LENTOTIETOAINEISTON VISUALISOINTI DATA- JA KÄYTTÄJÄLÄHTÖISESTI

Lappeenrannan–Lahden teknillinen yliopisto LUT

Tuotantotalouden diplomityö

2022

Eero Peltola

Tarkastajat: Professori Timo Kärri

Tutkijatohtori Antti Ylä-Kujala

TIIVISTELMÄ

Lappeenrannan–Lahden teknillinen yliopisto LUT

School of Engineering Science

Tuotantotalouden koulutusohjelma

Eero Peltola

Lentotietoaineiston visualisointi data- ja käyttäjälähtöisesti

Tuotantotalouden diplomityö

2022

74 sivua, 17 kuvaa, 5 taulukkoa ja 1 liite

Tarkastajat: Professori Timo Kärri ja tutkijatohtori Antti Ylä-Kujala

Avainsanat: data-analytiikka, ilmailu, sotilasilmailu, data, visualisointi

Lentokoneet tuottavat erittäin paljon lennon aikaista dataa eli lentotietoaineistoa. Dataa voidaan käyttää esimerkiksi lentäjien koulutuksessa, vianselvityksessä sekä huollon optimoinnissa. Ilmailualalla data-analytiikkaa on aiemmin toteutettu paljon yksittäisten lentojen näkökulmasta. Koko laivueen lentotietoaineistojen koontinäkymän toteutuksesta ei kuitenkaan ole merkittäviä esimerkkejä etenkään sotilasilmailuun liittyvässä kirjallisuudessa. Diplomityön teoriaosuudessa tutkitaan aiheen kirjallisuutta ja empiriaosuudessa toteutetaan käytännön toteutus eli demo lentotietoaineiston visualisoinnille.

Dataa voidaan analysoida sekä visualisoida hyvin monilla teknologioilla. Diplomityössä otetaan tarkempaan tarkasteluun kolme erilaista toteutustapaa; BI-työkalut, Pythonin Dash sekä JavaScriptin Vue. Työssä haastateltiin potentiaalisia loppukäyttäjiä käytännön demoon toteutettavien käyttötapausten keräämiseksi. Käyttötapaukset liittyivät lentotietoaineiston koontiin sekä lennon aikaisten tietojen käyttäjäystävälliseen visualisointiin. Lisäksi työssä haastateltiin diplomityön kohdeyrityksen omia ohjelmistokehittäjiä, joilta kerättiin kokemuksia teknologioista kirjallisuuden tueksi.

Työssä selvitettiin, että kaikilla tarkastelluilla työkaluilla olisi voitu toteuttaa demon käyttäjänäkymä. BI-työkalut edustivat helppokäyttöisintä low-code ratkaisua, kun taas JavaScriptin Vue mahdollisti eniten mahdollisuuksia visualisoinnin kustomoinnille. Lopullinen käytännön demo toteutettiin Pythonin Dashilla. Demon keskeisimpänä tuloksena mahdollistetaan koko laivueen lentohistorian lentojen haku lennonaikaisten parametrien perusteella. Lisäksi lentojen parametrien visualisointia kehitetään pidemmälle.

ABSTRACT

Lappeenranta–Lahti University of Technology LUT

School of Engineering Science

Industrial Engineering and Management

Eero Peltola

User-oriented visualizing of flight data

Master's thesis

2022

74 pages, 17 figures, 5 tables and 1 appendix

Examiners: Professor Timo Kärri and Post-doctoral researcher Antti Ylä-Kujala

Keywords: data-analytics, aviation, military aviation, data, visualization

Airplanes produce a lot of flight data. This data can be used, for example, in pilot training, troubleshooting and maintenance optimization. In the aviation sector, a lot of data analytics has previously been implemented from the perspective of individual flights. However, there are no significant examples of user interfaces for the entire fleet's flight data, especially from the military aviation point of view. The research is divided into theoretical and empirical parts. The theoretical part researches the literature of data analysis in aviation. In the empirical part, a practical demo for the visualization of flight data is carried out.

Data can be analyzed and visualized with many technologies. This research takes a closer look at three different alternatives for data visualization: BI tools, Python's Dash and JavaScript's Vue. Potential end users were interviewed in order to collect use cases for the demo. The use cases were related to combining the data from different flights and the user-friendly data visualization. In addition, case company's own software developers were interviewed to gather experiences of different technologies to support the literature.

The research concluded that the use cases for the demo could have been implemented with all of the examined tools. BI tools represented the easiest-to-use low-code solution, while JavaScript's Vue provided the most possibilities for customizing the visualizations. The final practical demo was developed using Python's Dash. The main result of the demo is the functionality to search flights from the entire fleet's flight history based on in-flight parameter values. In addition, the visualization of flight data was further developed.

KIITOKSET

Haluan kiittää kohdeyrityksenä toiminutta Patriaa kiinnostavan diplomityön mahdollistamisesta. Erityiskiitos etenkin Patrian puolelta ohjaajana toimineelle Simo Saariselle kaikista hyvistä kommenteista ja neuvoista, joita sain diplomityöprosessin aikana. Lisäksi suuri kiitos kaikille työssä haastatelluille ja muuten diplomityön toteutukseen osallistuneille. Lopuksi kiittäisin myös työn tarkastajia Timo Kärriä sekä Antti Ylä-Kujalaa asiantuntevasta ohjauksesta työn aikana.

Espoossa, 21.7.2022

Eero Peltola

LYHENNELUETTELO

BI	Business Intelligence. Suomennettuna liiketoimintatiedon hallinta.
Dash	Avoimen lähdekoodin Python-kirjasto, jolla voidaan luoda interaktiivisia selaimessa käytettäviä applikaatioita datan visualisoimiseksi.
Docker	Avoimen lähdekoodin virtualisointiteknologia. Mahdollistaa käyttöjärjestelmätason virtualisoinnin eli konttien suorittamisen isäntäkäyttöjärjestelmän alaisuudessa.
ETL	Tarkoittaa datan siirtämistä (Extract), muokkaamista (Transform) ja lataamista (Load).
FI	Fatigue Index. Suomeksi väsymisindeksi. Kuvaa lentokoneen rungon rasittuneisuutta.
HTML	HyperText Markup Language. Standardina oleva ohjelmointikieli, joka määrittelee dokumenttien esitystavan verkkoselaimessa.
OEM	Lyhenne sanoista Original Equipment Manufacturer, eli suomennenttuna alkuperäinen laitevalmistaja.
MRO	Akronyymi englannin kielen sanoista Maintenance, Repair, Overhaul. Tarkoittaa tuotannon ja kunnossapidon tarvikkeita.
SQL	Structured Query Language. Relaatiotietokannoissa käytetty standardoitu kyselykieli.

Sisällysluettelo

Kuvaluettelo

Taulukkoluettelo

1 Johdanto

Informaatioteknologian kehittyessä datan määrä on kasvanut eksponentiaalisesti globaalilla tasolla jo kauan ja kasvun uskotaan jatkuvan myös tulevaisuudessa (Holst, 2021). Datan merkitys korostuu etenkin toimialoilla, joilla sen perusteella tehdään merkittäviä turvallisuuskriittisiä päätöksiä. Ilmailuala on toiminut yhtenä edelläkävijöistä datan keräämisessä ja esimerkiksi sotilasilmailu on yksi eniten dataa kuluttavista puolustushaaroista (Augustin et al. 2021, s.1). Mahdollisuus luoda johtopäätöksiä useista eri tietolähteistä peräisin olevasta datasta antaa monia mahdollisuuksia parantaa ja kehittää kyvykkyyttä ja luotettavuutta ilmailussa.

1.1 Tutkimuksen tausta

Tämä tutkimus on toteutettu toimeksiantona Patria Aviation Oy:lle. Tutkimuksessa on käytettävissä huomattavan laaja data-aineisto (lentotietoaineisto), ja sen visualisoinnissa nykyistä havainnollisemmin ja helppokäyttöisemmin nähdään paljon mahdollisuuksia tukea tietoaineistoa päätöksenteon tukena käyttäviä loppukäyttäjiä. Yritys on halukas kartoittamaan datan visualisointiin sopivien teknologioiden etuja sekä haittoja sekä toteuttamaan tämän diplomityön yhteydessä toiminnallisen toteutuksen lentotietoaineiston visualisoinnille. Yritys on jo rakentanut on-premises ohjelmistoalustaa, jonka päälle tarvittava visualisointityökalu voidaan asentaa.

Tutkimuksen aihe, lentotietoaineiston analysointi sekä visualisointi, liittyy vahvasti datatieteiden kirjallisuuteen ja tässä työssä myös tarkastellaan aiheen kirjallisuutta sekä uusimpia suuntauksia datatieteisiin liittyvistä artikkeleista. Tutkimuksia datan koonnista sekä sen visualisoinnista on tehty aiemmin ja myös sotilasilmailuun liittyviä toteutuksia on mahdollista löytää kirjallisuudesta (Augustin. et al. 2021). Esimerkiksi Zhang (2018 s.12) mainitsee, että mahdollisuus tehdä lentokoneen suorituskyvyn arviointia sekä arvioida lentäjän lentotaitoja ovat tärkeitä toiminnallisuuksia, jotka tallennettu lentotietoaineisto mahdollistaa.

Aiemmissa toteutuksissa on nähtävissä, kuinka tiedon kerääminen eri järjestelmistä yhteen paikkaan sekä sen analysoiminen tekoälyn sekä koneoppivien mallien avulla visuaaliseen muotoon voi luoda paljon lisäarvoa loppukäyttäjälle. Kyseiset kirjallisuudesta löytyvät toteutukset ovat kuitenkin keskittyneet esittämään lentotietoaineiston analyysin pitkälti yksittäisen lennon näkökulmasta. Tämän työn soveltava osuus pyrkii jatkokehittämään jo olemassa olevia ratkaisuja siten, että yksittäisen lennon lentotietoaineistoa pystytään vertaamaan koneyksilön tai koko laivueen aiempaan dataan helposti.

1.2 Tutkimuksen tavoitteet ja rajaukset

Tutkimuksen tavoitteena on selvittää millä teknologioilla ja miten laajaa lentotietoaineistoa voidaan esittää niin, että vastataan loppukäyttäjien tarpeisiin datan analysoinnista sekä visualisoinnista parhaiten. Loppukäyttäjien tarpeet selvitetään tarkemmin avoimilla haastatteluilla ja ne keskittyvät lentotietoaineiston koostetumpaan sekä tehokkaampaan analysointiin.

Tutkimuksen keskeiset tutkimuskysymykset ovat

1. *Millaisilla teknologioilla voidaan toteuttaa käyttäjänäkymiä data-analytiikkajärjestelmiin?*

2. *Mitkä ovat eri datan visualisointiteknologioiden vahvuudet ja heikkoudet?*

3. *Millainen datan visualisoinnin käytännön toteutus soveltuu parhaiten kohdeyritykselle tutkimuksen kohteena olevan lentotietoaineiston visualisointiin?*

Tutkimus etenee teorian tasolta kohti empiiristä toteutusta. Tutkimuskysymyksistä ensimmäiseen pyritään vastaamaan teorian näkökulmasta, toiseen molempien sekä teorian että empirian näkökulmasta ja kolmanteen vastataan jo lähes kokonaan empirian näkökulmasta.

Tutkimuksessa data-analytiikkajärjestelmistä rajataan diplomityöhön tarkempaan tarkasteluun datan käsittely sekä visualisointi. Diplomityön puitteissa ei ole tarvetta kehittää yrityksen tietojärjestelmiä eikä hankkia uutta dataa. Työhön kuitenkin sisältyy mahdollisten visualisointityökalujen käyttöönotto ja datan mahdollinen aggregointi sekä eheyttäminen. Kehitettävän datan visualisoinnin käyttöliittymän tulee olla käytettävissä mikropalveluarkkitehtuurin mukaisessa järjestelmässä.

