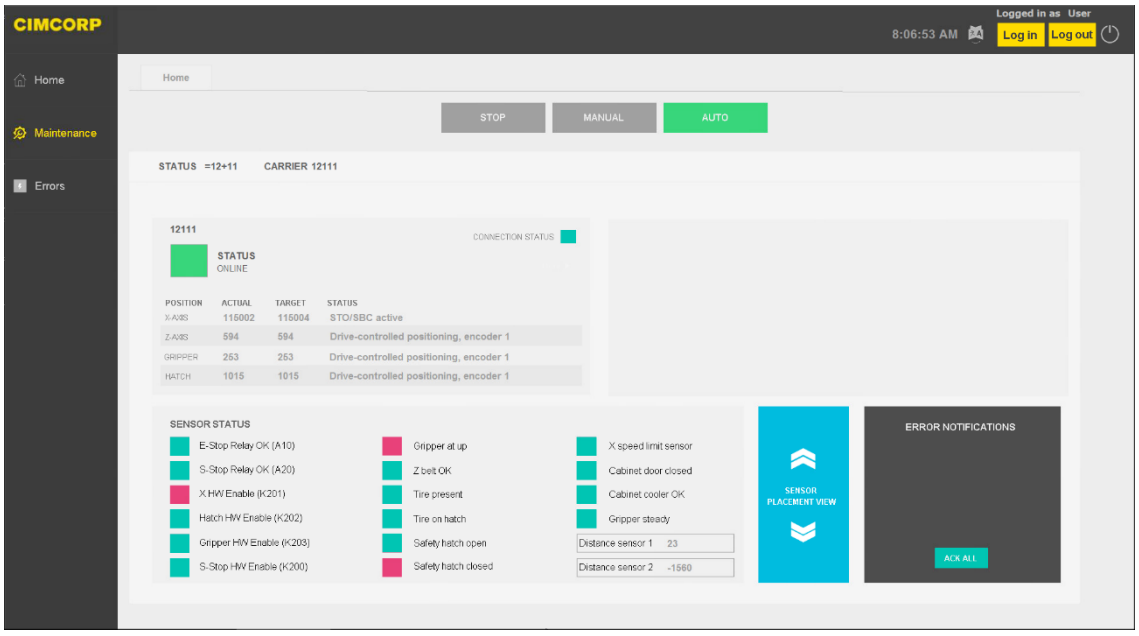
Graafisen käyttöliittymän lisäksi vaunun toiminnasta jää merkintöjä WCS:n lokitiedostoihin. Lokimerkintöjen pääasiallinen käyttötarkoitus ei ole laitetta koskevien käyttötietojen tallentaminen tai esittäminen, vaan ohjelman toimintaan liittyvien ongelmien ratkaiseminen. Tämän vuoksi lokiin merkittävät tapahtumat ja niiden formaatti vaihtelevat projekteittain. Tästä huolimatta lokitiedostoista on joissakin tapauksissa mahdollista kerätä Monorail-järjestelmän toiminnasta dataa, esimerkiksi ongelmatilanteiden ratkaisemiseksi.

Operointipaneelilta on mahdollista nähdä järjestelmän raiteiden vaunujen tilat tai katsoa tarkemmin yhden vaunun tietoja. Operointipaneelin tilanäyttö yksittäiselle vaunulle on kuva 8:n mukainen. Tilanäytöltä on nähtävissä muun muassa vaunun tila, servoakselien positiot, kohteet ja tilat sekä vaunun anturien tiloja. Tilanäytöltä on myös mahdollista nähdä aktiiviset virheet. Virhenäytöltä on mahdollista nähdä kaikki aktiiviset virheet tai virhehistoria raide- tai vaunukohtaisesti.

Datan keräämisen, analysoinnin ja esittämisen näkökulmasta Monorail-automaatiojärjestelmän käyttöliittymät on suunniteltu tukemaan laitteen operointia. Laitteen hetkelliset tilatiedot ja aktiiviset virheet ovat helposti ja kattavasti nähtävissä. Tällöin operaattorit pystyvät vaivattomasti seuraamaan laitteen toimintaa ja puuttumaan nopeasti mahdollisiin ongelmiin.



Kuva 7. WCS:n tilanäyttö vaunulle



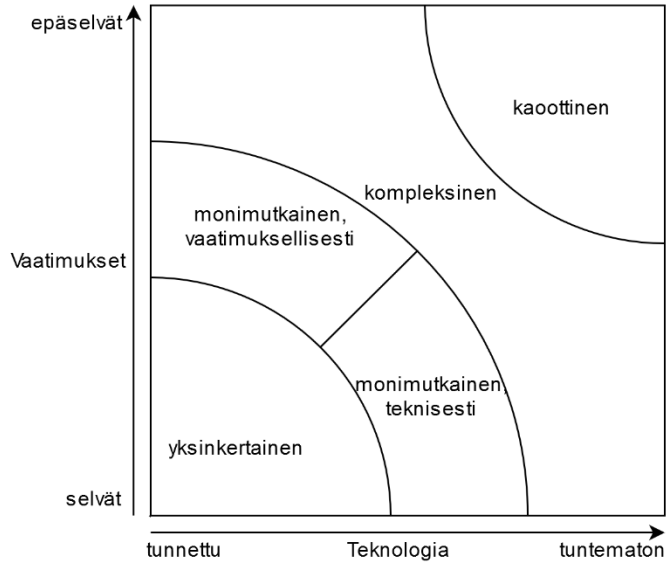
Kuva 8. Operointipaneelin tilanäyttö vaunulle

4.3 Tavoite ja toteutustavan valinta

Järvisten esittämistä innovaation toteuttamistapojen (esitetty kuva 5:ssä) vaiheista ensimmäinen eli lähtötilan tunnistaminen on suoritettu. Lähtötilan tunnistamisen lisäksi tunnetaan ratkaistava ongelma ja sen ratkaisuksi on tarjolla konsepti. Järvisten määritelmä konseptille on alaluvussa 3.1 mainittu: ”jonkin resurssin, teknisen, inhimillisen tai tiedollisen resurssin taikka niiden yhdistelmän, hyödyntäminen.” [12] Tämän työn konsepti tutkimusongelman ratkaisemiseksi on luvussa 2 käsitellyn Teollisuus 4.0:n tuomien teknillisten ja tiedollisten resurssien hyödyntäminen. Järvisten mallissa lähtötilasta voidaan pyrkiä kohti tavoitetilaa kolmea eri polkua pitkin (kuva 5) [12]. Koska tavoitetilaa ei ole spesifioitu, valittavaksi jää joko vaihejakoinen tai rinnakkainen toteutustapa. Järvisten mallissa vaihejakoisessa toteutustavassa tavoitetila spesifioidaan, jonka jälkeen suoritetaan varsinainen implementointiprosessi [12]. Vaihejakoinen toteutustapa ohjelmistonäkökulmasta tarkoittaa käytännössä vesiputousmallin mukaista ohjelmistokehitystä. Järvisten kuvaamassa rinnakkaisessa tavassa spesifiointi ja toteutus suoritetaan samanaikaisesti [12]. Rinnakkainen toteutustapa ohjelmistopuolella tarkoittaa käytännössä ketterien ohjelmistokehitysmenetelmien hyödyntämistä.

Modernissa ohjelmistokehitysprojektissa hyödynnetään ketteriä menetelmiä lähes oletusarvoisesti. Mutta ketterien menetelmien lisäksi valittavissa on muitakin kelvollisia vaihtoehtoja. Esimerkiksi Monorail-automaatiojärjestelmän turvallisuuteen liittyvä sovellusohjelmisto on toteutettu v-mallin mukaisesti. V-mallia on hyödynnetty, koska koneturvallisuus standardin SFS-EN ISO 13849-1 mukainen toteutus edellyttää sitä [20]. V-mallin etuna on keskittyminen validointiin ja todentamiseen. Tämän vuoksi se on hyödyllinen malli koneturvallisuuteen liittyvissä sovellusohjelmistoissa ja muissa kriittisissä sovelluksissa. Koska käypiä malleja on useita, täytyy soveltuvan mallin valinta tehdä perustellusti.

Kuva 9:ssä on Stacey-matriisin ideaan perustuen kuvattu ohjelmistokehityksen monimutkaisuutta vaatimusten selkeyden ja teknologian tunnettavuuden suhteen. Vaihejakoinen toteutus vaatii, että toteutus suunnitellaan etukäteen. Jotta toteutus olisi suunniteltavissa etukäteen, tulee toteutusta koskevat vaatimukset ja ne toteuttava teknologia tuntea. Toisin sanoen ollaan tilanteessa, jossa tunnetaan mitä pitää tehdä ja tiedetään, kuinka se on tehtävissä. Kuva 9:ssä tällaiset tilanteet kuuluvat yksinkertainen-alueelle ja vaihejakoiset toteutustavat soveltuvat niihin hyvin [21]. Tarkka suunnittelu ja työvaiheiden optimointi mahdollistavat tasalaatuisen ja toistettavan tuloksen tämän tyyppisissä projekteissa [21].



Kuva 9. Ohjelmistokehitykseen mukautettu Stacey-matriisi, perustuen lähteeseen [21].

Kun vaatimukset tai sovellettavat teknologiat eivät ole hyvin tunnettuja, siirrytään monimutkaiselle (engl. complicated) alueelle. Monimutkaisella alueella ongelma kannattaa yrittää jakaa pienempiin osaongelmiin ja ratkaista ne [21]. Esimerkiksi vaatimuksellisesti monimutkainen ongelma saattaa ratketa vaihejakoisesti, kunhan määrittelyvaiheessa pystytään päättämään tavoitteet [21]. Teknisesti monimutkainen ongelma puolestaan saattaa ratketa ketterästi iteroimalla [21].

Jos vaatimukset ovat epäselviä ja sovellettava teknologia tuntematonta, on ongelma kompleksinen (engl. complex) [21]. Kompleksisen ongelman ratkaisemisessa kokonaisvaltaisesti määritellyt prosessimenetelmät eivät enää toimi, vaan ketterien menetelmien käyttäminen on välttämätöntä [21]. Kompleksiset ongelmat vaativat tutkivaa lähestymistapaa, avoimuutta ja toistuvaa tarkastelua sekä mukautumista [21]. Jurgen Appelo toteaakin: ”Kompleksisia järjestelmiä ei rakenneta, vaan ne kasvatetaan.” [22] Tällä alueella oleville projekteille on ominaista aikaisemman kokemuksen puute vastaavista toteutuksista [21]. Projektin edetessä saatetaan myös törmätä yllättäviin tuntemattomiin tekijöihin (engl. unknown unknowns) [21].

Ääripäässä ongelma muuttuu kaoottiseksi. Tällöin sekä toteutustapa että tavoitteet ovat tuntemattomia ja riski epäonnistua on suuri [21]. Tästä johtuen ketteryyden tarve korostuu entisestään. Kaoottista ongelmaa ratkottaessa tavoitteena on siirtyä kaaoksesta kohti kompleksisuutta keskittymällä jonkin osaongelman ratkaisemiseen [21].