Nykyisellään data on pääasiassa yksittäisissä lentotiedostoissa. Alustavat vaihtoehdot visualisoinnille ovat BI-työkalut, Python sekä JavaScript. Tutkimuksen aihe keskittyy lentotietoaineiston visualisointiin, eli työssä rajataan pois ilmailun työkaluista lennonjohtoon liittyvät toteutukset. Lentotietoaineisto on peräisin sotilaslentokoneista, mutta tutkimuksessa ei käsitellä asejärjestelmiin liittyvää dataa. Näin tutkimuksen tulokset ovat hyvin rinnastettavissa myös siviili-ilmailun puolelle.

1.3 Tutkimuksen menetelmät ja aineisto

Tutkimuksen teoriaosuudessa tarkastellaan data-analytiikan sekä ilmailualan kirjallisuutta. Lisäksi data-analytiikassa käytettyjä visualisointitapoja selvitetään eri kirjallisuuslähteistä. Hakutermeinä toimivat muiden muassa ”Data Visualization”, ”Aviation”, ”Analytics”, ”BI-työkalut”, ”Dash” sekä ”Vue”. Haut tehtiin eri tietokantoihin, kuten LUT Primoon, Scopukseen, Google Scholariin, sekä yliopistojen julkaisualustoille kuten Aaltodocciin. Lisäksi blogeja sekä muuta informaatiota haettiin Google-hauilla kirjallisuuslähteiden tueksi.

Tutkimus on empiriaosuudessa toteutettu laadullisilla eli kvalitatiivisilla tutkimusmenetelmillä, jossa käytössä on haastattelut loppukäyttäjien tarpeisiin perehtyneille asiantuntijoille sekä ohjelmistokehittäjille. Puusan, Juutin ja Aaltion (2020) mukaan laadullisessa tutkimuksessa aineiston hankinta voi koostua useammasta eri menetelmästä ja tutkimuksessa ollaan kiinnostuneita tutkimusjoukon kokemuksista. Tämä työ on laadullinen tutkimus, jossa aineistoa hankitaan haastatteluilla sekä havainnoinnilla. Lisäksi työssä on konstruktiivisen tutkimuksen elementtejä, koska tarkoituksena on rakentaa tieteellistä otetta käyttäen demottava analytiikkatyökalu. Lukan (2001) mukaisesti konstruktiivisella tutkimuksella tarkoitetaan tutkimusta, jossa pyritään ratkaisemaan reaalimaailman ongelmia kehittämällä jokin uusi konstruktio.

Haastatteluissa on käytetty kahta eri haastattelumenetelmää; avoimia sekä puolistrukturoituituja teemahaastatteluja. Teemahaastattelussa haastattelua on ohjattu haluttuun teemaan, joko selvitettäessä loppukäyttäjän vaatimuksia tai eri toteutustapojen toimivuutta yrityksessä ohjelmistokehittäjiltä. Teemahaastattelu on vapaamuotoinen ja joustava haastattelumenetelmä, jossa haastattelu etenee etukäteen valittujen teemojen ja niitä koskevien tarkentavien kysymysten kautta (Puusa et al 2020). Haastattelut toteutettiin henkilö- sekä ryhmähaastatteluina. Yhteensä haastatteluja toteutettiin 10, joista neljä oli ryhmähaastatteluja ja kuusi henkilöhaastatteluja. Haastattelujen lisäksi työn empiirisessä osuudessa keskeisimpänä aineistona on käytetty laajaa satojen gigatavujen kokoista lentotietoaineistoa, josta kerrotaan tarkemmin luvussa 5.1.

1.4 Tutkimuksen rakenne

Tutkimus on jaettu kahteen suurempaan osakokonaisuuteen, työn alkupuolen teoriaosuuteen sekä työn loppupuolen empiiriseen osuuteen. Tutkimuksessa on koottu tietoa kirjallisuudesta, haastatteluista sekä kehittämällä käytännön toteutus lentotietoaineiston visualisointityökalusta. Näin tutkimuksessa yhdistyvät sekä teoreettinen, että empiirinen elementti kuten konstruktiivisen tutkimuksen luonteeseen kuuluu (Kasanen et al. 1991, s 323). Kuvassa 1 on esitetty työn keskeinen eteneminen.

Kuva 1. Työn eteneminen tiivistetysti

Työ on jaettu kuuteen päälukuun, joiden tarkempi tarkoitus on esitetty taulukon 1 input/output -kaaviossa. Kaavio esittää jokaisen luvun saamat syöttötiedot ja millaista tietoa ja tuloksia saadaan tuotettua. Input/output -kaavion tarkoitus on osoittaa, kuinka eri luvut muodostavat työstä yhtenäisen asiakokonaisuuden.

Taulukko 1. Input-output malli työn kulusta

	Input	Output
Luku 1: Johdanto	Tutkimuksen esittely ja johdanto.	Rajaukset, aihe, tutkimuskysymykset ja menetelmät
Luku 2: Data-analytiikkajärjestelmät ja datan visualisointi	Data-analytiikan termistö sekä teknologiat kirjallisuudesta	Data-analytiikan merkitys sekä eri teknologioiden esittelyt
Luku 3: Data-analytiikka ilmailualalla	Ilmailualan datan käsitteet ja kirjallisuus	Ilmailualan data-analytiikan esittely ja pääpiirteet
Luku 4: Visualisointityökalun valinta kohdeyritykselle	Asiantuntijahaastattelut sekä aiempien lukujen teoria	Haastattelutulokset, joista ilmenee kokemukset eri teknologioista sekä demolle halutut käyttötapaukset
Luku 5: Visualisointityökalun käytännön toteutus	Haastattelutuloksista saadut käyttötapaukset sekä teoria	Käytännön toteutuksen esittely
Luku 6: Johtopäätökset ja yhteenveto	Työn keskeiset tulokset	Tulosten yhteenveto, pohdinta sekä jatkotutkimusaiheet

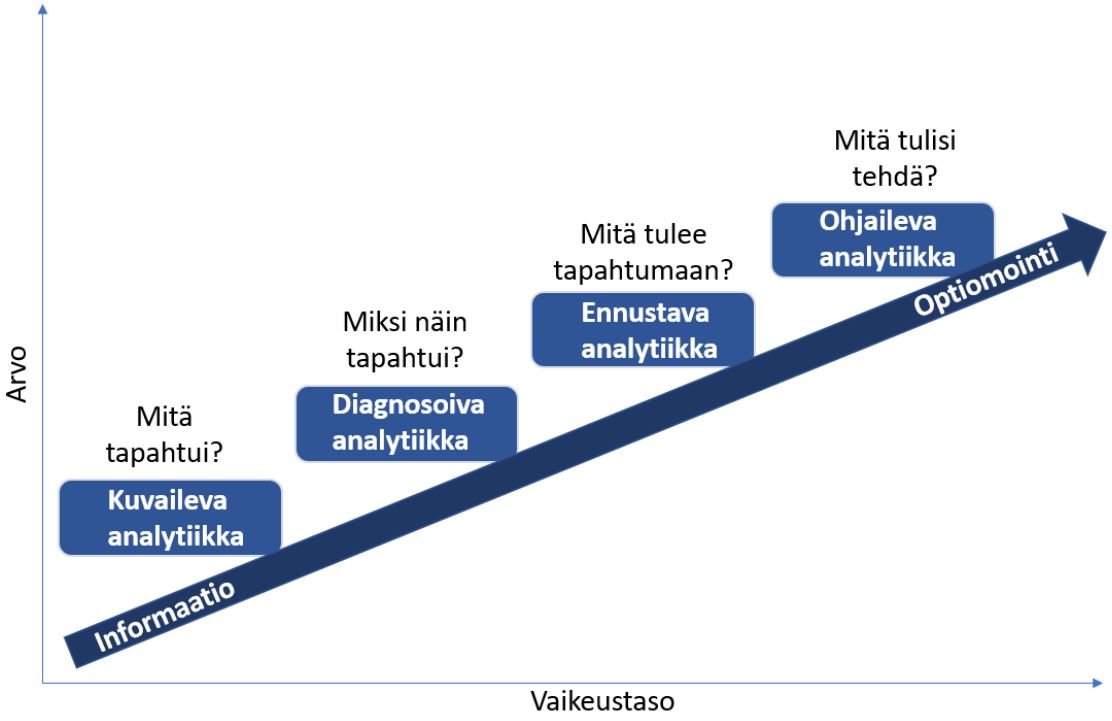
2 Data-analytiikka ja datan visualisointi

Tässä luvussa käsitellään data-analytiikka yleisellä tasolla ja esitellään data-analytiikassa käytettyjen visualisointimahdollisuuksien eroja. Datan hankinta, analysointi sekä visualisointi voivat kuitenkin olla hyvinkin erilaisia prosesseja riippuen millä toimialalla ja minkälaista dataa halutaan analysoida.

2.1 Data-analytiikka yleisesti

Data-analytiikalla tarkoitetaan prosessia sekä menetelmiä, joilla eri tavoin kerätystä tiedosta eli datasta pyritään luomaan korkeamman tason informaatiota päätöksenteon tueksi. Data-analytiikalla pyritään vastaamaan kysymyksiin, hankkimaan uutta tietoa sekä tunnistamaan trendejä. Se eroaa datatieteestä, joka on prosessi datasettien rakentamiseksi, siivoamiseksi ja strukturoimiseksi analyysiä sekä tiedon hankintaa varten (Stobierski, 2021; Runkler 2016, s.2). Data-analytiikka voi siis pitää sisällään esimerkiksi datan jalostamista johdolle ymmärrettävässä muodossa, datan keskittämistä ja yhdistämistä samaan paikkaan, uusien ohjelmistojen käyttöönottoa sekä datan visualisoimista. Kun data-analytiikkaa käytetään liiketoiminnassa, puhutaan usein liiketoiminta-analytiikasta (Badiru 2020, s.55). Data-analytiikka on hyvin laaja termi, joten työssä käytetyllä termillä *data-analytiikkajärjestelmä* tarkoitetaan niiden ohjelmistojen kokonaisuutta, joilla data-analyysiä tehdään.

Mahdollisesti yksi eniten analytiikan ja datatieteiden esityksissä käytetty malli on Gartnerin (2014) analytiikan maturiteettimalli, joka on esitetty kuvassa 2. Malli pyrkii kuvaamaan, kuinka data-analytiikkaa kehitetään yrityksissä asteittain kohti kehittyneempää tasoa. Myös Strbierski (2021) sekä Prabhu (2019, s.52) kertovat mallin mukaisesti, että data-analytiikka voidaan luokitella joko kuvailevaksi, diagnostisoivaksi, ennustavaksi tai ohjailevaksi. Gartnerin malli pyrkii osoittamaan, kuinka analytiikan taso vaikuttaa sen luomaan liiketoiminta-arvoon. Mallissa analytiikka on jaettu neljään eri maturiteettitasoon. Mitä kypsempi analytiikan taso, sitä vähemmän ihmisen tekemää työtä tarvitaan päätöksen tekemiseen sekä toteutukseen.

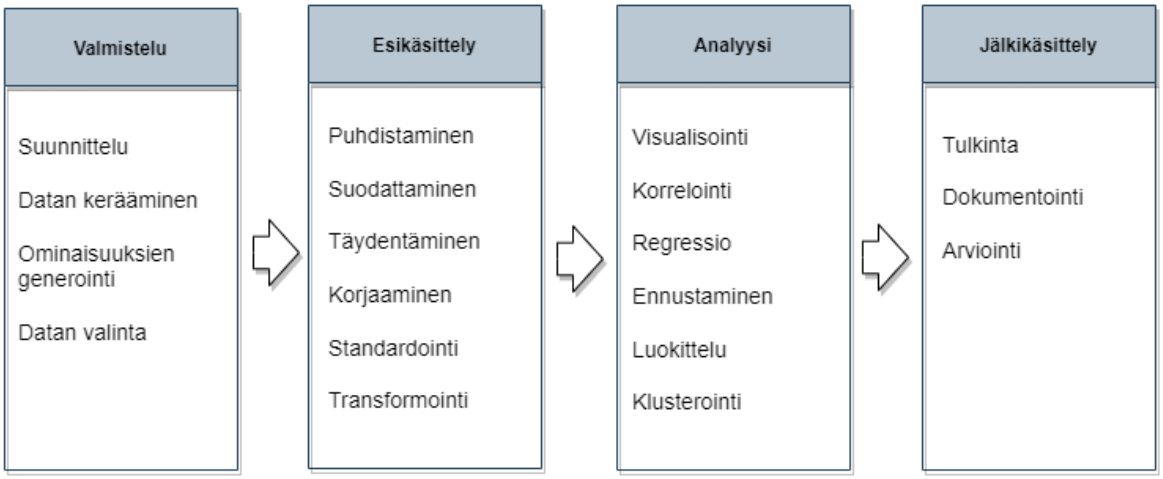


Kuva 2. Gartnerin (2014) analytiikan maturiteettimalli

Kuvassa 2 ensimmäisen tason kuvaileva analytiikka on menneisyyteen pohjautuvaa kattavaa datan esittämistä sekä visualisointia. Toisen tason diagnosoivalla analytiikalla pyritään taas löytämään syitä havaittujen ilmiöiden taustalla sekä löytämään datasta oleellinen tieto. Kolmannen tason tulevaisuuteen keskittyvä ennustava analytiikka tarkoittaa nimensä mukaisesti historiatietoon perustuvaa ennustamista algoritmien avulla. Neljännen tason ohjaava analytiikka vie tämän vielä pidemmälle käyttämällä kehittyneitä analytiikkatyökaluja tarjotakseen suosituksia päätöksenteon tueksi (Prabhu et. al 2019, s.52). Maturiteettimallia voidaan käyttää esimerkiksi kartoitettaessa analytiikan nykytilaa yrityksessä peilaamalla mallia omaan toimintaan. Toisaalta mallia on myös kritisoitu siitä, että todellisuudessa data-analytiikan kehitys ei ole täysin portaittaista, vaan kehitystä voi tapahtua useilla portailla samaan aikaan. Lisäksi malli jättää korostamatta, että todellinen arvo liiketoiminnalle tulee vasta kun analytiikka vaikuttaa päätöksentekoon. Aina ei siis välttämättä ole järkevintä odottaa alimman portaan tietovarastoprojektien valmistumista ennen kehittyneempiä analyysejä (Widjaja 2020). Analytiikka on usein myös eri portaita läpileikkaavaa.

2.2 Data-analytiikan prosessi

Kuvassa 3 on esitetty Runklerin et al. (2016) näkemys, millaisella prosessilla data viedään sen keräyksestä visualisoinnin tasolle. Runkler jakaa data-analytiikkaprojektin näihin neljään vaiheeseen, mutta myös tästä prosessista on olemassa hieman erilaisia versioita. Badirun (2020, s.55) mukaan data-analytiikkaan liittyy usein tarve saada tietoa projektinhallintaan, jolloin data-analytiikan päävaiheet ovat datan keräys, datan analyysi sekä esitys, päätöksenteko ja lopulta toiminnan aloittaminen. Keskeinen idea prosessille on kuitenkin sama eri lähteissä. Esimerkiksi usein käytetty termi ETL, joka tarkoittaa datan louhimista (Extract), muokkaamista (Transform) ja lataamista (Load) sisältyy Runklerin prosessin kahteen ensimmäiseen vaiheeseen.



Kuva 3. Datan visualisoinnin vuokaavio Runklerin (2016) mukaan.

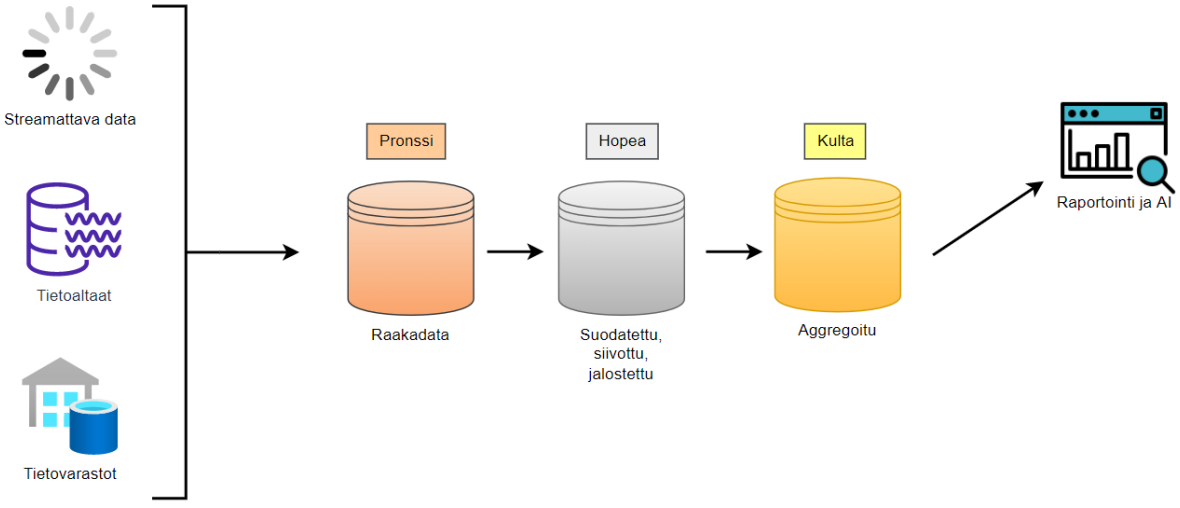
Ensimmäisessä valmistelun vaiheessa määritellään haluttu päämäärä sekä data hankitaan usein useasta eri lähteestä. Data voi olla strukturoidun datan lisäksi strukturoimatonta tai semi-strukturoitua dataa, eli esimerkiksi kuvia, videoita tai tekstejä (Chawla et al. 2018). Näiden monimuotoisempien datan muotojen tallentaminen ei onnistu perinteisissä relaatiotietokannoissa, vaan sitä voidaan kerätä esimerkiksi tietoaltaisiin (data lake), joissa data on tallennettu alkuperäisessä muodossaan (Llave 2018).

Datan hankinnan jälkeen esikäsittelyn vaiheessa se täytyy usein puhdistaa sekä muokata haluttuun, usein strukturoituun muotoon. Tässä vaiheessa on hyvä tunnistaa datan virheet, kuten esimerkiksi poikkeavat datapisteet (outliers). Esikäsittelyssä dataa voidaan myös esimerkiksi suodattaa, koska kaikkea dataa ei aina tarvita analyysin toteuttamiseksi (Runkler 2016, s. 23). Siistittyä dataa tyypillisesti tallennetaan tietovarastoon, josta se on helposti saatavilla analyysi- sekä BI- työkalujen käytettäväksi. Toisaalta on huomioitavaa, että tähän ei ole yhtä oikeaa ratkaisua. Esimerkiksi joissain tapauksissa analyysit ja raportit voidaan yhdistää suoraan tietoaltaisiin tai datalähteisiin (Llave 2018).

Kolmannessa analyysin vaiheessa tehdään varsinainen analyysi sekä visualisointi siistitylle datalle. Vo.T.H et al. (2017, s.59) mukaan datan visualisoiminen on yksi tärkeimmistä data-analytiikan osa-alueista, koska sen avulla pystytään näkemään analyyttiset tulokset, huomata poikkeavat datapisteet sekä tekemään päätökset mallien rakentamiselle. Datan visualisoinnin työkaluiksi esimerkiksi Ali et al. (2016) sekä Chawla et al (2018) julkaisuissa on mainittu Tableau, Microsoft Power Bi, SAS Visual Analytics, Plotly, Gephi, D3.js ja Excel. Toisaalta kuten Ali et al. (2016) mainitsee, visualisointityökaluja ei voida laittaa paremmuusjärjestykseen, vaan eri työkalujen sopivuus riippuu organisaation vaatimuksista.

Neljännessä vaiheessa dataa tulisi tulkita sekä analyysin avulla tehdä johtopäätöksiä päätöksenteon tueksi. Mikäli ei olla rakentamassa analytiikkatuotetta, varsinainen lisäarvo syntyy datasta vasta tässä vaiheessa. Siksi onkin erittäin tärkeää, että koko data-analytiikan prosessissa pidetään mielessä, mistä varsinainen lisäarvo oikeasti syntyy. Väärin toteutettuna data-analytiikan prosessi voi luoda tietoa, jota kukaan ei koskaan käytä (Widjaja 2020). Neljännessä vaiheessa voi analyysien jakaminen tulla haasteeksi. BI-työkaluja käytettäessä voidaan raporttien jakamisessa käyttää samojen palveluntarjoajien tarjoamia pilvipohjaisia tai on-premises tyyppisiä palveluita kuten Power BI serviceä. Myös ohjelmistopohjaisissa ratkaisuissa käyttäjän todentaminen sekä valtuuttaminen täytyy ottaa huomioon raporttien jakamisen tietoturvaa tarkasteltaessa.

On kuitenkin huomioitavaa, että data-analytiikan prosessille on olemassa useita erilaisia toteutustapoja ja eri ohjelmistoja. Kehitys on alalla myös hyvin nopeaa ja yritykset kehittävät jatkuvasti tehokkaampia keinoja datan käsittelylle. Polyzotisin sekä Zaharian (2021, s.3-4) mukaan datatieteissä yksi suurimpia trendejä on datan, koodin sekä siitä luotujen koneoppimismallien kokonaisvaltainen versionhallinta. Versionhallinta on ollut jo pitkään ohjelmistokehityksessä käytössä, mutta nykyään yhä useampi järjestelmä mahdollistaa koko datan sekä siitä kehitettyjen koneoppivien mallien historian tallentamisen. Esimerkiksi avoimen lähdekoodin teknologia Delta Lake on kehitetty yhdistämään aiemmin erilliset tietoaltaat, tietovarastot sekä jatkuvasti päivittyvät datalähteet yhteen helpommin hallittavaan paikkaan (Armbrust et al. 2020; Microsoft 2022c). Kuvassa 4 on esitetty Delta Laken teknologian mukainen datanhallinnan lähestymistapa, jossa data varastoidaan kolmessa eri tasossa. Siinä dataa jalostetaan pronssilta kultatasolle, josta sitä voidaan visualisoida parhaiten siten, että kyselyitä ei tarvitse tehdä koko raakadatalle (Microsoft 2022c). Data-analytiikan prosessin aikana raakadataa itsessään ei siis muokata, vaan siitä tehdään yhä jalostetumpia kopioita.



Kuva 4. Datan laadun eri tasot data-analytiikkaprosessissa. (Mukaillen Microsoft 2022c).