Sopivan toteutustavan valitsemiseksi on, aikaisempien kappaleiden perusteella, tunnistettava vaatimusten selkeys sekä tieto sovellettavista teknologioista. Monorail Dashboar-

din tarpeen lähtökohtana oleva ongelma tunnetaan ja sen ratkaisuksi on tarjolla Teollisuus 4.0:aa hyödyntävä konsepti. Konsepti ei kuitenkaan ole niin selkeä, että olisi järkevästi määriteltävissä lista vaatimuksia, joiden täytyttyä ongelma olisi ratkaistu. Tämän lisäksi konseptia on vaikea esitellä eri sidosryhmille abstraktilla ideatasolla. Tämän takia myöskään sidosryhmiltä ei ole mahdollista helposti kerätä vaatimuksia. Voidaankin siis arvioida, että vaatimukset eivät ole selkeitä, mutta toisaalta eivät myöskään täysin tuntemattomia.

Sovellettava teknologia ei myöskään ole täysin selvillä. Taulukko 1:ssä mainituista teknologioiden kategorioista kuitenkin kolme neljästä eli fysikaalisdigitaaliset rajapinnat, verkkoteknologiat ja tietojenkäsittelyteknologiat vaikuttavat soveltuvan hyvin tutkimusongelman ratkaisemiseen. Teollisuus 4.0:n avainteknologioista varsinkin esineiden internet, kyberfysikaaliset järjestelmät ja visualisointiteknologiat tulevat oletettavasti olemaan osa ratkaisua. Potentiaalisten teknologioiden tunnistaminen ei kuitenkaan tarkoita, että niitä olisi mahdollista onnistuneesti soveltaa, varsinkaan kun niiden käytöstä ei ole kokemusta. Teknologioidenkin osalta voidaan arvioida, että kyseessä ei ole rutiininomainen soveltaminen, mutta toisaalta ei myöskään täysin tuntematon tilanne.

Vaatimus/teknologia -akseleilla päädytään siten kuva 9:n kompleksille alueelle. Tämän takia Järvisten kuvaamasta mallista [12] valitaan rinnakkainen toteutustapa, joka on kompleksiselle alueelle ominainen [21]. Ketterän ohjelmistokehityksen ajatusten mukaan projektin tavoite ja vaatimukset määritellään alkuvaiheessa kevyesti ja korkealla abstraktiotasolla.

4.4 Toteutus

Ensimmäisen spesifiointikierroksen jälkeen aloitettiin implementointi. Toteutusta ohjasivat tunnistettujen vaatimusten lisäksi alaluku 2.3:ssa kuvatut Teollisuus 4.0:n suunnitteluperiaatteet. Näistä varsinkin tiedon läpinäkyvyys, tekninen avustaminen ja virtualisointi vaikuttivat erityisen osuvilta tavoitteiden saavuttamiseksi. Käyttäjien avustamisen laajaa vaatimusta lähestyttiin ongelmareduktion heuristiikalla. Tiedon läpinäkyvyydestä ja kerätyn tiedon jalostamisesta tekniseksi avustamiseksi tunnistettiin kolme osaongelmaa: datan kerääminen, datan analysointi ja datan visualisointi. Näitä osaongelmia käsitellään seuraavissa alaluvuissa.

4.4.1 Datan kerääminen

Teollisuus 4.0 ja siihen liittyvät avainteknologiat, kuten esineiden internet ja kyberfysikaaliset järjestelmät ovat tuoneet uusia mahdollisuuksia datan keräykseen teollisista jär-

jestelmistä. Monorail-automaatiojärjestelmä hyödyntää Ethernet pohjaista kommunikointia sekä WCS-solu -tasolla että solu-vaunu -tasolla. WCS-solu -tasolla kommunikointi hyödyntää Common Industrial Protocol –pohjaista kommunikaatiota. Solu-vaunu -tasolla käytetään EtherNet/IP pohjaista kommunikointia.

Modulaarisuuden suunnitteluperiaatetta noudattava ratkaisu datan keräämiseen olisi hyvin voinut olla uuden itsenäisen, esimerkiksi OPC UA:n (engl. Open Platform Communications Unified Architecture) perustuvan, järjestelmän kehittäminen. Huomattava määrä laitedataa lähetään joka tapauksessa WCS:ään ja sen ajoittainen tallentaminen onnistuu helposti. Tämän takia päätettiin hyödyntää jo olemassa olevaa infrastruktuuria ja kehittää sitä inkrementaalisesti datakeräyksen näkökulmasta.

Tämän pohjalta syntyi datankeräyksen ensimmäinen versio. Tässä versiossa WCS:n ja vaunun välinen kommunikointi tallennetaan tietyin väliajoin (yleensä noin 0,5–1 sekunnin välein) comma-separated values -tiedostomuodossa (CSV) WCS-palvelimen levylle. Tällä tekniikalla näytteenottotaajuutta on vaikea lisätä, sillä sitä rajoittavat sekä käytetty kommunikointiprotokolla että kommunikoivien laitteiden laskentateho ja määrä. On huomattava, että näytteidenottojen välillä tapahtuvat muutokset eivät tallennu.

Tämän ongelman ratkaisemiseksi PLC:n puolelle toteutettiin lokitustoiminnallisuus, jossa valitut tiedot tallennetaan PLC:n muuttujiin ja nollataan, kun WCS on lukenut ne. Tällä tekniikalla on mahdollista tallentaa dataa PLC:n kiertotaajuudella eli noin millisekuntitasolla. Tekniikan ongelmana on, että kerättävät datapisteet on määriteltävä PLC:n ohjelmassa ja ne käyttävät PLC:n rajallista muistia.

WCS-palvelimella on rajallinen määrä levytilaa, jota tarvitaan Monorail-automaatiojärjestelmästä kerätyn datan lisäksi myös muihin tarkoituksiin. Tämän vuoksi data tallennetaan laitekohtaisesti päivittäin omaan tiedostoonsa, jota säilytetään viikon ajan. Ratkaisun ongelmana on, että tiedot katoavat, jos niitä ei tallenneta muualle viikon kuluessa. Koska WCS palvelin on asiakkaan tiloissa tehtaalla, pääsy dataan vaatii etäyhteyden muodostamista. Yhteydenotto ja tiedostojen siirtäminen pidempiaikaiseen tallennuspaikkaan vaatii jonkin verran toistuvaa käsityötä. Tämän vuoksi se ei ole toimiva ratkaisu laajemmassa mittakaavassa. Toisaalta automatisoitu ja tietoturvallinen etäyhteyden muodostaminen ja tiedostojen siirto ja tallennus ei myöskään ole triviaali ongelma. Projektin alkuvaiheessa tämä oli kuitenkin hyväksyttävä kompromissi, sillä nopea pääsy dataan oli ensiarvoisen tärkeää.

Datan keräämisen seuraava iteraatio oli datan tallennusmuodon kehittäminen. Datan keräämisen aloitusvaiheessa tiedon määrä oli vähäistä, eikä analyysin välivaiheita välttä-

mättä tarvinnut tallentaa. Jo muutaman kuukauden päästä tietoa oli niin paljon, että tietojen aggregointi raakadatasta alkoi olla työlästä. Lisäksi päiväkohtaisten tiedostojen säilyttäminen, käsittely ja jakaminen olivat kömpelöä. Tämän vuoksi tarvittiin parempi ratkaisu tiedon tallentamiselle. Käytännössä ainoa vaihtoehto tiedostopohjaiselle tallentamiselle on tietokanta.

Tietokannan suunnittelussa valittiin, ajetaanko kantaa itse vai ostetaanko se pilvipalveluna sekä tietokannan toteutustapa. Kanta päätettiin tässä vaiheessa toteuttaa Cimcorp Oy:n sisäverkossa, sillä varsinkin pilvessä toteutettuna kannan tietoturva epäilytti. Kanta olisi varmasti toteutettavissa turvallisesti pilvipalvelunakin, mutta se vaatisi hyvää suunnittelua ja perehtymistä. Koska projekti oli vasta prototyyppivaiheessa, tietoturvan varmistaminen olisi hidastanut kehitystyötä. Asiakkaalle ei myöskään ollut tarjota riskin vastineeksi vastaavaa hyötyä, sillä potentiaalista lisäarvoa tuottavien analyysien ja visualisointien kehittäminen oli vasta alkuvaiheessa.

Toinen tärkeä valinta kannalle oli sopivan toteutustavan päättäminen. Eri tietokantatyyppejä on valtavasti, mutta ne voidaan karkeasti jakaa kahteen kategoriaan: SQL:ää (engl. structured query language) hyödyntäviin relaatiokantoihin ja trendikkäisiin NoSQL-kantoihin (engl. Not Only SQL). NoSQL kannoista testattiin muun muassa dokumenttipohjaista MongoDB:tä sekä aikasarjoihin perustuvaa InfluxDB:tä. Lopulta kantatyypiksi valikoitui kuitenkin perinteinen SQL-kanta.

Perinteinen SQL-kanta sopi hyvin rakenteisen laitedatan tallentamiseen ja oli yksinkertainen toteuttaa tehokkaiden työkalujen avulla. Esimerkiksi Pythonin kehittyneitä kirjastoja hyödyntämällä CSV-tiedoston luku ja kirjoitus SQL-kantaan onnistuu vain muutamalla rivillä, ohjelma 1:ssä kuvatulla tavalla. Lisäksi SQL-kannaksi oli valittavissa mikä tahansa SQLAlchemyn tukemista versioista. Tässä työssä kehitys aloitettiin PostgreSQL:llä ja myöhemmin kanta vaihdettiin melko vaivattomasti Oraclen SQL-kantaan.

Eräs tämän työn tärkeimmistä mahdollistajista olivat pitkälle kehittyneet kirjastot. Näiden kirjastojen avulla oli mahdollista nopeasti testata ja kehittää datan käsittelyyn liittyviä ohjelmistoja. Jos datan keräämisen apuohjelmia olisi lähdetty kehittämään itse, olisi varsinaisten tulosten saamiseen mennyt huomattavasti pidempi aika ja projekti olisi luultavasti jäänyt tekemättä. Nyt nopeasti saaduilla, lupaavilla, tuloksilla oli mahdollista perustella projektin hyödyllisyyttä ja siihen käytettyjä resursseja.

```
1.  import pandas as pd
2.  import sqlalchemy
3.
4.  #Lue example.csv Pandas dataframeksi
5.  example_df = pd.read_csv("example.csv")
6.
7.  #Muodosta engine määrittely
8.  db_string = 'postgresql://esa:esimerkki@localhost:1234/mydatabase'
9.  engine = sqlalchemy.create_engine(db_string)
10.
11. #Tallenna dataframe SQL tauluun
12. example_df.to_sql('example', con=engine)
```

Ohjelma 1. CSV-tiedoston tallentaminen tietokantaan.