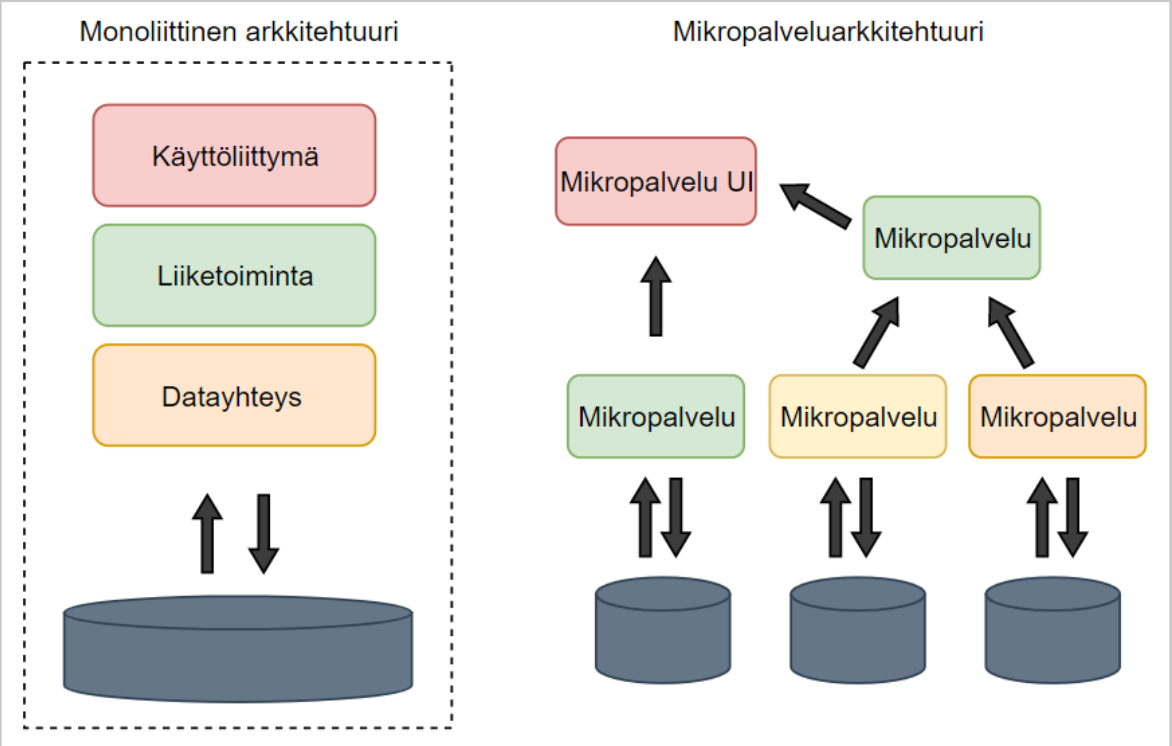
Mikäli dataa on erittäin paljon, tuo se lisähaasteensa datan visualisoinnille. Niin sanotun Big Datan visualisointi on haastavaa sen määrän, erilaisuuden sekä nopeuden takia. Big datalla

tarkoitetaan datasettejä, joissa korostuu datan suuri määrä, lisääntymisnopeus sekä moninaisuus. Termiä käytetään datatieteissä hyvin paljon ja sille on useita määritelmiä. Vuonna 2001 analyytikko Doug Laney määritteli Big Datan kolmen V:n avulla määrä, nopeus, moninaisuus (Volume, Velocity, Variety), mutta nykyään jotkut määrittelevät sen jo 17 V:n ja 1 C:n avulla. Joka tapauksessa kaikissa määritelmissä yhteistä on se, että dataa kerätään jatkuvasti enemmän sekä sitä on haastavaa käsitellä sen monimutkaisuuden sekä määrän takia (Panimalar et al. 2017). Toisaalta Prabhu et al. (2019, s.2-3) nimeävät Big Dataksi internetin kokoluokan superlaskennan, jossa käytetään teknologioina hajautettua tietojenkäsittelyä, rinnakkaista prosessointia, klusterilaskentaa sekä hajautettuja levyjärjestelmiä. Haastavaa suurten datamassojen kanssa työskentelyssä on se, kuinka esitetään tehokkaasti datan visualisoinnin ja analyysien tulokset. Suurten datamassojen visualisoinnissa on tärkeää tunnistaa kiinnostavat yhteydet ja kaavat datan seasta. Visualisoinnissa tulee ottaa huomioon sopiva määrä dimensioita, joiden avulla dataa esitellään. Liian yksinkertainen esitystapa voi jättää kiinnostavia yhteyksiä näyttämättä, mutta taas liian moniulotteinen esitystapa voi olla loppukäyttäjälle liikaa (Chawla et al. 2018).

2.3 Mikropalveluiden käyttö data-analytiikassa

Teknologian kehitys datatieteissä on saattanut luoda kuvan, että data-analytiikan suunnittelu, testaus ja käyttöönotto olisi muuttunut helpommaksi viime vuosina. Kuitenkin asia saattaa olla jopa päinvastainen ja alalla on vielä monia haasteita, kuten esimerkiksi puute osaavista ammattilaisista. Nykypäivänä suosiota keränneet hajautetut järjestelmät voivat olla hyvin haastavia toteuttaa etenkin suurten datamäärien kanssa. Mikropalveluteknologiat ovat kuitenkin mahdollistaneet lupaavia lähestymistapoja näiden ongelmien ratkaisemiseksi (Staegemann et al. 2021). Viime vuosina useissa suurikokoisissa järjestelmissä on siirrytty perinteisestä monoliittisesta ohjelmistosta kohti palvelukeskeistä arkkitehtuuria, jossa tietojärjestelmien osat toimivat itsenäisinä palveluina. Mikropalveluarkkitehtuuri voidaan myös nähdä osana tätä kehitystä (Ponce et al. 2019).

Mikropalveluilla (microservices) tarkoitetaan pieniä ohjelmia, joilla on yksittäinen toimiva, skaalattava sekä erikseen testattava vastuutehtävä. Mikropalvelut eroavat perinteisestä monoliittisesta sovelluksesta siten, että perinteisen sovelluksen moduuleja ei voi suorittaa itsenäisesti. Mikropalvelut ovat yleistyneet huomattavan nopeasti. Konsepti itsessään luotiin 2000-luvun lopulla, mutta termi ”microservices” alkoi yleistyä vasta 2014. Kuitenkin esimerkiksi IT alan markkinatutkimuksia tekevä International Data Corporation (IDC) ennusti vuoden 2019 viisivuotisennusteessaan, että vuonna 2022 mikropalveluita käytettäisiin 90 % uusissa IT alan sovelluksissa. (Columbus 2019; Larrucea 2018)



Kuva 5. Mikropalveluarkkitehtuuri verrattuna monoliittiseen sovellukseen. (Mukaillen Nakazawa 2018; Johnson & Shiff 2021)

Mikropalveluita käytetään sekä suoritetaan usein konteissa (containers), joista tunnetuimpia ovat Docker -yrityksen vuonna 2013 kehittämät Docker-kontit. Avoimen lähdekoodin Docker -alustan kontit mahdollistavat eristetyn ajoympäristön sovelluksille ja palveluille (Preeth et al. 2015). Aiemmin samaa on voitu tehdä esimerkiksi virtuaalipalvelimilla, mutta niiden heikkoutena on aina toimimiseen vaadittu käyttöjärjestelmä sekä etukäteen varatut

resurssit. Konttien suorittamisessa voidaan käyttää niiden hallintaohjelmistoa, kuten esimerkiksi Kubernetesta (Bucchiarone et al. 2020, s. 17). Mikropalveluarkkitehtuurissa korostuu tietoturvan tärkeys ja turvallinen palvelujen välinen tiedonkulku. Koska hajautetun ohjelmiston mikropalveluita ajetaan lähtökohtaisesti useilla koneilla, myös tietoturvaa tarvitaan usealla eri tasolla verrattuna perinteiseen ohjelmistoon. Tietoturva on huomioitava fyysisten palvelinten laiteohjelmistoissa, palvelinten käyttöjärjestelmissä, kontinhallintaohjelmistoissa ja muissa mahdollisissa välikerroksissa, omissa ohjelmistoissa, konttien välisessä tietoliikenteessä sekä pääsynhallinnassa (Indrasiri & Siriwardena, s. 313–345). Oauth2 on esimerkki protokolasta, joka määrittelee turvalliseksi katsotut käytännöt mikropalvelujen autentikaatiolle (Bucchiarone et al. 2020, s.285). Kuvassa 5 on esitetty, kuinka perinteinen monoliittinen ohjelmisto eroaa mikropalveluista.

Mikropalvelut mahdollistavat ohjelmiston eri osien samanaikaisen kehityksen helpommin ja ne voidaan tarvittaessa tehdä eri ohjelmistokielillä. Lisäksi yksi suurimmista hyödyistä on pienempi vaadittu työmäärä muutoksille sekä ylläpidolle. Muutoksia varten pelkästään muokattu mikropalvelu tarvitsee ottaa uudelleen käyttöön (Staegemann et al. 2021). Ohjelmiston erittely eri mikropalveluihin voi olla etenkin data-analytiikassa hyödyksi, koska data-analytiikan prosessi pitää sisällään monia hyvin erilaisia ja usein testatessa hajoamisherkkiä vaiheita. Esimerkiksi datan yhdistäminen, koneoppivien mallien opettaminen ja käyttöliittymän suorittaminen voivat olla eri mikropalveluissaan. Tällöin kehitystyö voi yksinkertaistua, analytiikkaratkaisusta voidaan tehdä modulaarisempi sekä esimerkiksi datan käsittelyssä voidaan mikropalvelulle allokoida rajoitettu määrä muistia ja laskentatehoa helpommin.

Toisaalta mikropalveluilla on kuitenkin vielä haasteita. Vaikka mikropalvelut ovat lisääntyneet viime vuosina, kirjallisuudessa olisi vielä lisäkysyntää ohjelmistojen suunnittelulle sekä toteutukselle hyväksi havaituista käytännöistä. Mikropalveluarkkitehtuuria voi olla haastava toteuttaa toimivasti, joten se ei aina välttämättä ole oikea ratkaisu kaikille sovelluksille (Bucchiarone et al. 2020, s. 10-11; Anderson 2020). Mikropalveluarkkitehtuurin yleinen haaste tulee sen joustavuudesta ja etenkin sen hallinnasta. Suurissa toteutuksissa voi nousta haasteeksi tunnistaa mihin kaikkiin

mikropalveluihin mikropalvelu X:n rajapinnassa tehty muutos vaikuttaa. Kuitenkin Staegemann et al. (2021) mainitsevat mikropalvelujen olevan potentiaalinen lähestymistapa Big Datan analysoimiseksi.

2.4 Vaihtoehdot visualisointiin data-analytiikkajärjestelmissä

Visualisointityökalun tulee olla yhteensopiva käyttötarkoitukseensa. Pääsääntöisesti mitä monimutkaisempaa sekä kustomoidumpaa analyysiä loppukäyttäjä haluaa tehdä, sitä vähemmän valittu visualisointityökalu voi olla valmis pakettiratkaisu, kuten BI-työkalu. Tässä luvussa esitellään yleisimmät ja tähän diplomityöhön halutut visualisoinnin työkalut. Samoja työkaluja tarkastellaan sekä vertaillaan myöhemmin työn empiirisessä osuudessa luvussa 4.5 kohdeyrityksen ohjelmistokehittäjille tehtyjen haastattelujen kautta. Tämän lisäksi vertailua eri visualisointityökaluista on koottu luvun 4.6 taulukkoon 5.

2.4.1 BI-työkalut

Tähän diplomityöhön valittiin tarkempaan tarkasteluun eniten käytetyt sekä markkinoiden johtavat työkalut Microsoft Power BI sekä Tableau, jonka Salesforce osti vuonna 2019. Gartner (2022) julkaisee vuosittain paljon siteerattuja ohjelmistoalan vertailuja. Kuvassa 6 on esitetty Gartnerin BI-työkalujen nelikenttä, jossa on eritelty BI-työkalujen markkina-asemaa. Microsoftin Power BI sekä Salesforcen Tableau ovat selkeästi kaksi suosituinta BI-työkalua. Tässä diplomityössä ei keskitytä eri BI-työkalujen eroihin syvällisesti, koska aiemmat tutkimukset ovat osoittaneet suosituimpien BI-työkalujen päätoiminnallisuuksien olevan pitkälti samanlaisia, ja suurimpien erojen olevan palvelujen erilaisissa myyntipaketeissa (Oliveira & Bernardino 2020). Toisaalta erojakin on ja yrityksen tulisi aina tarkistaa valittavan työkalun toimivuus omaan käyttötapaukseen. Esimerkiksi on-premises raporttipalvelin voi Tableaulla toimia myös Linuxilla, kun taas Power BI -palvelin vaatii Microsoftin oman ympäristön (Tableau 2022, Microsoft 2022b).

Kuva 6. Gartnerin BI-työkalujen vertailun nelikenttä (Gartner 2022)

Microsoftin **Power BI** on pysynyt sekä Gartnerin (2022), että myös Forresterin (2021) BI-työkalujen vertailujen ensimmäisellä sijalla yhtäjaksoisesti useiden vuosien ajan. Edullinen hinta sekä helposti ladattava ilmaisversio ovat vahvistaneet Power BI:n markkina-asemaa. Lisäksi Microsoftilla on ollut jo valmiiksi massiivinen asiakaskunta Microsoft Office toimisto-ohjelmien menestyksen takia. Laaja käyttäjäkunta on puolestaan mahdollistanut sen, että Power BI:n käyttöön liittyviä ohjeistuksia sekä oppaita löytyy internetistä todennäköisesti kilpailijoita enemmän.

Microsoft (2022a) tarjoaa Power BI -tuotetta eri asiakasryhmille useissa vaihtoehtoisissa kokoonpanoissa. Järjestelmän pohjana toimii Power BI Desktop (PBI Desktop), joka on

ilmainen Windows-käyttöjärjestelmälle asennettava sovellus. PBI Desktopilla pystyy hakemaan dataa lähes kaikenlaisista yrityksen sisäisistä lähteistä ja/tai pilvipalveluista, analysoimaan sitä ja luomaan interaktiivisia raportteja. Sulavan (2019) mukaan PBI Desktop ei ole kuitenkaan itsessään loppukäyttäjille tarkoitettu valmis BI-työkalu, koska raporttien jakamiseen sekä osaan sen toiminnallisuuksista vaaditaan lisenssi. Lisenssejä on neljää päätyyppiä:

- Power BI Free on ilmainen henkilökohtaiseen käyttöön tarkoitettu lisenssi. Se mahdollistaa raporttien julkaisun julkiseksi internettiin, mutta ei yksityistä jakamista käyttäjien välillä.
- Power BI Pro on 8,4 €/kk/käyttäjä pääasiassa pienille ja keskisuurille organisaatioille suunnattu lisenssi. Sisällön jakaminen on julkisen julkaisemisen lisäksi mahdollista ainoastaan Pro-käyttäjien välillä, joten kaikilla sisältöä tuottavilla, jakavilla ja katsovilla käyttäjillä tulee myös olla Pro-lisenssi. Power BI Pro kuuluu valmiiksi Microsoftin Office 365 E5 toimisto-ohjelmapakettiin.
- Power BI Premium on alkaen noin 4200 €/kk tai 16,9 €/kk/käyttäjä ja on suunnattu etenkin suurille organisaatioille. Se sisältää valmiiksi kehittyneempiä analyysityökaluja, kuten tekoälytoimintoja tekstin ja kuvien tulkitsemiseksi. Raportteja pystytään jakamaan myös Free-käyttäjille. Kapasiteettikohtaiseen Premium -lisenssiin kuuluu myös Power BI Report Server, joka mahdollistaa raportoinnin yrityksen sisäisesti Microsoft-ympäristössä. (Microsoft 2022b)
- Power BI Embedded on Platform-as-a-Service (PaaS)-periaatteella toimiva BI-kehittäjille suunnattu lisenssi, joka mahdollistaa raporttien upotuksen pelkästään kustomoituihin sovelluksiin. Hinnoittelu perustuu käyttöaikaan ja on noin 600€/kk jatkuvassa käytössä (Sulava 2019).

Tableau on puolestaan Salesforcen tarjoama BI-työkalu, jolla on hyvin laaja sekä vankka kannattajakunta. Tableau tunnetaan etenkin sen intuitiivisesta sekä visuaalisesta datan käsittelystä (Gartner 2022). Alkuvuonna 2022 Tableau on hinnoiteltu 15-42-

70€/kuukausi/käyttäjä riippuen onko käyttäjällä analyysien katselu-, muokkaus-, vai valmistusoikeus. Tämän lisäksi lisäkustannuksia muodostuu vähintään 250 €/kk, mikäli käytössä on Tableau Online. Ostettavien lisenssien minimimäärät kuitenkin poistuivat vuonna 2021, joten työkalu on nyt aiempaa helpommin myös pienempien yritysten käytettävissä. (Tableau 2022; Solita 2021). Knowitin (2020) tarjonnan mukaisesti Tableaun ohjelmisto koostuu neljästä osasta:

- Tableau Desktop on Tableaun työpöytäsovellus, jolla voidaan luoda interaktiivisia raportteja.
- Tableau Prep Builder on visuaalinen ohjelmisto datan yhdistelyyn, muokkaukseen sekä puhdistamiseen. Ei ole välttämätön ja raportit voidaan luoda myös käyttämättä tätä.
- Tableau Server mahdollistaa Tableau-raporttien jakamisen organisaation sisäisessä verkossa. Se voi toimia myös yleisimmissä IaaS-pilvissä, kuten Amazon web servicesissä tai Microsoft Azuressa.
- Tableau Online on Tableau Serveriä vastaava valmis pilvipalvelu, jossa Tableau raportteja voidaan jakaa. Työkalua ostettaessa tulee valita joko Tableau Server tai Tableau Online.

2.4.2 Python

Python on yksi tärkeimmistä sekä suosituimmista koodikielistä, jota käytetään datatieteissä, koneoppimisessa sekä yleisessä ohjelmistokehityksessä. Sillä on laaja ja aktiivinen tieteellinen yhteisö taustalla. Python on tulkattu ohjelmointikieli, joten pääsääntöisesti sen koodi on hitaampaa kuin käännetyillä ohjelmointikielillä, kuten C++:lla. Python on kuitenkin erinomainen alusta ohjelmoida, jos vaatimuksena ei ole erittäin lyhyt suoritusaika tai haastavat resurssien käyttöastevaatimukset (McKinney 2017, s.2–3). Python on avoimen lähdekoodin ohjelmointikieli, joten sen käytöstä ei tarvitse erikseen maksaa.

Data-analytiikan näkökulmasta Pythonilla on useita hyödyllisiä kirjastoja. NumPy (Numerical Python), tarjoaa työkalut datan järjestelyyn jonoiksi, sekä näiden välisten laskutoimitusten funktioita. Numpyn päälle rakennettu Pandas tarjoaa työkalut datan järjestelyyn, muokkaamiseen sekä aggregointiin tehokkaissa DataFrameissa (Vo.T.H et al. 2017, s.56). DataFramet ovat taulukkomuotoisia datan säilytysmuotoja, jotka ovat erittäin paljon käytettyjä datan käsittelyssä. Muita data-analytiikalle oleellisia kirjastoja ovat koneoppivia malleja sisältävä scikit-learn sekä suosituin datan visualisoinnin kirjasto matplotlib. (McKinney 2017, s.4–7)

Python mahdollistaa myös interaktiivisten selaimessa käytettävien applikaatioiden luomisen. Tätä varten on kehitetty kirjastoja, kuten Dash. Dash on avoimen lähdekoodin Python-kirjasto, jolla voidaan luoda interaktiivisia selaimessa käytettäviä applikaatioita datan visualisoimiseksi. Se mahdollistaa BI-työkalujen kaltaisten dashboardien luomisen, kuitenkin siten, että käyttäjän interaktiot pystyvät käynnistämään tietolähteessä huomattavasti monimutkaisempaa koodia kuin BI-työkalut pääsääntöisesti kykenevät. Dash mahdollistaa visualisoinnin huomattavasti vapaammin kuin BI-työkalut, joissa visualisointiin on rajattu määrä kuvaajatyylejä käytettävissä. Dash mahdollistaa taustalla olevan tietolähteen, kuten SQL-tietokantaa kuvaavan Dataframen muokkaamisen raportin selaamisen aikana, kun taas BI-työkaluissa taustalla oleva tietolähde pysyy usein muodoltaan staattisena. Dash toimii kolmen keskeisen teknologian avulla. Flask tarjoaa verkkopalvelintoiminnot, React.js renderöi käyttöliittymän verkkosivulle ja Plotly.js luo sovelluksessa käytettävät kuvaajat. Dashin on luonut ja sitä kehittää kanadalainen yritys Plotly (2017). Dashin lisäksi on olemassa muitakin selainapplikaatioita varten luotuja Python -kirjastoja. Näistä eräs mainittavan arvoinen on Streamlit, joka on vielä Dashiakin yksinkertaisemmalla syntaksilla varustettu kirjasto. Se kuitenkin soveltuu tällä hetkellä enemmän prototypointia varten, kuin varsinaisten tuotteiden tekemiseen (Hwang 2020).

2.4.3 Javascript

JavaScript on ohjelmointikieli, joka on yksi verkkosivustojen pääteknologioista HTML, CSS sekä DOM kanssa. Se mahdollistaa verkkosivustojen dynaamiset toiminnallisuudet ja sillä voi luoda datan visualisointeja verkkosivustoille sekä selainpohjaisiin käyttöliittymiin. Näitä interaktiivisia verkkosivuja kutsutaan usein myös verkkosovelluksiksi tai selainpohjaisiksi applikaatioiksi. (Mikkonen & Taivalsaari 2008; MDN Web Docs 2022) Pythonin tavoin myös JavaScript on ilmainen ohjelmointikieli.

JavaScript-kehyksillä (framework) tarkoitetaan JavaScriptin kirjastoja, joita on luotu mahdollistamaan modernien dynaamisten sekä interaktiivisten applikaatioiden tekeminen. Suosituimpia kehyksiä ovat esimerkiksi React, Angular, Vue sekä Ember (MDN Web Docs 2022). JavaScript-kehyksiä käytetään etenkin verkkosivustojen käyttöliittymissä, mutta nykyään myös useat työpöytäsovellukset on luotu käyttäen web-teknologioita. Tästä esimerkkinä on Spotify, jonka työpöytäsovelluksen käyttöliittymä on tehty samalla React -sovelluksella, kuin selaimessa käytettävä versio (Spotify 2021). JavaScript kehittyy muiden työkalujen tavoin nopeasti. Greif (2022) seuraa JavaScriptin kehitystä vuosittain tehtävillä kyselyillä ja vuonna 2022 kyselyyn vastasi 16 085 JavaScript kehittäjää. Kyselyjen perusteella tällä hetkellä React sekä Vue ovat eniten käytetyt JavaScript-kehykset. TypeScriptiä käytti 69 % vastaajista, kun sama luku oli vuonna 2016 21 %. TypeScript tarkoittaa JavaScriptin päälle tehtyä oliokeskeistä ohjelmointikieltä, eli se tukee luokkia, perimistä sekä kapselointia. Käytännössä TypeScript käännetään JavaScriptiksi, kun se suoritetaan selaimessa. TypeScript helpottaa kehittäjän työtä, koska se lisää kielen tyypityksen sekä syntaksin, joka auttaa huomaamaan virheet jo editorissa. (TypeScript 2022)

JavaScriptiä käytetään paljon datan visualisoinnissa, koska se mahdollistaa interaktiiviset sekä hyviin kustomoitavat selainpohjaiset applikaatiot. Näiden applikaatioiden kehitystyö vaatii kuitenkin kehittäjältä jonkin verran suunnitelmallisuutta, koska kaikki visualisointikirjastot eivät ole yhteensopivia kaikkien JavaScript-kehysten kanssa. Lisäksi visualisointityökalua kehittäessä tulee huomioida käytettyjen graafien toimivuus visualisoitavalle datalle. Keskeisimpiä interaktiivisten kuvaajien teknologioita ovat SVG

(Scalable Vector Graphics) sekä HTML5 Canvas. Näissä keskeisimpänä erona on se, että SVG tallentaa datan muuttujiin suorituksen ajaksi ja ei siten vaadi uudelleenpäivitystä interaktiivisiin toiminnallisuuksiin. Canvas puolestaan voi tulla käytännölliseksi, mikäli halutaan piirtää useita tuhansia datapisteitä. (MDM Web Docs 2022; Barchart 2019)

2.4.4 Muut visualisointityökalut

Datan visualisointiin on olemassa erittäin kattava työkalupakki ja monilla eri työkaluilla voidaan luoda vastaavia toteutuksia. Kaikkia työkaluja ei ole järkevää tutkimuksen aikarajoitteista johtuen käydä läpi, mutta esimerkiksi Chawla et al. (2018) sekä Ali et al. (2016) ovat listanneet näitä työkaluja tarkemmin. Datan käsittelyyn sekä visualisointiin on olemassa erittäin paljon maininnan arvoisia työkaluja, joita ei otettu tutkimuksessa tarkempaan tarkasteluun.