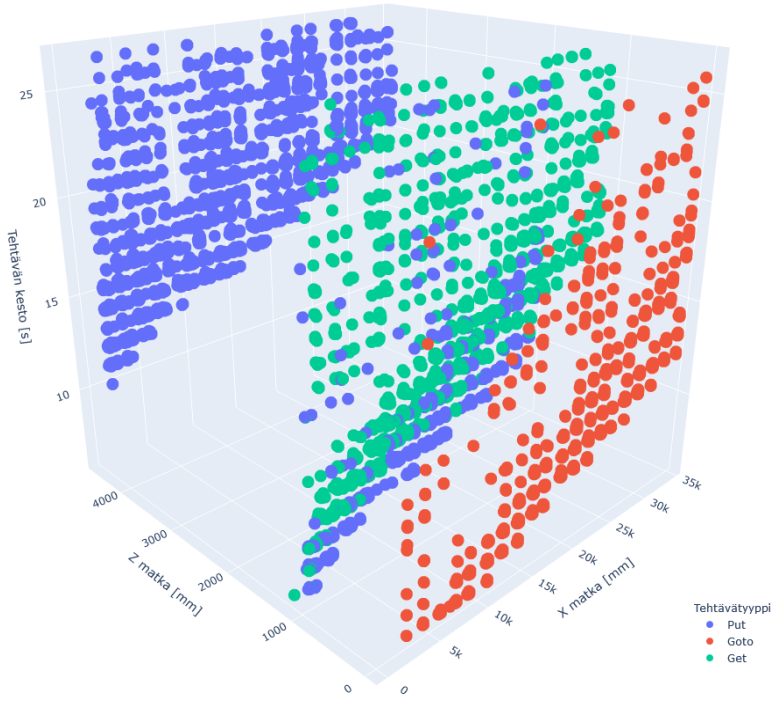
Tietokanta helpotti datan parissa työskentelyä vähentämällä tarvetta tiedostojen siirtelylle. Nyt analysointivaiheessa raakadata, raakadatasta aggregoidut arvot sekä analyysien tulokset oli mahdollista tallentaa yhtenäiseen ja helposti käsiteltävään muotoon. Lisäksi tietokantaan oli mahdollista asettaa loogisia rajoituksia. Tällainen rajoitus voisi olla esimerkiksi, että tietyllä laitteella, tietyllä ajanhetkellä voi olla vain yksi joukko tilatietoja. Rajoitusten etuna on, että kantaan ei ajeta samoja tietoja useampaan kertaa, mikä voisi sotkea analyysit. Myös lukuisat kehitysvaiheen ohjelmistovirheet jäivät kiinni näiden rajoitusten seurauksena.

4.4.2 Datan analysointi

Data-analyysin pääasiallisena tarkoituksena tässä projektissa oli löytää kerätystä datasta muutama matalalla roikkuva hedelmä. Näiden tarkoituksena on konkreettisesti osoittaa kerätystä datasta saatava hyöty. Kuten tämä projekti yleisesti, niin myös datan analysointi pyrittiin toteuttamaan jonkin kehyksen mukaisesti. Vaikka dataa lähdettiin vasta eksploratiivisesti tutkimaan, niin sovellettiin tässä vaiheessa löyhästi CRISP-DM –mallia (engl. CRoss-Industry Standard Process for Data Mining). CRISP-DM koostuu kuudesta päävaiheesta: projektin ymmärtäminen, datan ymmärtäminen, datan valmistelu, mallintaminen, arviointi ja julkaiseminen [23].

Ensimmäinen mielenkiinnon kohde Monorail-automaatiojärjestelmän toiminnassa olivat vaunujen tahtiajat. CRISP-DM mukaan ensimmäinen vaihe on selvittää projektin tavoitteet. Tässä käsiteltävän projektin tavoitteena on selvittää suorittavatko kaikki laitteet tehtävät yhtä nopeasti vai onko niiden välillä ei toivottuja eroja. Edellisessä alaluvussa kuvatun datakeräyksen avulla kerättiin dataa vaunujen suorittamista tehtävistä. Tehtävistä tallennettiin muun muassa tehtävien aloitus- ja lopetusajat sekä vaunun positio tehtävän alussa ja lopussa. Näiden avulla on mahdollista laskea tehtävään kulunut aika sekä teh-

tävässä ajettu x- ja z-matka. Yhteensä tarkasteltiin 125836 käyttöönoton aikaista tehtävää, joista 465:stä puuttui jokin arvo. Lisäksi jotkin tehtävät olivat keskeytyneet virheeseen, jonka seurauksena niiden kesto saattoi pahimmillaan venyä esimerkiksi viikonlopun yli. Poikkeamia ja muutamaa puuttuvaa arvoa lukuun ottamatta kerätty raakadata vaikutti laadukkaalta.



Kuva 10. Tehtävien kestot x- ja z-akselien suhteen tehtävätyypeittäin värikoodattuna.

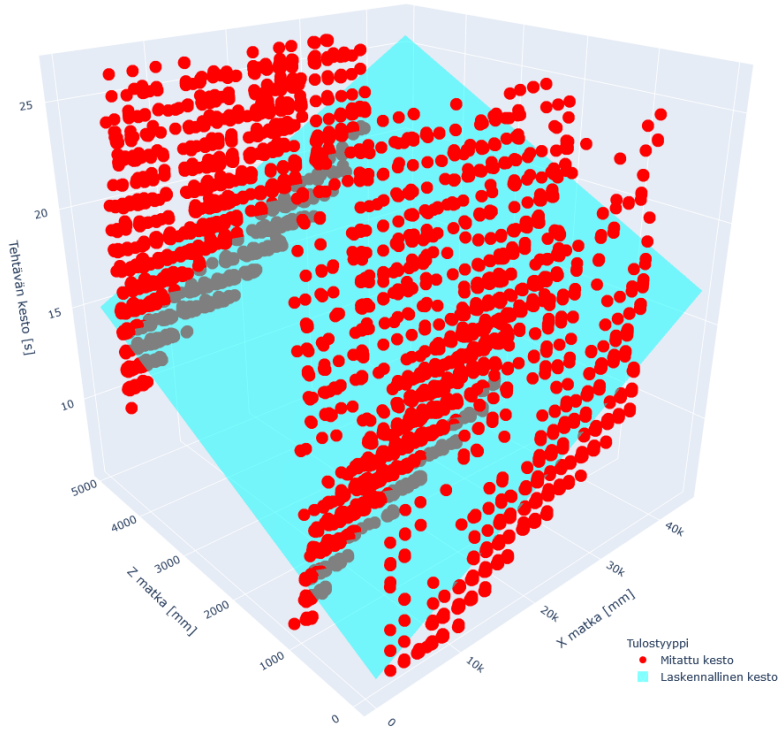
Datan ymmärtämiseen tehokas työkalu on visualisointi. Tässä diplomityössä dataa visualisoidaan pääasiassa Plotlyn graafisilla kirjastoilla käyttäen sen Python-rajapintaa. Kuva 10:ssä nähdään tehtävien kestot x- ja z-akselien suhteen. Kuvassa tehtävätyypit on värikoodattu. Esimerkiksi kuva 10:stä nähdään punaisella piirretyt Goto-tehtävät, joilla ei ole ollenkaan z-komponenttia. Lisäksi on nähtävissä, että tehtävillä on selkeä minimikesto, jota nopeammin niitä ei ole mahdollista suorittaa, mutta huomattava määrä tehtäviä kestää minimiaikaa pidempään.

Dataan tutustumisen jälkeen CRISP-DM ohjaa datan valmisteluun. Tätä analyysia varten datasta otettiin käyttöön vain onnistuneet tehtävät, joiden kestot olivat suurempia, kuin 5. persentiili ja pienempiä kuin 95. persentiili. Seuraava vaihe CRISP-DM:ssä on varsinainen mallintaminen. Tämän analyysin tavoitteena oli selvittää ovatko vaunujen tahtiajat samansuuruisia vai eivät. Tehtävien kestoa oli kuitenkin vaikea suoraan vertailla keskenään, sillä kestoon vaikuttaa muun muassa x- ja z-suuntiin ajettu matka. Tämän

vuoksi tahtiajoille päätettiin muodostaa yksinkertainen malli, jonka avulla niille olisi mahdollista laskea arvioitu tahtiaika. Tahtiajalle muodostettiin seuraava yhtälö:

$$t = ad_x + bd_z + c,$$

jossa t on tehtävän suoritusaika, a, b ja c ovat sovitettavia vakioita, d_x on absoluuttinen x-suuntaan kuljettava matka ja d_z on absoluuttinen z-suuntaan ajettu matka. Yhtälössä x- ja z-liikkeet on oletettu lineaarisiksi ja ne on laskettu yhteen. Tämän jälkeen yhtälön vakiot sovitettiin scipy-kirjaston curve_fit-funktiolla onnistuneista tehtävistä kerättyyn dataan. Sovitettu malli on visualisoitu kuva 11:sta. Kuva 11:sta on nähtävissä, että mallinnetut tulokset ovat lähellä tehtävien minimiaikaa, sillä suurin osa tehtävistä toteutui lähellä minimiaikaa. Kuvasta tätä ei ole yksiselitteistä nähdä, sillä tulokset piirtyvät suurilta osin toistensa päälle.



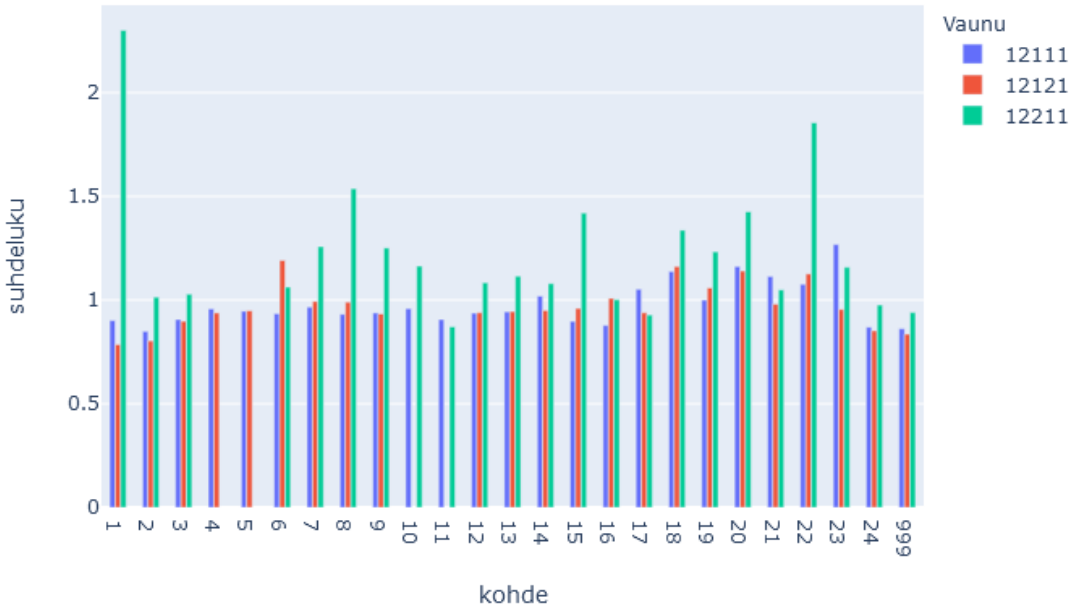
Kuva 11. Sovitettu malli tehtävien kestolle.

Sovitetun mallin avulla tehtäville on mahdollista saada laskennallinen tahtiaika. Mitatusta ja lasketusta ajasta on mahdollista laskea suhdeluku kuluneelle ajalle. Suhdeluvulla on mahdollista tutkia, suoriutuuko vaunu tehtävistään arvioitua nopeammin vai hitaammin, riippumatta tehtävissä ajetusta matkasta.