R on datatieteissä käytetty ohjelmointikieli, joka pitää sisällään kaikkiin data-analytiikan tehtäviin tarkoitetut kirjastot. R on kuitenkin pääasiassa soveltuva dataseteille, jotka voivat mahtua tietokoneen keskusmuistiin. R onkin soveltuvampi nopeaan prototyyppien luomiseen kuin laajaan kehitystyöhön (Prabhu et al. 2019, s.27).

Matlab on tieteessä sekä teollisuudessa paljon käytetty ohjelmisto sekä ohjelmointikieli. Se kehitettiin pääasiassa numeerista laskentaa varten. Matlabilla on kattava valikoima tieteellisiä sekä insinöörityössä käytettäviä visualisointeja. (Paluszek & Thomas 2021, s.101) Matlab tarjoaa myös osassa tuotteistaan käyttöliittymien sekä datan visualisoinnin työkaluja. Käyttöliittymien rakentaminen Matlabilla vaatii kuitenkin maksullisen lisenssin, eikä se ole yhtä laajasti käytettyjä kuin esimerkiksi BI-työkalut datan visualisoinnissa. (Matlab 2022)

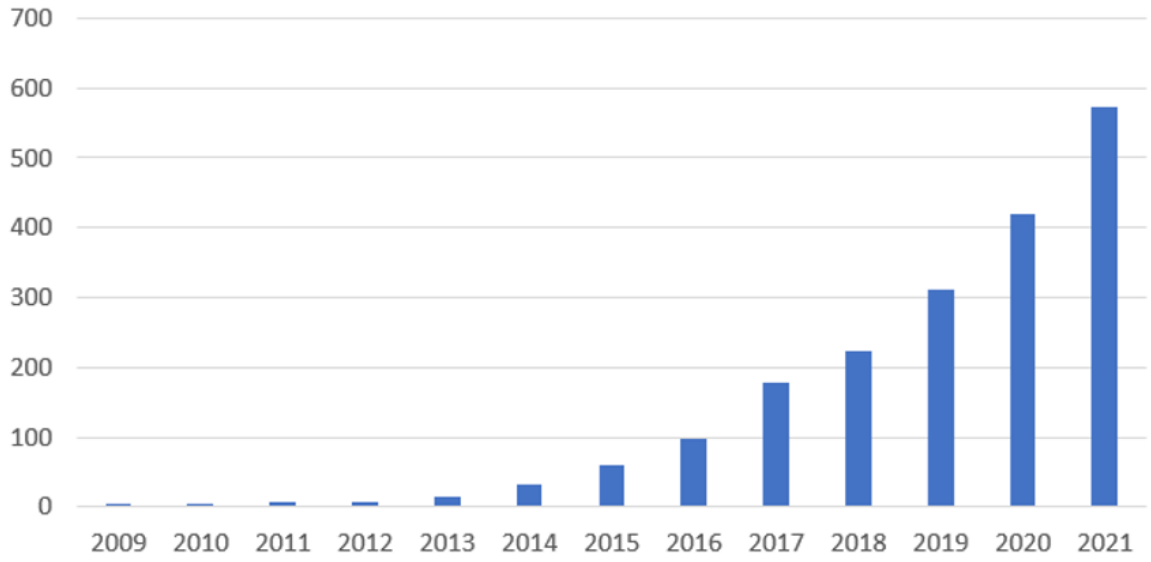
3 Data-analytiikka ilmailualalla

Tässä tutkimuksessa data-analytiikkaa lähestytään etenkin ilmailualan näkökulmasta. Richterin ja Waltherin (2016, s. 274–275) mukaan digitalisaatio on mullistamassa ilmailuteollisuuden arvoketjun. Uudet teknologiat mahdollistavat siirtymisen ennaltaehkäisevästä huollosta kohti ennustavaa huoltoa. Zhangin (2018, s. 14) mukaan lentokoneiden lentotietoainestoa on käytetty aiemmin esimerkiksi vianhakuun sekä diagnosointiin kaasuturbiineissa, lentokoneen rungon kuormituksen arvioinnissa, sekä ilman lentäjän komentoa tapahtuneiden liikkeiden tutkinnassa. Täten lennon ajalta tallennetun datan tehokkaampi hyödyntäminen voi luoda merkittäviäkin kustannussäästöjä sekä parannusta koneiden käyttöasteeseen niiden elinkaaren aikana. Kustannussäästöjen lisäksi yhtenä merkittävä motivaattorina toimii lentoturvallisuuden lisääminen.

Sekä siviili- että sotilasilmailun datan käsittelyssä keskeisessä osassa ovat MRO (Maintenance, Repair, Overhaul) ohjelmistot. MRO-ohjelmistoilla tallennetaan ja hallitaan dataa esimerkiksi lentokoneiden osista huoltoja varten. Ilmailualalla huoltosektori on eritäin monimutkainen kokonaisuus, jossa toimii välikäsiä sekä toimijoita, joiden täytyy jakaa dataa samalla varmistuen sen tietoturvasta. Toimitusketjut osien valmistajilta (Original Equipment Manufacturers, OEM) voivat olla pitkiä sekä monimutkaisia, joten ennustettavuudesta on erityistä hyötyä alalla (Efthymiou et al. 2022; Richter & Walter 2016).

3.1 Ilmailualan data-analytiikan kehitys

Kuvassa 7 on esitetty Chung et al. (2020) tutkimuksen mukaisesti, kuinka datatieteisiin sekä analytiikkaan liittyvien ilmailualan artikkeleiden määrä on ollut merkittävässä kasvussa viime vuosina. Julkaistujen artikkeleiden määrää Google Scholarissa on pidettävä vain suuntaa antavana, mutta ne selvästi osoittavat kuinka kiinnostus ilmailualan analytiikkaan on lisääntymässä ja että kyseessä on ajankohtainen sekä tärkeä aihe. Chung et.al. mukaan ilmailualan artikkeleissa esiintyy etenkin teemat Big Data, ennustaminen, koneoppiminen sekä ilmailun logistiikka.



Kuva 7. Ilmailualan datatieteeseen ja analytiikkaan liittyvien julkaisujen määrä. (Chung et al. (2020) mukaisesti haettu julkaisuja Google Scholarista hakusanoilla “data science”, “data analytics” ja “aviation”).

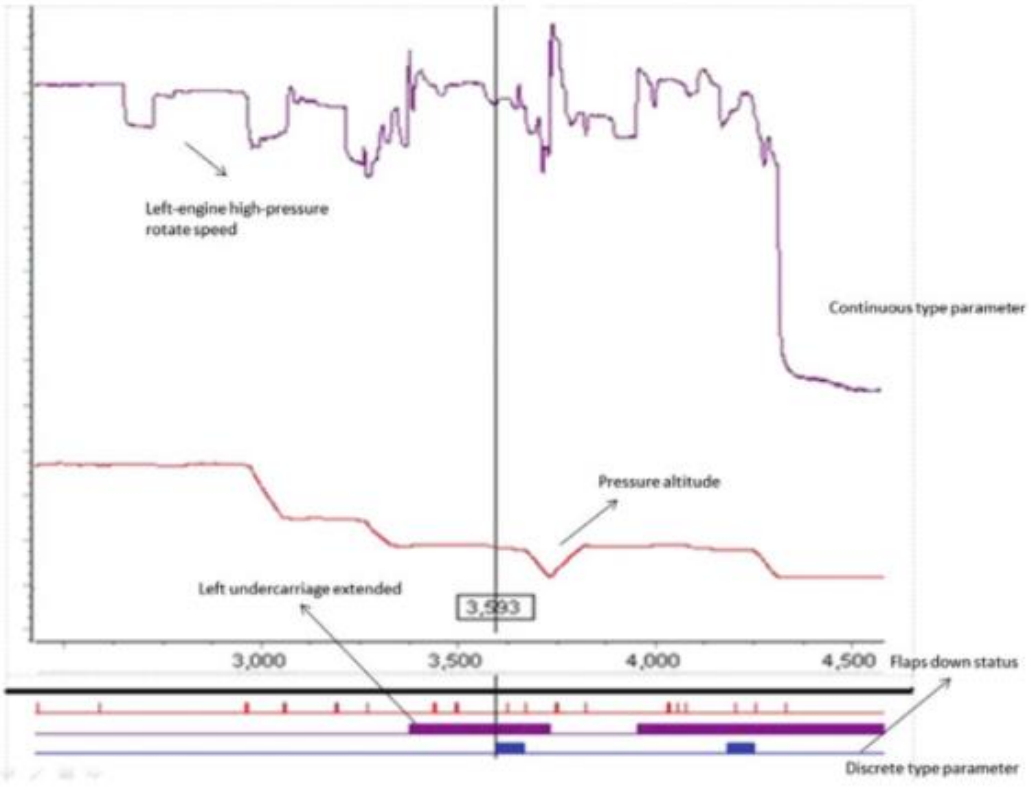
Eräs ilmailualan digitalisaation tutkimuksia kokoava taho on Digital Avionics Systems Conference (DASC). Julkaisuissa on tällä hetkellä esillä etenkin lisääntyvä autonomia ilmailun järjestelmissä. Tekoäly droneissa, miehittämättömissä ilma-aluksissa, suuremmissa ilma-aluksissa sekä näiden yhteydessä maassa sekä avaruudessa oleviin järjestelmiin vaikuttavat olevan tällä hetkellä kuumimpia tutkimuskohteita ilmailualalla. Etenkin uudet koneoppimista sisältävät teknologiat vaativat tutkimustyötä, jotta niiden toimivuudesta sekä turvallisuudesta voidaan varmistua. (DASC 2022)

3.2 Ilmailualan data ja lentotietoaineiston määritelmä

Ilmailualan data voi käsittää esimerkiksi lennonjohdon dataa, huoltodataa tai lennon aikana lentokoneen lentotietojärjestelmän tallentamaa lentotietoaineistoa. Lentotietoaineistot pitävät sisällään paljon aikasarjadataa, joka on tyypillinen datan tallennustapa esimerkiksi

säädatalle tai tuotantolaitteistolle (Wang et al. 2022). Se ei siis pääsääntöisesti vaikuta eroavan muusta teollisuuden prosessidatasta, joten datan käsittelyssä voidaan käyttää helposti myös muiden toimialojen työkaluja. Toisaalta lentodatassa ja erityisesti sotilasilmailun lentodatassa tyypillistä on, että kaksi saman lentokoneen tekemää lentoa eivät ole aikasarjaltaan samanlaisia, koska ulkoilman muuttujat, lentokoneen suorituskyky sekä lentotehtävät voivat vaihdella huomattavasti. (Zhang 2018, s.11-12)

Lentokoneiden lentotietojärjestelmä tallentaa tietoa lennolta pääsääntöisesti ennalta määrätyllä taajuudella. Zhangin (2018, s.11) mukaan lentotietoaineisto voidaan jakaa siten jatkuviin sekä diskreetteihin arvoihin. Kuvassa 8 on esitelty esimerkki jatkuvista ja diskreeteistä arvoista. Jatkuvat arvot muuttuvat jatkuvasti ja saavat yleensä mitä tahansa lukuja, joten esimerkiksi lentokorkeus on jatkuva arvo. Diskreetit arvot puolestaan saavat ennalta määrättyjä arvoja ja ovat yleensä binäärisiä eli ovat joko 1 tai 0. Esimerkiksi diskreetti arvo voi olla laskutelineiden asento, mutta diskreetit arvot voivat kuvata myös yksittäistä tapahtumaa.