Mallintamisen jälkeen seuraava vaihe CRISP-DM:ssä on tulosten arviointi. Kun tahtiaikojen eroja lähdettiin selvittämään, huomattiin, että vaunut ajoivat eri nopeusohjeilla. Tämä operointipaneelilta asetettava nopeusohje (hidas, normaali tai täysi nopeus) ei ollut mu-

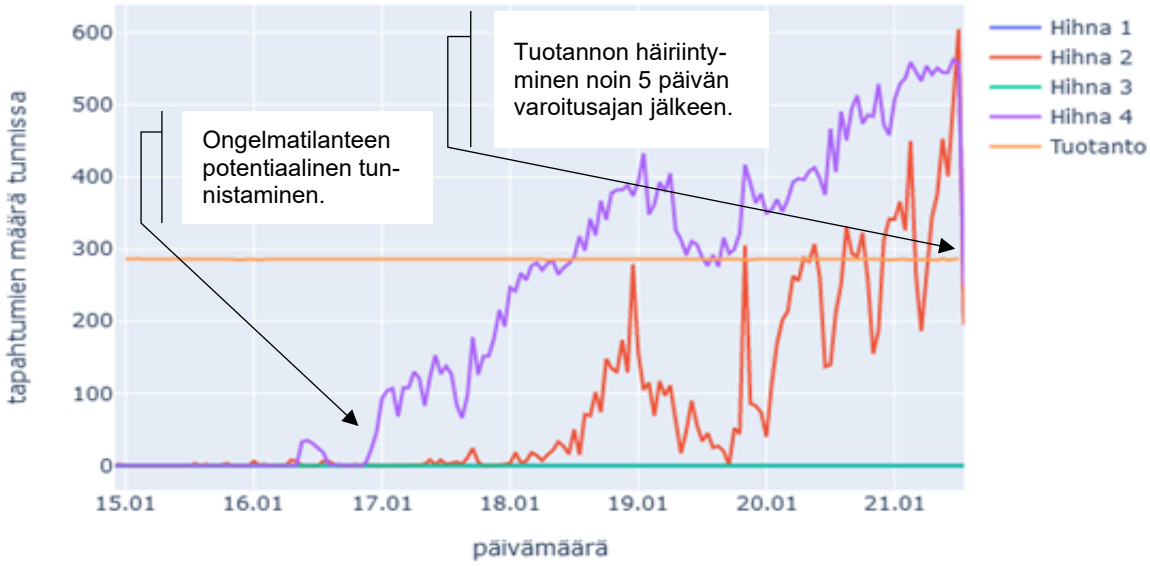
kana kerätyssä datassa, vaan se piti päätellä. Nopeusohjeen mukainen toiminta ei luonnollisesti ole virheellistä, joten nopeusohje tulee saada malliin mukaan. Tämä puolestaan edellyttää nopeusohjeen lisäämistä datankeräykseen. Tämän takia CRISP-DM:ssä lähdetään uudelle iterointikierrokselle, kunhan tarvittavaa dataa on saatu kerättyä. Alustavana tuloksena saatiin malli, jolla pystyttiin tunnistamaan eroja vaunujen tahtiajoissa.

Mallia soveltamalla on mahdollista tunnistaa poikkeamia erilaisissa tilanteissa. Esimerkiksi kuva 12:sta on nähtävissä vaunujen tehtäviin kuluneiden aikojen suhdetta mallin ennustamaan aikaan. Suhdeluvut on jaoteltu kohteittain eli ne vastaavat eri poiminta- ja laskupaikkoja. Kuva 12:sta perusteella on selkeästi nähtävissä, että vaunu 12211:n kohteeseen 1 kohdistuvat tehtävät kestävät reilusti yli kaksi kertaa enemmän kuin malli antaisi olettaa. Tämän kaltaisen analyysin avulla on mahdollista tunnistaa järjestelmän ongelmakohtia ja puuttua niihin mahdollisimman aikaisessa vaiheessa.



Kuva 12. Tehtäviin kuluneen ajan suhde malliin vaunuittain.

Kuva 13:sta on visualisoituna esimerkkitapaus mittaukseen perustuvasta ennakoivasta huollosta. Esimerkkitapauksessa testivaunu ajoi paikallaan toistuvia jättö- ja poimintatehtäviä. Kuva 13:sta on nähtävissä miten eri hihnan kireyttä mittaavista antureista tulevien virhesignaalien määrä kasvaa noin viiden päivän ajan, kunnes vaunu lopulta menee tuotannon pysäyttävään virheeseen. Ennakoivan analytiikan avulla tämä ja muut vastaavat tilanteet olisi mahdollista tunnistaa ennen virheen tapahtumista. Tällöin laitteelle olisi mahdollista suunnitella huolto ennen pysäyttävän virheen ilmaantumista. Näin suunniteltu tuotanto ei häiriintyisi eli laitteen kokonaistehokkuus parantuisi.



Kuva 13. Esimerkki mittaukseen perustuvan ennakoivan kunnossapidon mahdollistamasta datasta.

4.4.3 Datan visualisointi

Teollisuus 4.0:n suunnitteluperiaatteiden näkökulmasta datan visualisointi on tärkeää. Datan visualisointi toteuttaa varsinkin tiedon läpinäkyvyyden suunnitteluperiaatetta, mutta myös yhteenliitettävyyden suunnitteluperiaatetta. Yhteenliitettävyyden periaatetta noudatetaan, sillä tavoitteena on helpottaa laitteen ja käyttäjän yhteistyötä yhteisen tavoitteen suorittamiseksi. Datan, analyysien ja visualisointien toimittamiseksi sitä tarvitsevalle käyttäjälle päätettiin suunnitella alusta, jonka avulla informaatiota olisi modulaarisesti esitettävissä. Käytännössä järjestelmä tarjoaa neljä erilaista rajapintaa dataan: tietokannan, API:n (engl. Application Programming Interface), Jupyter Notebook -pohjat ja selainpohjaisen käyttöliittymän.

Tietokannan avulla osaava käyttäjä saa käytännössä rajattoman pääsyn raakadataan ja voi sen jälkeen hyödyntää sitä parhaaksi näkemällään tavalla. API-rajapinta mahdollistaa varsinkin selainpohjaisille sovelluksille laajasti tuetun tavan päästä dataan käsiksi. API:n avulla on lisäksi yksinkertaisempaa jakaa monimutkaisempaa tietoa, kuin tietokantarajapinnalla. Esimerkiksi kuva 10:n kaltainen interaktiivinen visualisointi on vaivatonta lähettää API:n avulla toiseen sovellukseen. Jupyter Notebook -pohjien avulla on mahdollista tarjota käyttäjälle työkalu kokeiluun ja prototyyppien nopeaan hahmotteluun. Jupyter Notebook -pohjien käyttäminen vaatii osaamista, mutta on kuitenkin selkeästi helpompaa, kuin täysin tyhjältä pohjalta aloittaminen. Valmiita pohjia soveltamalla käyttäjän on mahdollista nopeasti ratkaista yksilöllinen ongelmansa ilman toisen osapuolen

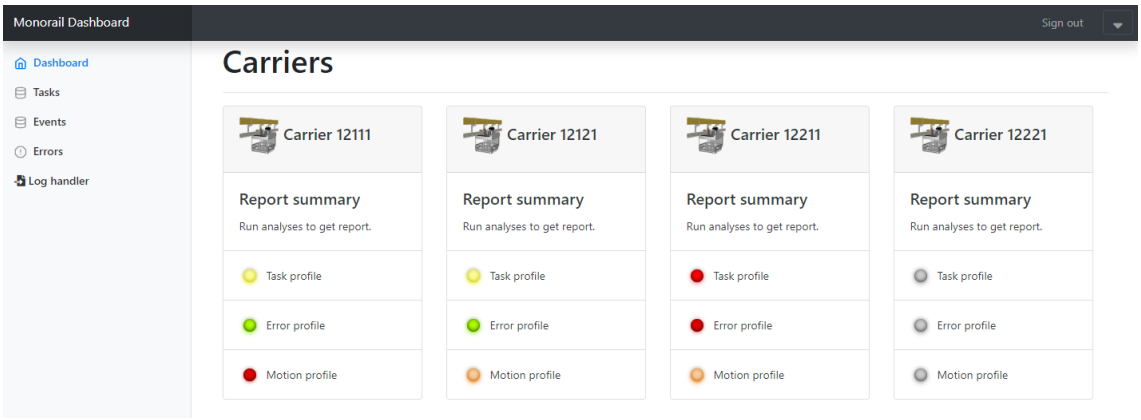
odottamista. Selainpohjainen käyttöliittymä on vaihtoehdoista käyttäjälle helpoin ja viimeistellyin, mutta tarjoaa vain valmiiksi valittuja visualisointeja.

Tätä projektia lähdettiin kehittämään vahvasti Jupyter Notebooksin päälle. Jupyter Notebooksin avulla myös visualisointien kehittäminen oli tehokasta, sillä lopputulos oli heti nähtävissä sekä helposti jaettavissa. Monet graafiset kirjastot kuten Matplotlib, Seaborn tai Plotly tukevat visualisointien piirtämistä suoraan Jupyter Notebooksin työkirjaan. Lisäksi esimerkiksi Plotlyn tuottamat visualisoinnit ovat interaktiivisia. Plotlyn visualisoinnit tukevat muun muassa zoomaamista, yksittäisten datapisteiden arvojen näyttämistä, datan suodattamista ja visualisoinnin tallentamista suoraan kuvaajasta. Esimerkiksi kuva 10:n kolmiulotteinen hajontakuvio on toteutettu Plotlyllä. Selaimessa tai Jupyter Notebookilla avattaessa kuvaajasta voidaan esimerkiksi valita vain tietyn tyyppiset tehtävät ja pyörittää kuvaaja kulmaan, josta kiinnostava arvo näkyy selkeämmin. Lisäksi Plotly on hyvin ekspressiivinen kirjasto, jonka vuoksi moni kuvaaja on mahdollista piirtää vain yhdellä rivillä koodia. Tarvittaessa kuvaajia pystyy myös muokkaamaan hyvin laajasti. Näiden ominaisuuksien vuoksi Plotly muodostui pääasialliseksi visualisointien piirtotyökaluksi.

Pelkkä visualisointien tuottaminen ei riitä, vaan ne pitää pystyä myös toimittamaan niitä tarvitsevalle käyttäjälle. Tätä tarkoitusta varten aloitettiin selainpohjaisen käyttöliittymän rakentaminen. Selainpohjaisen sovelluksen etuna on hyvin laaja sovellettavuus, sillä selainpohjaista materiaalia voi katsoa lähes kaikilla älylaitteilla tietokoneista puhelimiin. Sovelluksen nimeksi tuli Monorail Dashboard.

Monorail Dashboardin toteuttamisen tärkeimmät suunnitteluperiaatteet olivat luonnollisesti tiedon läpinäkyvyyden lisääminen ja käyttäjän avustaminen, mutta myös modulaarisuus. Modulaarisuuden tarkoituksena oli rakentaa Monorail Dashboard niin, että sitä olisi helppo inkrementaalisesti laajentaa. Tulevaisuudessa modulaarisen Monorail Dashboardin tulisi olla myös helposti räätälöitävissä kuhunkin käyttökohteeseen sopivaksi.

Dashboardin palvelinpuoli toteutettiin muun projektin tapaan pääasiassa Pythonilla. Ohjelmistokehyksenä käytettiin Flask-kirjastoa, joka on hyvin kevyt, mutta helposti laajennettava. Flask tarjoaa kaikki palvelimen prototyyppivaiheessa tarvitsemat ominaisuudet ilman erillistä määrittelyä. Kehitysvaiheen jälkeen Flask tarjoaa myös työkalut pilvipalvelimelle julkaisemiseen tai paikallisesti täysverisen tuotantopalvelimen luomiseksi. Flaskiin on tarjolla myös monia laajennuksia, kuten esimerkiksi Flask-SocketIO-kirjasto reaaliaikaiseen tiedonvälittämiseen ja Flask-RESTful REST-rajapinnan (engl. REpresentational State Transfer) toteutukseen.



Kuva 14. Monorail Dashboardin käyttöliittymä

Dashboardin palvelinpuoli hyödyntää pääasiassa Bootstrap-kehyksen pohjalta tehtyjä sivuja. Bootstrapin hyödyntäminen nopeutti prototyypin rakentamista huomattavasti ja mahdollisti sen toteuttamisen suhteellisen vähäisellä selainpuolen osaamisella. Dashboard rakentuu HTML:llä (engl. HyperText Markup Language) kuvatuille ja CSS-kielellä (engl. Cascading Style Sheet) muotoilluille pohjille. Pohjissa on sivun perusrakenteen lisäksi käytetty Handlebars-syntaksilla koodattuja sapluunoita. Sapluunojen avulla palvelimelta saatava data on mahdollista visualisoida. Esimerkiksi kuva 14:sta näkyvät, vaunujen tilaa kuvaavat, kortit on rakennettu Handlebars-sapluunalla.

Palvelin- ja selainohjelmisto kommunikoivat HTTP API:n (engl. HyperText Transfer Protocol, engl. Application Programming Interface) ja SocketIO:n avulla. HTTP API on pyritty suunnittelemaan REST-arkkitehtuurin mukaisesti, jolloin siitä voidaan käyttää nimeä REST API. REST-arkkitehtuuri valittiin, koska se sopii hyvin laitedatan ja siihen liittyvien analyysien esittämiseen. Lisäksi REST-arkkitehtuuri tarjoaa vakiintuneita suunnitteluperiaatteita, jolloin järjestelmään on helpompi liittyä. Tämä puolestaan noudattaa muun muassa Teollisuus 4.0:n yhteenliitettävyyden suunnitteluperiaatetta.

Käytetyistä teknologioista SocketIO sopii hyvin reaaliaikaisen tiedon, kuten esimerkiksi vaunun hetkellisen tilan lähettämiseen. REST API puolestaan tarjoaa hyvin laajan tuen käytetylle protokollalle. Esimerkiksi tiedon tallentaminen eri tason välimuisteihin on laajasti tuettua. Tietyissä tilanteissa tämä nopeuttaa selainpuolen käyttöliittymän toimintaa ja keventää palvelimen kuormaa. Prototyyppivaiheessa kaikki Dashboardin laitedataan liittyvä kommunikointi toteutettiin SocketIO:n avulla. Dashboardin jatkokehityksessä palvelimen ja selaimen välisen kommunikoinnin suunnitteluun kannattaa kiinnittää huomiota. Prototyyppivaiheessa tätä päätöstä ei vielä ollut mahdollista tehdä, koska esimerkiksi sisällön jakautumista reaaliaikaisen ja historiallisen datan välillä ei tiedetty.

Toinen huomionarvoinen asia selainpuolen ohjelmiston tulevaisuuden kehityksessä on sovelluskehyksen valinta. Prototyyppivaiheen selainohjelmisto pohjautuu enimmäkseen

vain Bootstrapin malleihin ja minimaaliseen JavaScriptiin. Sovelluskehyksen valinta tulee ajankohtaiseksi, jos tulevaisuudessa sovellusta tarvitsee huomattavasti laajentaa tai huomataan, että jokin sovelluskehys tarjoaa jonkin tarvittavan ominaisuuden.

Prototyyppivaiheen Dashboardin ulkoasu on nähtävissä kuva 14:sta. Dashboardin etusivulla kuvataan järjestelmän vaunut omilla korteillaan. Korttien on tiedonläpinäkyvyyden lisäämiseksi ja käyttäjän avustamiseksi koottu yhteenvedot eri analyysityyppien tuloksista. Tämänhetkiset analyysit on luokiteltu sen perusteella, liittyvätkö ne tehtäviin, virheisiin vai fyysisiin liikkeisiin. Modulaarisuuden suunnitteluperiaatetta noudattaen uusien luokkien luominen ja analyysien lisääminen onnistuvat ilman koodimuutoksia. Analyysien yhteenvetojen tulokset ovat nopeasti nähtävissä värikoodatuista indikaattoreista. Tämän avulla käyttäjä voi nopealla silmäyksellä nähdä järjestelmän tilan. Analyysien tarkemmat sisällöt on mahdollista nähdä yhteenvetojen otsikoihin upotettujen linkkien kautta.

Päänäkymän lisäksi Dashboard tarjoaa valikkorakenteen, joka näkyy kuva 14:sta vasemmassa reunassa. Valikkorakenteeseen voidaan helposti lisätä linkkejä uusiin osioihin. Tällä hetkellä pääsivun lisäksi Dashboardiin on toteutettu alasivut tehtäville, tapahtumille ja virheille. Näiltä sivuilta löytyy yksityiskohtaisempia analyysejä kunkin sivun aiheeseen liittyen.

4.5 Innovaation arviointi

Suunnittelutieteellisen tutkimuksen toinen päävaihe on innovaation arviointi [12]. Arvioinnin kannalta on huomattava, että tässä työssä luotiin periaatteessa täysin uusi sovellus, joka ei ole suoraan verrattavissa vanhaan toteutukseen. March ja Smith toteavat, ettei tällaisessa tilanteessa tarvita todellista suorituskyvyn arviointia [4]. Heidän mukaansa tutkimuksen tulos riippuu tällöin innovaation uutuudesta ja tehokkuuden perustelujen vakuuttavuudesta [4]. Tässä työssä haluttiin kuitenkin perehtyä myös arviointikriteereihin ja suunnitella kevyt kriteeristö, jonka avulla työtä ja sen kehitystä olisi mahdollista arvioida.

Monorail Dashboardin kohdalla tavoitteena on helpottaa automaatiojärjestelmän käyttämistä helpottamalla siihen liittyvien ongelmien ratkaisemista. Tässä yhteydessä käyttäminen ja ongelmat on hyvä ymmärtää mahdollisimman laajasti. Tällöin määritelmä kattaa mahdollisimman suuren osan ohjelmiston potentiaalisesta käytöstä, jolloin määritelmää ei tarvitse toistuvasti päivittää. Esimerkiksi niin käyttäjän avustaminen virhetilanteen selvittämisessä kuin myös tuotekehityksen seuraavan kehityskohteen valitseminen kuuluvat tässä saman määritelmän alle.

Seuraavissa alaluvuissa käsitellään Monorail Dashboardin realisaation arviointia yleisesti käyttäen Järvisten keräämiä mittareita. Tämän jälkeen arvioidaan realisaation komponenttien toteutusta. Lopuksi muodostetaan oma mittaristo Monorail Dashboardin kehityksen seuraamiseksi.

4.5.1 Realisaation arviointi

Tässä luvussa arvioidaan Monorail Dashboardin realisaatiota luvussa 3.2 listattujen kahdeksan mittarin avulla. Ensimmäiset Järvisten tunnistamista mittareista ovat tehokkuus ja vaikuttavuus. Tehokkuuden arvioimiseksi tulee tunnistaa realisaatioon sijoitetut panokset ja niillä saavutetut tulokset. Tarkkojen hyötyjen ja panosten mittaaminen on äärimmäisen vaikeaa. Koska järjestelmä on saatu toteutettua ja sitä on päästy hyödyntämään muutamassa todellisessa projektissa, on siihen sijoitetuista panoksista ja saaduista hyödyistä mahdollista muodostaa kohtuullinen arvio.

Monorail-automaatiojärjestelmän ja Monorail Dashboardin tapauksessa pystyttiin hyödyntämään avoimen lähdekoodin kirjastoja sekä olemassa olevaa infrastruktuuria, joten panokset syntyivät lähinnä järjestelmän kehittämisestä aiheutuneesta työstä. Pitkälle kehittyneet avoimen lähdekoodin kirjastot kuten esimerkiksi SQLAlchemy, Pandas, Plotly, Jupyter Notebooks ja Flask mahdollistivat nopean ohjelmistokehityksen. Ohjelmiston kehitys oli nopeaa, sillä datan keräämiseen, analysointiin ja esittämiseen tarvittavasta toiminnallisuudesta suurin osa oli saatavana valmiina. Tämän vuoksi myös järjestelmän kehittäminen onnistui melko pienellä panostuksella. Kokonaisuudessaan tarvittavat panokset jäivät siten hyvin maltillisiksi.

Järjestelmästä saatavat hyödyt ovat myös vaikeasti mitattavissa. Esimerkiksi jonkin virhetilanteen pysyvästä korjaamisesta saatavat kokonaishyödyt ovat vaikeasti arvioitavia. Virheen poistamisen suhteellinen osuus koneen kokonaistehokkuuteen on vaikea osoittaa. Lisäksi koneen operoimisen helppous ja virheen korjaamiseen aikaisemmin kulunut aika ovat vaikeasti arvotettavissa. Tästä huolimatta Monorail Dashboardista on ollut merkittävää apua sitä hyödyntäneissä testiprojekteissa. Datan avulla on ollut mahdollista täydentää operaattoreiden usein hyvinkin vajavaisia kuvauksia tapahtumista. Tämän vuoksi virhetilanteet ja niiden syntysyyt ovat selvinneet huomattavasti edeltäneitä projekteja tehokkaammin. Lisäksi datan ja sen visualisointien avulla monimutkaisetkin tilanteet ovat olleet helpommin kommunikoitavissa eri sidosryhmille. Karkeasti arvioiden pelkästään näiden testiprojektien aikana säästetyt asiakastuen työtunnit vähintäänkin kattavat prototyypin kehitykseen käytetyt tunnit.

Järjestelmän reaktiivinen käyttö on hyödyllistä, mutta sen suurin potentiaali on datan ennakoivassa hyödyntämisessä. Ennakoivan analytiikan avulla on mahdollista merkittävästi parantaa laitteiden kokonaistehokkuutta puuttumalla ongelmiin ennen kuin ne vaikuttavat tuotantoon. Kehitystyön aikana löydettiin potentiaalisia kohteita ennakoivan analytiikan kehittämiseen, mutta niiden toteuttaminen ei ollut mahdollista tämän diplomityön puitteissa.