Kuva 8. Jatkuvaa sekä diskreettiä lentodataa. Diskreetit parametrit kuvaajan alaosassa. (Zhang 2018, 11)

Zhangin (2018, s.12–13) mukaan lentoja voidaan arvioida lennon laadun sekä lentokoneen suorituskyvyn perusteella. Sotilaslentokoneiden lennon laatua arvioidaan eri tavalla kuin siviililentokoneita, koska ne lentävät monipuolisemmin. Sotilaslentokoiden lennoista on tunnistettavissa viisi eri lentotilaa, jotka ovat karkeasti suomennettuna; kiertely, syöksy, silmukka, kallistuminen sekä tasainen lento. Näiden lentotilojen tunnistaminen on oleellista, kun halutaan vertailla lennon aikaisia tapahtumia keskenään. Lentokoneen suorituskyvyn arviointi puolestaan perustuu aerodynaamisten parametrien sekä lentokoneen lennonohjausjärjestelmän toiminnan arviointiin. Mitattuja arvoja voidaan esimerkiksi verrata tuulitunnelianalyyseihin ja ennalta tehtyihin ennusteisiin.

3.3 Sotilasilmailun näkökulma

Suomessa ilmavoimien päälentokaluston muodostavat Boeing F/A-18 C/D Hornet -monitoimihävittäjät. Näiden lisäksi käytössä on koulutus-, kuljetus- ja yhteyskonekalustoa, joita on esitetty kuvassa 9. (Puolustusvoimat 2022)

Kuva 9. Suomen puolustusvoimien lentokalustoa. Vasemmalta ylhäältä alkaen Valmet L-70 Vinka, Grob G 115E, BAE Systems Hawk sekä Boeing F/A-18 Hornet (Puolustusvoimat 2022)

Sotilasilmailun analytiikkaratkaisuista on siviili-ilmailua vaikeampi löytää julkaistua tietoa, koska alalla käsiteltävät asiat ovat useammin salattuja. Kuitenkin myös sotilasilmailusta on mahdollista löytää ajankotaisia lähteitä. Eräs uusimmista sotilasilmailun analytiikkaratkaisuja käsittelevistä tieteellisistä artikkeleista on Augustinin et al. (2021) julkaisu. Siinä rakennetaan H-60 Black Hawk helikoptereiden datalle pilvipohjainen työkalu, jolla eri lähdejärjestelmistä peräisin olevaa dataa voidaan analysoida kootummin ja tehokkaammin. Visualisointi on nähtävissä kuvassa 10. Tutkimuksen päätelminä oli, että sotilasilmailun asealustoilla on paljon datalähteitä, kuten huolto- ja lentotietoja. Näiden datojen esittämisessä yhtenäisissä sekä helppokäyttöisissä dashboardeissa antaa erittäin paljon lisäarvoa. Lisäarvon mahdollistaa etenkin eri datalähteiden aikasarjojen automaattinen yhdistäminen sekä korjaus, algoritmien käyttäminen datan eheytyksessä sekä asennettujen komponenttien käytön seuranta huollon tukena. Tämä mahdollistaa kokonaiskuvan luomisen jokaisen runkoyksilön käytöstä sekä huollosta. Toisaalta myöskään tässä käyttöliittymässä ei vaikuta olevan mahdollisuutta analysoida lentotietoaineistoja kootummin yksittäisten alusten tai koko laivueen historian kautta.

Kuva 10. Esimerkki eri lähteissä olevien huolto- ja lentotietojen yhdistämisen pohjalta tehdystä dashboardista. Rakennettu käyttäen Pythonia ja JavaScriptin D3.js kehystä. (Augustin et al. 2021)

Ilmailualalla varaosien ja muun materiaalin varastoinnin sekä hankinnan suunnittelu on perinteisesti perustunut kulutusdataan. Sotilasilmailussa on kuitenkin viime vuosina pyritty siirtymään tilannepohjaisesta huollosta kohti ennustavaa huoltoa. Esimerkiksi Hao et al. (2020) luomassa sotilaslentokoneiden huollon optimoinnin menetelmässä käytetään lentotietoihin perustuvaa osien vikatiheyttä. Tietoja yhdistämällä tulevien lentotehtävien profiileihin, voidaan varaosien varastotasoa optimoida niin, että lentokoneiden käytettävyysastetta pystytään paremmin hallitsemaan. Vastaavaa sotilasilmailun ennustavaa huoltoa ovat kehittäneet myös Bayoumi ja Matthews (2020). Myös Suomessa on tutkittu koneoppivien mallien käyttöä F/A-18:n lentotietoaineistolla huoltotarpeiden ennustamiseksi (Parts et al. 2017 s. 163–165). Tutkimustulokset ovat osoittaneet, että koneoppivat mallit ja muut analytiikkaratkaisut voivat mahdollistaa huoltotarpeiden tarkemman ennustamisen.

Lentotietoaineistoa käytetään myös lennon aikana tapahtuneiden lentokohtaisten G-arvojen ylityksien seurannassa sekä väsymisindeksin (Fatigue Index, FI) laskemisessa. FI-arvojen seuranta ei kuitenkaan ota huomioon turbulenttisen ilmavirran aiheuttamaa väsymistä, joka on pääasiallinen rakenteiden väsymisen aiheuttaja. Tätä varten FI-arvojen seurannan lisäksi

on kehitetty muita lentokonekohtaisia seurantatyökaluja. Esimerkiksi F/A-18 Hornetin osille on kehitetty ainakin jonkinlainen käyttöliittymä väsymisdatan tarkasteluun, mutta monet aihetta tarkemmin käsittelevistä julkaisuista ovat salattuja. (Viitanen & Siljander 2021, s.12, 41, 71)

Taulukko 2. Viime vuosien julkaisuja, joissa käsitellään sotilasilmailun lentotietoaineiston analytiikkaa.

Julkaisun nimi	**Lähde**	**Tutkimuksen keskeinen teema**
Time Series Analysis Methods and Applications for Flight Data (Kirja)	Zhang 2018	Lentotietoaineiston käsittely
Boiling down aviation data: Development of the aviation data distillery	Augustin et al. 2021	Lentotietoaineistojen yhdistäminen yhdelle koontinäytölle
A Data-based Expert System for Aero-Engine Gas Path Fault Diagnosis	Sun et al. 2021	Työkalu suihkumoottorin vianselvitykseen
Prediction of the fatigue lifetime of the Portuguese Air Force Epsilon TB-30 aircraft	Barros et al. 2020	Lentotehtävien väsymisvaurioiden mittaaminen ja sen vaikutus huollon suunnitteluun.
Method for Optimising Mission-Specific Inventory of Aviation Materials	Hao et al. 2020	Huollon optimointi.
Condition-based maintenance to predictive maintenance: a use case on selected USARMY military aircraft	Bayoumi & Matthews 2020.	Huollon optimointi.
A Review of Aeronautical Fatigue Investigations in Finland April 2019 - April 2021	Viitanen & Siljander 2021	Väsymisvaurioiden analysointi.
S4Fleet – Service Solutions for Fleet Management	Parts et al. 2017 s. 163-165	F/A-18 huollon optimointi.

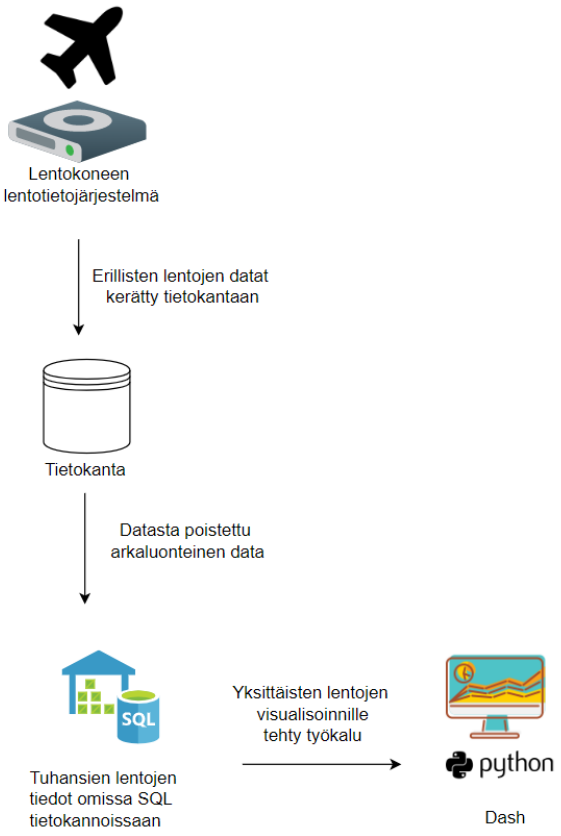
Taulukkoon 2 on koottu viime vuosina julkaistuja sotilasilmailun analytiikkaan liittyviä julkaisuja. Keskeisinä teemoina julkaisuissa toistuvat kustannussäästöjen haku huollon optimoimisella, lentojen turvallisuuden parantaminen sekä paremman tilannekuvan luominen eri datalähteiden yhdistämisellä. Kaikkiaan ilmailualan data-analytiikka vaikuttaa siis ainakin ei-lennonaikaisen analytiikan osalta kamppailevan yhä samojen keskeisten haasteiden kanssa, kuin mitä on nähtävissä muualla yritysmaailmassa. Dataa on paljon, mutta se on hajautettuna eri lähteissä ja sen yhdistämiseen, analysoimiseen sekä käyttäjäläheiseen visualisoimiseen on yhä lisääntyvää kysyntää.

4 Visualisointityökalun valinta diplomityöhön

Tässä luvussa esitellään diplomityön empiirisen osuuden tausta ja sen toteutustapa. Luvussa esitellään visualisointityökalua varten tehtyjen haastattelujen tulokset sekä niiden pohjalta luodut käyttötapaukset visualisointityökalun toiminnallisuuksille.

4.1 Kohdeyrityksen esittely sekä nykyiset toteutukset datan visualisoinnille

Kohdeyritys on kansainvälinen puolustus-, turvallisuus- ja ilmailualan elinkaaren tukipalvelujen, lentokoulutuksen ja teknologia- ja ohjelmistoratkaisujen tuottaja. Kohdeyritys on toiminut alalla jo pitkään ja sillä on kattavaa yrityksen sisäistä alan osaamista.



Kuva 11. Diplomityön aloitushetken datan jalostuksen prosessi yksinkertaistettuna.

Tällä hetkellä lentotietoaineistoa käsitelläkseen loppukäyttäjä käyttää erillistä järjestelmää, jossa data haetaan datalähteestä yksittäisinä lentotiedostoina. Näitä lentotiedostoja on visualisoitu ja analysoitu esimerkiksi Excelissä. Lisäksi kohdeyritys on tehnyt jo alustavan toteutuksen Pythonin Dashilla, jossa lentotietoaineistoa pystytään tarkastelemaan yksittäisten lentojen osalta. Tätä työkalua ei kuitenkaan ole viety loppukäyttäjälle tuotantokäyttöön asti. Kuvassa 11 on esitetty työn alkuhetken tilanne. Nyt visualisoitavaa dataa on suodatettu sen verran, että siitä on poistettu paikkatietoja ja asejärjestelmiin liittyviä tietoja. Alkutilanteen Dash -visualisointityökalu toimii mikropalveluarkkitehtuurin mukaisesti omassa Docker -kontissa. Nykyisestä toteutuksesta sekä työssä tehdystä jatkokehityksestä kerrotaan tarkemmin luvussa 5.