Kolmas Järvisten tunnistama mittari on toteutuksen vaikutukset ympäristölle ja käyttäjille. Positiivisia ympäristönvaikutuksia voidaan saavuttaa toiminnan optimoinnilla. Optimoinnin avulla on mahdollista välttää turhaa työtä ja suorittaa tarpeellinen työ energiatehokkaammin. Varsinkin laitteen kulumista ja rikkoutumista estävillä toimilla on mahdollista saavuttaa suurtakin säästöä niin rahallisesti kuin ympäristön kannalta mitattuna.

Neljäs, viides ja kuudes mittari liittyvät sosiaalisiin, poliittisiin ja historiallisiin vaikutuksiin. Monorail Dashboardin aiheuttamat sosiaaliset, poliittiset tai historialliset vaikutukset tuskin ovat merkityksellisiä. Monorail Dashboard ei ole uusi tai mullistava sovellus, kuten esimerkiksi sosiaalisen median sovellukset aikanaan olivat. Onnistuessaan Monorail Dashboard tuo inkrementaalisen parannuksen perinteiseen automaatiojärjestelmään. Helpottunut tiedonsaanti saattaa aiheuttaa organisaation toimintaan muutoksia. Esimerkiksi perinteinen laitteiden määräaikaishuolto saattaa muuttua dataan pohjautuvaksi ennakoivaksi huolloksi. Tällaiset ja muut datan mahdollistamat muutokset näkyvät myös organisaatiotasolla vähintäänkin työtapojen muutoksena. Onnistuessaan Monorail Dashboardin toimintaa tullaan laajentamaan mahdollisuuksien mukaan kattamaan koko Cimcorp Oy:n toimitus, jolloin vaikutukset koskettavat koko tehdasta. Tällöin on mahdollista kehittää edistyneempiä analyysejä, joilla puolestaan on suurempi vaikutus toimintaan. Tällaisessa tilanteessa luonnollisesti myös vaikutukset olisivat laajempia ja merkittävämpiä.

Seitsemäs tunnistettu mittari on investointien arviointi. Sovellusta arvioitaessa sen taloudellinen kannattavuus on tärkeä elementti. Sovelluksen käyttäjä luultavasti ilahtuu tehtäviä helpottavasta työkalusta, mutta hankintapäätökset tehdään pääasiassa taloudellisin perustein. Monorail Dashboardin ongelma on suhteellisen pitkä arvoketju datasta varsinaiseen lisäarvoon. Lisäarvon saavuttamiseksi koko datan keräyksen, analysoinnin ja esittämisen ketjun on toimittava. Lisäksi käyttäjien ja organisaation pitää myös osata hyödyntää järjestelmän uutta kyvykkyyttä. Toisaalta suhteellisten arvokkaiden tuotantokoneiden hyödyntämisen pienikin parannus tuo merkittävää lisäarvoa ohjelmiston käytöstä syntyviin kuluihin nähden. Ohjelmiston tuottaman lisäarvon tarkemman määrittämisen ongelmaksi muodostuvat laskentatoimen perusongelmat: laajuusongelma, mittaamisongelma, arvostusongelma ja jakamisongelma [24].

Vanhoissa järjestelmissä tätä voidaan osittain helpottaa vertaamalla toimintaa vanhaan. Tässä ongelmaksi muodostuu datan puute, sillä laajamittaisempi kerääminen alkaa vasta, kun Monorail Dashboard otetaan käyttöön. Järjestelmissä, joissa Monorail Dashboard on ollut mukana alusta alkaen taas vertailukohtaa ei ole. Laskentatoimen perusongelmat olisi hyvä tiedostaa ohjelmaa suunniteltaessa siten, että dataperusteisesti olisi mahdollista perustella saavutettuja hyötyjä. Esimerkiksi kuva 13:sta nähtävässä analyysissä on esitetty sekä ongelman tunnistamisen potentiaalinen ajankohta että sen todellinen ilmaantumishetki. Järjestelmän tulisi tukea myös tämän kaltaisten tietojen keräämistä sen hyödyllisyyden osoittamiseksi.

Kahdeksas mittari on toteutuksen huolto- ja kokonaiskustannukset. Huolto- ja kokonaiskustannuksia on vaikea arvioida, sillä toteutuksen testikäyttö on vasta aloitettu. Huomattavaa on, että ohjelmistotuotteiden ja varsinkin analytiikkasovellusten huoltokustannukset saattavat olla merkittävän suuret. Huoltokustannuksiin vaikuttaa vahvasti, kuinka hyvin Monorail Dashboardiin pystytään toteuttamaan itsepalvelumallia, jolloin käyttäjät voisivat itse kantaa osan mukauttavan huollon kustannuksista.

4.5.2 Komponenttien toteutuksen arviointi

Datan kerääminen toteutettiin Monorail-automaatiojärjestelmän jo valmiiksi hyödyntämiä rajapintoja laajentamalla. Tämän toteutuksen etuna on, että se onnistui hyvin pienellä työmäärällä ja toiminnallisuus on pääosiltaan mukana niin uusissa kuin vanhoissa Cimcorp Oy:n projekteissa. Toteutuksen haittapuolena on, että dataa keräävä komponentti on riippuvainen tämän vuoksi WCS:n toiminnasta, jolloin siihen kohdistuvat muutokset pitää koordinoida WCS:n kehityksen kanssa. Toisena vaihtoehtona pohdittiin OPC UA:ta, jolloin komponentti olisi voinut toimia täysin itsenäisesti. Tämän toteutuksen haittapuolena oli suurempi työmäärä ja toteutuksen vaatima erillinen käyttöönotto kaikkiin tarvittaviin projekteihin.

Analytiikan kehittämiseen käytettiin Jupyter Notebook -alustaa sekä Pythonin hyvin pitkälle kehittynyttä datatiede-kirjastojen ekosysteemiä (muun muassa Numpy, Scipy, Pandas). Näiden kirjastojen avulla dataa oli mahdollista lukea ja käsitellä hyvin helposti. Esimerkiksi päivä tai tuntikohtaisten koosteiden ja pivot-taulukkojen tekeminen onnistuu muutamalla komennolla. Myös mallintamisen ja koneoppimisen teknologiat olivat tarjolla.

Visualisoinnin toteuttamisessa hyödynnettiin myös Pythonin tarjoamia avoimen lähdekoodin kirjastoja. Kuvaajien piirtämisessä käytettiin lähinnä Plotlyä. Plotlyn etuna ovat sen äärimmäisen tehokas rajapinta, toteutus useille ohjelmointikielille ja mahdollisuus tallentaa kuvaajia monissa eri formaateissa. Plotly tukee lukemattomia eri kuvaajatyyp-

pejä ja niiden piirtäminen onnistuu useimmiten yhdellä koodirivillä. Koska Plotly on toteutettu myös JavaScriptillä, oli Pythonillä kehitetyt kuvaajat mahdollista piirtää natiivisti myös selaimessa välittämällä pelkästään data ja kuvaajan määrittely JSON-formaatissa. Varsinainen käyttöliittymä toteutettiin web-pohjaisena. Sen pohjana toimi Pythonin Flask-kirjasto, jonka avulla web palvelin oli helppo toteuttaa. Selainohjelmisto toteutettiin hyödyntämällä Bootstrap-kirjastoa ja natiivia JavaScriptiä.

Näitä tekniikoita hyödyntämällä saatiin aikaan toimiva prototyyppi. Erinomaisten kirjastojen avulla kehittäminen onnistui nopeasti ja kustannustehokkaasti. Prototyypin kehitystyön valmistuttua teknologiavalinnat osoittautuivat toimiviksi. Projektin toteuttamisesta saadun kokemuksenkaan myötä ei ole tullut vastaan teknologiaa, jolla olisi haluttu korvata alkuvaiheessa valittu teknologia. Ongelmana Pythonin valtavan ekosysteemin hyödyntämisessä oli mahdollisten vaihtoehtojen kartoittamiseen ja niihin tutustumiseen vaadittu aika. Avoimen lähdekoodin ja valtavan käyttäjämäärän vuoksi tarjonnasta löytyi paljon kirjallisuutta, jonka avulla tarjonnan läpikäyminen onnistui tehokkaasti ilman salapoliisityötä.

4.5.3 Mittariston kehittäminen

Realisaatioiden mittarien teoriapohjan perusteella (luku 3.2) laadittiin Monorail Dashboardille oma mittaristo (liite A). Mittariston avulla Monorail Dashboardia on mahdollista verrata innovaatiota edeltäneeseen tilanteeseen, sen tuleviin iteraatioihin tai muihin sitä vastaaviin ratkaisuihin. Mittariston kehyksenä käytettiin GQM-lähestymistapaa (engl. Goal Question Metric, tavoite-kysymys-metriikka). GQM-lähestymistavassa mittarit muodostetaan määrittämällä käsitetason tavoite, siihen liittyvät operationaalisen tason kysymykset ja lopulta näihin liittyvät määrälliset mittarit [25]. Mittaristo laadittiin kahden päätavoitteen ympärille. Ensimmäinen tavoite on ”Lisätä tiedon läpinäkyvyyttä Monorail-automaatiojärjestelmän käyttäjille.” ja toinen tavoite on ”Tarjota teknistä avustusta Monorail-automaatiojärjestelmän käyttäjille.” (liite A). Tavoitteet perustuvat kahteen tärkeimmäksi arvioituun Teollisuus 4.0:n suunnitteluperiaatteeseen eli tiedon läpinäkyvyyteen ja tekniseen avustamiseen (taulukko 2).

Mittariston kyvykkyyttä on vaikea arvioida *ex ante*. Mittariston kyvykkyys selviää käytön aikana, kun sen hyödyt ja puutteet voidaan havaita käytännössä. Tämän vuoksi mittariston tuloksiin tulee suhtautua järkevällä varauksella ja peilata sen tuloksia käytännön tilanteeseen. Nyt kehitettyä mittaristoa ei kannata pitää lopullisena versiona, vaan sitä tulee kehittää sen käytöstä saatujen oppien mukaisesti. Esimerkiksi yhteenliitettävyyden suunnitteluperiaatetta (taulukko 2) ei ole tässä versiossa huomioitu, mutta tulevaisuu-

dessa liittyminen toisiin järjestelmiin saattaa olla ensiarvoisen tärkeää. Yhteenliitettävyyden merkitys korostuu esimerkiksi tilanteessa, jossa useilla asiakkailla on jo olemassa oma IoT-ympäristönsä, jonne Monorail-automaatiojärjestelmän data halutaan integroida. Tällöin toimivan mittariston tulisi huomioida nämä uudet vaatimukset niiden edellyttämällä tavalla.

Innovaation arvioinnin toinen vaihe on varsinainen arviointi kehitettyjen mittarien avulla [4]. Tämän diplomityön tuloksena kehitetyn prototyypin arviointi on esitetty liite B:ssä. Kaikkia liite A:ssa määriteltyjä mittareita ei ollut mahdollista käyttää, sillä tarvittavia tietoja ei osattu kerätä kehitystyön alkuvaiheessa. Toisia mittareita on ollut mahdollista käyttää vain osittain. Esimerkiksi suunnitelluista käyttäjäryhmistä prototyyppiä on käytetty vain tuotekehityksessä, käyttöönotossa ja asiakastuessa. Puutteista huolimatta mittarien avulla on mahdollista muodostaa kuva ohjelmiston nykytilasta ja varsinkin verrata sitä tuleviin iteraatioihin.

5. JOHTOPÄÄTÖKSET

Tässä diplomityössä toteutettiin suunnittelutieteellisesti ohjelmiston realisaatio. Monorail Dashboard -työnimellä tehdyn ohjelmiston tarkoituksena on testata Monorail-automaatiojärjestelmän ongelmanratkaisun helpottamista datan keräämisen, analysoinnin ja visualisoinnin avulla. Ohjelmiston suunnittelun pohjana käytettiin Teollisuus 4.0:n tarjoamia teknologioita ja suunnitteluperiaatteita.

Työn tuloksena saatiin aikaan uusi ohjelmisto, jonka avulla on mahdollista kerätä, analysoida ja esittää Monorail-automaatiojärjestelmään liittyvää dataa. Marchin ja Smithin mukaan suunnittelututkimuksen peruskysymys on selvittää, toimiiko artefakti [4]. Koska järjestelmä toimii ja sitä on pystytty hyödyntämään jo muutamassa toimitusprojektissa, on tutkimuksessa onnistuttu selvittämään tämä peruskysymys.

Varsinaisia tutkimuskysymyksiä on kolme. Näistä ensimmäinen ja tärkein on: ”Onko Monorail-automaatiojärjestelmän datan hyödyntäminen kannattavaa?” Datan hyödyntäminen voidaan tässä työssä katsoa kannattavaksi, jos siitä saatavat hyödyt ylittävät niiden saamiseksi käytetyt panokset. Tarkkojen hyötyjen ja panosten mittaaminen on äärimmäisen vaikeaa. Koska järjestelmä on saatu toteutettua ja sitä on päästy hyödyntämään muutamassa todellisessa projektissa, on siihen sijoitetuista panoksista ja saaduista hyödyistä mahdollista muodostaa kohtuullinen arvio.

Aikaisemmin dataa on hyödynnetty suurissa tuotantolaitoksissa, joissa prosessin vähäinenkin tehostaminen on saattanut tuoda miljoonaluokan tuloksia. Teollisuus 4.0:n tuoma kehitys on laskenut huomattavasti datan käsittelyyn tarvittavaa panosta. Tämän vuoksi dataa on kannattavaa hyödyntää yhä pienemmissä järjestelmissä.

Luvun 4.5.1 arvioinnin perusteella Monorail Dashboardin kehitys oli Teollisuus 4.0:n vuoksi kustannustehokasta. Lisäksi valmiita kirjastoja hyödyntämällä oli mahdollista saada aikaan kehittynyttä toiminnallisuutta, josta oli konkreettista hyötyä järjestelmän käytössä.

Vastaus ensimmäiseen tutkimuskysymykseen ”Onko Monorail-automaatiojärjestelmän datan hyödyntäminen kannattavaa?” on kyllä. Datan hyödyntäminen on kannattavaa, sillä jo prototyyppivaiheessa hyödyt ylittivät sijoitetut panokset. Lisäksi järjestelmän rakentaminen on luonut pohjan kehittyneemmän analytiikan kehittämiseksi, jonka avulla on mahdollista saavuttaa merkittäviä hyötyjä.

Toinen tutkimuskysymys on: ”Mitkä ovat tärkeimmät komponentit Monorail-automaatiojärjestelmän dataa hyödyntävässä järjestelmässä?” Monorail Dashboardia kehitettäessä tärkeimmiksi komponenteiksi muodostuivat dataa keräävä, analysoiva ja esittävä komponentti. Nämä komponentit toistuvat diplomityön rakenteessa, sillä ne tunnistettiin hyvin nopeasti työn alettua. Jokainen näistä komponenteista täyttää käyttäjän avustamisessa välttämättömän toiminnallisuuden. Komponentit toimivat itsenäisesti ja ne voidaan toteuttaa toisistaan poikkeavilla tekniikoilla. Komponentit voidaan järjestää datan virtauksen mukaan dataa keräävään, analysoivaan ja esittävään komponenttiin. Näistä aina edellistä tarvitaan seuraavan toimintaan ja varsinainen lisäarvo saadaan vasta, kun data on kulkenut koko ketjun läpi keräyksestä käyttäjälle. Komponenttien perustoiminnan valmistuttua, niitä oli mahdollista kehittää inkrementaalisesti ja toisistaan riippumattomasti.

Dataa keräävän ja esittävän komponentin toteuttaminen on melko suoraviivainen tekninen ongelma. Niiden toteuttaminen on selkeää, kunhan tiedetään, mitä dataa halutaan kerätä ja esittää. Datan analysointi puolestaan tarjoaa lähes loppumattomat mahdollisuudet hyödyllisen tiedon löytämiseen kerätystä datasta. Teollisuus 4.0:n myötä varsinkin ennakoivan analytiikan rooli on korostunut. Monorail Dashboardissa ehdittiin vain raapaisemaan pintaa analytiikan potentiaalista. Analytiikan kehittäminen on selkeä kohde jatkotutkimukselle, sillä sen avulla on mahdollista tuottaa suurta lisäarvoa.

Kolmas tutkimuskysymys oli: ”Miten nämä komponentit kannattaisi toteuttaa?” Tämä kysymys on täysin riippuvainen toteuttavasta järjestelmästä. Monorail Dashboardin kohdalla muutama rajoite määritti valitut kehitystavat. Tärkeimpänä määrittävänä ehtona olivat kustannustekijät. Jotta Monorail Dashboard olisi taloudellisesti kannattava, sen tulisi olla kustannuksiltaan maltillinen, sillä pienen automaatiojärjestelmän optimoinnilla ei ole saavutettavissa suurta lisäarvoa. Myös asiakkaiden haluttomuus avata automaatioverkosta yhteyttä pilveen sekä pilvipalveluiden aiheuttamat tietoturvakysymykset rajoittivat pilvipalveluiden hyödyntämistä prototyyppivaiheessa. Kuten luvussa 4.5.2 todetaan, näiden määrittelyjen vuoksi kehitys tapahtui vahvasti avoimia ohjelmistoja ja valmista infrastruktuuria hyödyntäen.

Kokonaisuutena arvioiden työ saavutti sille asetetut tavoitteet. Sen tuloksena saatiin kehitettyä toimiva prototyyppi, jonka avulla saatiin vastaukset tutkimuskysymyksiin. Prototyypin toteuttaminen helpottaa tulevaisuudessa myös muiden tuotteiden datan hyödyntämisen hyödyllisyyden arviointia ja toteutuksen suunnittelua. Lisäksi Monorail-automaatiojärjestelmän kannalta prototyyppi tarjoaa hyvän lähtökohdan kehityksen jatkamiselle.

Jatkokehityksen kannalta kriittinen asia on datan keräämisen automatisointi mahdollisimman pitkälle. Tuotantokäytössä datan keräyksen manuaaliset vaiheet eivät ole enää

järkeviä, vaikka ne prototyypissä vielä ovat hyväksyttäviä. Esimerkiksi hälytysten ja ennakoivien ilmoitusten saamiseksi järjestelmällä pitää olla pääsy ajantasaiseen dataan. Lisäarvon kannalta analytiikan kyvykkyyden kehittäminen on tärkeää. Itsepalvelumallin kehittäminen on myös tutkimisen arvoista, sillä sen avulla käyttäjien olisi mahdollista nopeammin ja yksilöidymmin hyödyntää järjestelmää. Lisäksi itsepalvelumallin avulla voisi olla mahdollista saavuttaa huomattavia kustannussäästöjä. Säästöjä olisi mahdollista saada, jos käyttäjät pystyisivät itse toteuttamaan osan uusista vaatimuksista, jolloin toiminnallisuutta lisäävän huollon määrä laskisi.

Tärkeimpänä kehityskohteena on data-analyysin kehittäminen. Prototyypissä toteutettiin vain muutaman tapauksen data-analyysi järjestelmän potentiaalin esittämiseksi. Tuotantokäytössä olevan järjestelmän arvo määräytyy pääasiallisesti sen analyysien kyvykkyyksien mukaan, joita datan kerääminen ja esittäminen tukevat.

LÄHTEET

[1] Kagermann , Helbig J, Hellinger , Wahlster W. Recommendations for implementing the strategic initiative INDUSTRIE 4.0: Final report of the Industrie 4.0 Working Group. Frankfurt:; 2013.

[2] Lasi H, Fettke P, Kemper HG, Feld T, Hoffmann M. Industry 4.0. Business & Information Systems Engineering. 2014 Elokuu; 6(4): 239-242.

[3] Sommerville I. Software engineering. 9th ed. Boston: Pearson; 2011.

[4] March S, Smith G. Design and natural science research on information technology. Decision Support Systems. 1995; 15(4): 251-266.

[5] Aken JEV. Management Research Based on the Paradigm of the Design Sciences: The Quest for Field-Tested and Grounded Technological Rules. Journal of Management Studies. 2004 March; 41(2): 219-246.

[6] Thuan N, Drechsler A, Antunes P. Construction of Design Science Research Questions. Communications of the Association for Information Systems. 2019 Jan; 44(1): 332-363.

[7] Tampereen yliopisto. Opinnäytetyön kirjoitusohje tekniikan alalla: Tekniikan kandidaatintyön ja diplomityön kirjoittaminen. Tampereen yliopisto, Tekniikan alojen tutkinto-ohjelmat; 2019.

[8] Culot G, Nassimbeni G, Orzes G, Sartor M. Behind the definition of Industry 4.0: Analysis and open questions. In: International Journal of Production Economics; 2020 Tammikuu. p. 107617.

[9] MADE IN CHINA 2025. [Online]. [cited 2020 Heinäkuu 3. Available from: http://english.www.gov.cn/2016special/madeinchina2025/.

[10] Ghobakhloo M. The future of manufacturing industry: a strategic roadmap toward Industry 4.0. Journal of Manufacturing Technology Management. 2018 Oct; 29(6): 910-936.

[11] Hermann M, Pentek T, Boris O. Design Principles for Industrie 4.0 Scenarios. In: 2016 49th Hawaii International Conference on System Sciences (HICSS); 2016. p. 3928-3937.

[12] Järvinen P, Järvinen A. Tutkimustyön metodeista. [uud. p.] ed. Tampere: Opinpajan kirja; 2011.

[13] Hevner A, March S, Park J, Ram S. Design Science in Information Systems Research. MIS Quarterly. 2004 March; 28(1): 75 - 105.

[14] Poppendieck M, Poppendieck T. Lean Software Development: An Agile Toolkit: Addison Wesley Professional; 2003.

[15] Orlikowski W. Evolving with Notes : organizational change around groupware technology. IDEAS Working Paper Series from RePEc [Internet]. 1995.

[16] Kling R. Defining the boundaries of computing across complex organizations. In Boland RJ. Critical issues in information systems research. New York: John Wiley & Sons, Inc.; 1987. p. 307 - 362.

[17] Dahlbom B, Mathiassen L. The future of our profession. Communications of the ACM. 1997 Jun; 40(6): 80-89.

[18] Lano K, Haughton H. Software maintenance research and applications. In: NordData; 1992; Tampere. p. 123-143.

[19] Schlesinger P, Rahman N. Self-Service Business Intelligence Resulting in Disruptive Technology. The Journal of computer information systems. 2015; 56(1): 11-21.

[20] Metalliteollisuuden Standardisointiyhdistys ry. Koneturvallisuus. Turvallisuuteen liittyvät ohjausjärjestelmien osat. Osa 1: Yleiset suunnitteluperiaatteet Helsinki: Suomen Standardoimisliitto SFS ry; 2015.

[21] Maretzke. When to use waterfall, when agile? [Online].; 2019 [cited 2020 7 23. Available from: https://www.agile-minds.com/when-to-use-waterfall-when-agile/.

[22] Appelo J. Simple vs. Complicated vs. Complex vs. Chaotic. [Online].; 2008 [cited 2020 7 23. Available from: https://noop.nl/2008/08/simple-vs-complicated-vs-complex-vs-chaotic.html.

[23] Berthold M. Guide to intelligent data analysis how to intelligently make sense of real data Lontoo: Springer; 2010.

[24] Virkkunen H. Teollisuuden kertakustannukset : niiden degressio sekä käsittely kustannuslaskennassa. Helsinki: Liiketaloustieteellinen tutkimuslaitos; 1951.

[25] Basili V, Caldiera G, Rombach HD. THE GOAL QUESTION METRIC APPROACH. Encyclopedia of software engineering. 1994;: 528-532.

[26] Lueth KL. Will the industrial internet disrupt the smart factory of the future? [Online].; 2015 [cited 2020 Kesäkuu 7. Available from: https://iot-analytics.com/industrial-internet-disrupt-smart-factory/.

LIITE A: MONORAIL DASHBOARDIN ARVIOINNIN MITTARIT

Tässä liitteessä on esitetty Monorail Dashboardin arviointiin käytettävät mittarit. Mittarit on esitetty GQM-lähestymistavan mukaisessa formaatissa.

Tavoite 1 – Lisätä tiedon läpinäkyvyyttä Monorail-automaatiojärjestelmän käyttäjille.

Tavoite 1	Tarkoitus	Lisätä
	Ongelma	tiedon läpinäkyvyyttä
	Kohde	Monorail-automaatiojärjestelmän
	Näkökulma	käyttäjille.
Kysymys	Q1.1	Onko käyttäjän tarvitsema tieto järjestelmässä?
Mittarit	M1.1.1	Kerättyjen datapisteiden määrä.
	M1.1.2	Kerättyjen datapisteiden suhde kaikkiin käyttäjien data keräyksen vaatimuksiin.
Kysymys	Q1.2	Löytääkö käyttäjä tarvitsemansa tiedon järjestelmästä?
Mittarit	M1.2.1	Visualisoitujen datapisteiden määrä.
	M1.2.2	Esitettyjen datapisteiden suhde käyttäjien tarvitsemiin datapisteisiin.
	M1.2.3	Käyttäjien kokemus tiedon löytämisen helppoudesta.
Kysymys	Q1.3	Toimiiko tiedon kerääminen tehokkaasti?
Mittarit	M1.3.1	Tiedonkeräämiseen käytetty kapasiteetti (mm. levytila, laskentateho)
	M1.3.2	Tiedonkeräämisen ylläpitoon käytetty kapasiteetti (mm. työtunnit, pilvipalveluiden maksut)
Kysymys	Q1.4	Onnistuuko puuttuvan tiedon lisääminen keräykseen?

Mittarit	M1.4.1	Aika tarpeen tunnistamisesta datapisteen lisäämiseen keräykseen.
	M1.4.2	Lisättyjen datapisteiden suhde tunnistettuihin datapisteisiin.
	M1.4.3	Kehittäjän mielipide järjestelmän ketteryydestä (1-5).
Kysymys	Q1.5	Helpottaako informaation saatavuus työtä?
Mittarit	M1.5.1	Sidosryhmittäin koottu mielipide esitetyn informaation hyödyistä.

Tavoite 2 – Tarjota teknistä avustusta Monorail-automaatiojärjestelmän käyttäjille.

Tavoite 2	Tarkoitus	Tarjota
	Ongelma	teknistä avustusta
	Kohde	Monorail-automaatiojärjestelmän
	Näkökulma	käyttäjille.
Kysymys	Q2.1	Onko käyttäjän ongelmaan olemassa analyysi?
Mittarit	M2.1.1	Analyysien lukumäärä
	M2.1.2	Analyysien lukumäärä / tunnistettujen analyysitarpeiden lukumäärä
	M2.1.3	Analyysien kyvykkyys
Kysymys	Q2.2	Toimivatko analyysit tehokkaasti?
Mittarit	M2.2.1	Analyysien kehittämiseen käytetyt resurssit (käytetty suoritusteho, pilvipalveluiden kustannukset ja muut vastaavat)
	M2.2.2	Analyysien tuottamiseen tarvittavat resurssit (käytetty suoritusteho, pilvipalveluiden kustannukset ja muut vastaavat)
Kysymys	Q2.3	Helpottavatko analyysit käyttäjien työtä?
Mittarit	M2.3.1	Sidosryhmittäin koottu mielipide analyysien hyödyistä.

LIITE B: MONORAIL DASHBOARDIN POC VERSION ARVIOINTI

Tässä liitteessä arvioidaan diplomityönä valmistunut POC-tason (engl. proof of concept) toteutus Monorail Dashboardista liite A:n mittaristoa hyödyntäen. Mittaristoa tulisi kehittää suuntaan, jossa siitä olisi mahdollista saada numeerisia tunnuslukuja. Tässä vaiheessa koettiin kuitenkin hyödyllisemmäksi täyttää mittaristoa laadullisesti, jotta arviointiin tarvittava informaatio välittyisi tehokkaammin.

Tavoite: Lisätä tiedon läpinäkyvyyttä Monorail-automaatiojärjestelmän käyttäjille.

Q1.1 Onko käyttäjän tarvitsema tieto järjestelmässä?

M1.1.1 Kerättyjen datapisteiden määrä

33 tag-arvoa + 23 tapahtumaan jokaista vaunua kohden.

Esimerkiksi yhteen tag-arvoon on kerätty 24 binäärisen anturin tiedot. Toisena esimerkkinä yhdessä tapahtumassa välitetään kaikkien neljän moottorin lämpötilat.

M1.1.2 Kerättyjen datapisteiden suhde kaikkiin käyttäjien data keräyksen vaatimuksiin.

100 %, mutta käytössä vasta hyvin rajalliset käyttäjävaatimukset.

Q1.2 Löytääkö käyttäjä tarvitsemansa tiedon järjestelmästä?

M1.2.1 Visualisoitujen datapisteiden määrä

Tapahtumat, tehtävät ja virheet ovat visualisoituina.

M1.2.2 Esitettyjen datapisteiden suhde kaikkiin käyttäjien tarpeisiin.

100 %, mutta käytössä vasta hyvin rajalliset käyttäjävaatimukset.

M1.2.3 Käyttäjien kokemus tiedon löytämisen helppoudesta.

Perusasiat nähtävissä ja visualisoituina.

Q1.3 Toimiiko tiedon kerääminen tehokkaasti

M1.3.1 Tiedonkeräämiseen käytetty kapasiteetti (mm. levytila, laskentateho).

Noin 0.5 GB/vaunu/kuukausi tiedon tallentamiseen.

M1.3.2 Tiedonkeräämisen ylläpitoon käytetty kapasiteetti (mm. työtunnit, pilvipalveluiden maksut)

Tiedostot on kerätty manuaalisesti eri kohteista. Työaikaa kuluu noin 5~15 minuuttia/vaunu/viikko, kun vaunujen määrä on vähäinen. Kannan hyödyntämisessä on huomattavasti varaa optimointiin, mutta sitä tuskin kannattaa tehdä POC vaiheessa.

Q1.4 Onnistuuko puuttuvan tiedon lisääminen keräykseen?

Suhteellisen helposti. PLC koodi ja tag-määrittelyt ovat pahimmat pullonkaulat, varsinkin vanhoissa järjestelmissä. Muutamia tapahtumia lisättiin POC vaiheessa olemassa olevan järjestelmän datankeräykseen onnistuneesti. POC vaiheessa tunnistettiin myös tarve moottorien lämpöjen ja tehojen keräämiseen. Nämä olivat helppoja lisätä tulevaan projektiin mutta jätettiin toteuttamatta olemassa olevaan järjestelmään. Toisaalta tällöin ongelma oli lähinnä servo-ohjaimien firmwaren päivittämisessä eikä niinkään tiedonkeräyksessä.

M1.4.1 Aika tarpeen tunnistamisesta datapisteen lisäämiseen keräykseen

Tätä dataa ei kerätty POC vaiheessa.

M1.4.2 Lisättyjen datapisteiden suhde tunnistettuihin datapisteisiin.

100 %, mutta käytössä vasta hyvin rajalliset käyttäjävaatimukset.

M1.4.3 Kehittäjän mielipide järjestelmän ketteryydestä (1-5).

Arvio: 4. Ongelmana muutokset moniin erillisiin järjestelmiin.

Q1.5 Helpottaako informaation saatavuus työtä?

Tuotekehityksessä, käyttöönotossa ja asiakastuessa informaatiota on pystytty jo onnistuneesti hyödyntämään.

M1.5.1 Sidosryhmittäin koottu mielipide esitetyn informaation hyödyistä.

Alustavasti positiivisia, mutta vasta hyvin rajatulla käyttäjämäärällä.

Tavoite: Tarjota teknistä avustusta Monorail-automaatiojärjestelmän käyttäjälle.

Q2.1 Onko käyttäjän ongelmaan olemassa analyysi?

M2.1.1 Analyysien lukumäärä

Analyysit ovat vasta kehitysvaiheessa, muutamia esimerkkejä lukuun ottamatta.

M2.1.2 Analyysien lukumäärä / tunnistettujen analyysitarpeiden lukumäärä

Sekä analyysit että niiden tarpeiden selvitys ovat vasta työn alla.

M2.1.3 Analyysien kyvykkyys

Vaunun toiminnalle on kehitetty analyyseja muutamalle osa-alueelle, joiden avulla sen toimintaa on mahdollista seurata. Lisäksi on esitetty potentiaalinen kohde yhdelle ennakoivan huollon analyysille.

Q2.2 Toimivatko analyysit tehokkaasti (esimerkiksi ajoaika)

Pitkälle aikavälille tehtävät analyysit kestävät pitkään eivätkä ole aina tarpeellisia. Aikavälin suodattaminen ja oletusarvot tulisivat olla paremmin esillä websovelluksessa.

M2.2.1 Analyysien kehittämiseen käytetyt resurssit (käytetty suoritusteho, pilvipalveluiden kustannukset ja muut vastaavat)

Käytössä sisäverkossa toimiva kanta + paikallisesti ajettava ohjelma.

M2.2.2 Analyysien tuottamiseen tarvittavat resurssit (käytetty suoritusteho, pilvipalveluiden kustannukset ja muut vastaavat)

Käytössä sisäverkossa toimiva kanta + paikallisesti ajettava ohjelma.

Q2.3 Helpottavatko analyysit työtä?

M2.3.1 Sidosryhmittäin koottu mielipide analyysien hyödyistä.

Tuotekehityksessä, käyttöönotossa ja asiakastuessa tuote on koettu hyödylliseksi, mutta hyvin rajallisella käyttäjämäärällä.