4.2 Haastattelujen toteutus

Sommervillen (2016, s.101-111) mukaan ohjelmistojen vaatimusmäärittelyssä vaatimukset voidaan jakaa karkeasti joko toiminnallisiin ja ei-toiminnallisiin vaatimuksiin. Toiminnalliset vaatimukset tarkoittavat toiminnallisuuksia, joita ohjelmiston tulisi pystyä tekemään. Tässä tutkimuksessa toiminnallisia vaatimuksia selvitettiin loppukäyttäjien tarpeisiin perehtyneille asiantuntijoille tehdyillä ryhmähaastatteluilla. Ei-toiminnalliset vaatimukset puolestaan eivät suoraan liity käyttäjän ohjelmistolle antamiin vaatimuksiin, vaan ovat järjestelmään liittyviä vaatimuksia. Esimerkiksi suorituskykyyn ja järjestelmän tietoturvaan liittyvät vaatimukset ovat ei-toiminnallisia vaatimuksia. Ei-toiminnallisia vaatimuksia saatiin kohdeyrityksen ohjelmistokehittäjiltä, jotka ovat kehittäneet analytiikkaratkaisua, jossa visualisointityökalu tulee toimimaan. Ei-toiminnallisilla vaatimuksilla haettiin vastausta siihen, mikä visualisointityökalu sopii parhaiten lentotietoaineiston visualisointiin juuri kohdeorganisaation toimintaympäristössä. Haastattelut on toteutettu puolistrukturoituina haastatteluina yksittäisille henkilöille sekä teemahaastatteluina ryhmähaastattelun muodossa.

4.3 Visualisointityökalun toiminnallisten vaatimusten määrittely

Tutkimus aloitettiin selvittämällä loppukäyttäjien toiveita sekä ideointia asiantuntijoille tehdyillä ryhmähaastatteluna. Haastattelut keskitettiin yhdelle päivälle, jossa haastateltiin kolme eri asiantuntijaryhmää. Kaikille osallistujille annettiin esitiedoksi esitettävä aihe ja mitä siitä halutaan selvittää. Taulukossa 3 on esitetty tehtyjen haastattelujen ajankohdat, osallistujien määrät ja pääpiirteittäin mitä rooleja osallistujilla on ollut. Kaikki haastattelut olivat 1,5 h pituisia. Haastattelujen tavoitteena oli selvittää sovelluksen priorisoidut käyttötapaukset diplomityössä toteutettavaa laajuutta varten.

Taulukko 3. Asiantuntijoiden haastattelujen osallistujamäärä ja osallistujien roolit.

Haastattelu	**Osallistujamäärä**	**Osallistujien roolit**
Koulutuskonetiimi Toteutettu 31.1.2022	6	2 x Manager (Air) 1 x Manager (Data & Digital) 1 x Systems Architect 1 x Project Manager
Tehtävätuki Toteutettu 31.1.2022	10	3 x Manager (Air) 1 x Manager (Data & Digital) 3 x Manager (Fleet Availability) 3 x Software/System Engineers
Lentokonehuolto Toteutettu 31.1.2022	7	3 x Manager (Air) 1 x Manager (Data & Digital) 3 x Manager (Fleet Availability)

Haastattelujen teemana oli lentotietoaineiston hyödyntämismahdollisuudet data-analytiikan avulla sekä dataan perustuvan visualisointityökalun ideointi. Kaikki haastattelut olivat samanlaisia avoimia haastatteluja. Ensin esiteltiin aihe sekä käytiin läpi mitä on aiemmin

toteutettu datalla. Tämän jälkeen keskusteltiin aiemmasta toteutuksesta ja kerättiin ideoita ryhmiltä tämän diplomityön empiirisen osuuden jatkokehitystä varten. Yleisesti ottaen haastattelut olivat samankaltaisia, mutta jokaisessa tunnistettiin toisista haastatteluista poikkeavia ideoita. Osa haastateltavista oli useammassa haastattelussa mukana.

Koulutuskonetiimille tehdyssä haastattelussa esiin nousi etenkin lentokoneen käytön yhdistäminen vikaantumisiin, kuten sen selvittäminen millaiset lennot aiheuttavat mitäkin vikaantumisia. Lisäksi esiin nousi esitettävän lentotiedon parametroitavuus esimerkiksi ohjelmistoversion tai laitteiden käytettävyyden avulla. Tietty laite voi aiheuttaa vikoja useamman eri lentokoneen runkoyksilössä. Laitenumeron yhdistäminen kuvaajiin saattaisi auttaa tunnistamaan vikaantuvat laitteet sekä näin auttaa niiden uusimisessa.

Tehtävätukitiimille tehdyssä haastattelussa ideointia esiintyi etenkin lentokoneiden huollon kehittämiseksi datan avulla. Sen avulla pystyttäisiin esimerkiksi ennakoimaan huoltotehtäviä ja niiden kestoja paremmin sekä säätämään huoltojärjestelmää paremmaksi. Käytännössä jos laitteessa on ollut vikailmoitus, datan avulla voitaisiin selvittää tarkemmin, millaisessa tilanteessa vika on tullut. Lisäksi lentäjän osuuden selvittäminen vikaantumisissa herätti ajatuksia, koska esimerkiksi osa lentäjistä saattaa kirjata ongelmat herkemmin. Lentäjätieto ei kuitenkaan ollut saatavilla tässä datasetissä, ja vaatisi erityistä pohdintaa henkilötietojen käsittelyn osalta. Haastatteluissa myös nousi esiin trendien tunnistamisen tärkeys.

Lentokonehuollon tiimille tehdyssä haastattelussa ideointia oli esimerkiksi, kuinka eri lentotehtävät kuormittavat lentoja ja kuinka kuormittavuudesta voitaisiin estimoida laitteiston elinikää. Lisäksi kustannusten yhdistäminen huoltoihin ja materiaaleihin nähtiin potentiaaliseksi mahdollisuudeksi. Myös tietojen yhdistäminen esimerkiksi huollon ja logistiikan suunnitteluun nousi ideana esille. Työkalusta olisi hyötyä, jos siitä pystyisi näkemään suoraan lentodataan perustuvat huoltoehdotukset sekä mahdollisesti myös arvion euromääräisistä säästöistä. Lisäksi säätiedoista saatava lisäinformaatio nousi keskustelussa esille. Yhdistämällä lentoalueella oleva säätieto lentotietoaineistoon, pystyttäisiin saamaan lisää muuttujia mukaan analyyseihin sekä ongelmanselvitykseen.

Yhteenvetona asiantuntijahaastatteluista luotiin loppukäyttäjiä mahdollisimman hyvin palvelevat käyttötapaukset, jotka esitellään luvussa 4.4. Lisäksi haastattelussa nousi esiin, että koko tuoteputki pitää olla valmiina, kun lopullista visualisointityökalua esitellään tai tuodaan loppukäyttäjälle. Eli koko prosessi datan keräämisestä, analyysityökalun asentamisesta sekä käyttämisestä tulisi olla mietitty. Loppukäyttäjälle pitäisi jäädä vain vähän hoidettavia asioita, kuten datan mahdollinen päivitys ajoittain, mikäli ratkaisu sitä vaatii.

Osa haastatteluissa esiin nousseista ajatuksista vaatisi muista tietojärjestelmistä olevan datan hakua sekä yhdistämistä nyt jo valmiiksi kerättyihin lentotietoaineistoihin. Tällaisia ideoita olivat esimerkiksi huoltotehtävien keskimääräisen keston, lentäjätiedon tai lentokentän säätiedon yhdistäminen analyyseihin. Osa näistä tiedoista, kuten säätieto on helpommin saatavilla, kun taas osa mahdollisesti kiinnostavista tiedoista olisi haastavampaa yhdistää lentotietoaineistoon ainakin tämän diplomityön yhteydessä.

4.4 Työkalulle halutut käyttötapaukset

Käyttötapaukset on selvitetty kohdeyrityksen asiantuntijoille tehdyillä ryhmähaastatteluilla, jotka on esitetty luvussa 4.3. Ryhmähaastatteluissa esiin nousseet sekä tähän työhön valitut käyttötapaukset visualisointityökalun toiminnallisuuksista olivat:

1. Mahdollisuus skaalata ja siirtyä helposti yhdestä koneesta koko laivueen historiaan, ja mahdollisuus selvittää koko laivueen osalta tapahtumia sekä niiden frekvenssiä. Nykyisin käytössä olevilla työkaluillakin jokaisen lennon tiedot ovat kyllä haettavissa, mutta dataa ei saa helposti koostetussa / aggregoidussa muodossa.

2. Lentoparametrien tutkimista auttava helppokäyttöinen kuvaajien plottaus, suodatus ja haku: kuvaajaan selkeästi esitettynä datapisteet, joilla valittu parametri ylittää/alittaa annetun haku- tai suodatuskriteerin. Esimerkkinä: lentokorkeus on ollut yli tietyn korkeuden tai kun kone on laskeutumassa tietyllä nopeudella.

3. Säätiedon yhdistäminen ja tuominen osaksi lentotiedon analyysia nousujen ja laskujen osalta. Esimerkki: näytetään nousu ja/tai laskuhetken olennaisimmat säätiedot lentokentällä.

Visualisointityökalun edistyneempi lentotietoaineiston käsittely helpottaa laitteiden vikaantumisten syiden selvityksessä. Tämä puolestaan voi auttaa optimoimaan lentokoneiden käyttöä aina niiden elinkaaren loppuun asti, joka puolestaan näkyy suurempana koneiden käytettävyysprosenttina sekä huoltokulujen vähentymisenä. Loppukäyttäjien palautteen mukaisesti visualisointi auttaa hahmotuksessa verrattuna lukuihin Excelissä. Valittavan visualisointityökalun pitää olla mahdollisimman yhteensopiva myöhemmin lisättäviin koneoppimisratkaisuihin. Lisäksi sen tulee olla skaalattavissa myös laajempiin tietoaineistoihin tulevaisuudessa.

4.5 Visualisointityökalun ei-toiminnallisten vaatimusten määrittely

Kun kohdeyrityksen asiantuntijoille oli tehty haastattelut visualisointityökalulle halutuista käyttötapauksista, haastateltiin seuraavaksi kohdeyrityksen ohjelmistokehittäjiä ei-toiminnallisten vaatimusten tunnistamiseksi. Haastattelujen tavoitteena oli selvittää eri visualisointityökalujen kokemuksia ja soveltuvuutta kohdeyrityksen toimintaympäristöön. Taulukon 4 haastatteluissa käytetty haastattelurunko on nähtävissä liitteessä 1.

Taulukko 4. Ohjelmistokehittäjien haastattelujen osallistujamäärä ja osallistujien roolit.

Haastattelu	**Osallistujamäärä**	**Osallistujat**	**Aihe**
Ryhmähaastattelu 7.2.2022	3	Data Scientist A Software Engineer Software Architect	BI-työkalujen soveltuvuus
Puolistrukturoidut haastattelut 25.2.2022 4.3.2022	2 x 1	Data Scientist A Data Scientist B	Python Dash
Puolistrukturoidut haastattelut 22.3.2022 24.3.2022	2 x 1	Software Engineer Software Architect	JavaScript Vue

4.5.1 BI-Työkalujen haastattelu

BI-työkalujen soveltuvuuden haastattelussa aihetta lähestyttiin Tableaun soveltuvuutena datan visualisoinnissa. Haastattelu toteutettiin avoimena ryhmähaastatteluna, johon osallistui ohjelmistokehityksessä mukana olleita ja Tableauta aiemmin käyttäneitä henkilöitä. Kohdeyrityksessä oli kokemusta Tableaun käytöstä visualisoinnissa ja ainakin yksi käytännön toteutus oli toteutettu kohdeyritykseen Tableaulla, mutta se ei ole enää käytössä nykyisissä ratkaisuissa. Tässä aiemmin tehdyssä kokeilussa oli käytössä samankaltainen datasetti kuin nyt toteutetussa diplomityössä.

Tableaun etuna nähtiin BI-työkaluista yleisesti valmiina löytyvät pääsynhallinnan työkalut sekä mobiilituki. Tableaun nähtiin toimivan Dashboardien kanssa, joissa dataa interaktiivisesti selataan ja suodatetaan. Tietolähteenä tulisi kuitenkin olla staattinen tietokanta ja esimerkiksi koneoppivat ratkaisut tulisi tehdä erikseen backendissa. Jos visualisointityökalun vaatimukset rajoittuvat siihen, että työkalussa valmiiksi olevat

kuvaajat riittävät ja data on valmiiksi SQL-muodossa, niin silloin Tableau voi olla hyvä sekä toimiva myös isoillakin datamäärillä.

Tableaun kokeilussa vuonna 2020 oli huomattu, että se ei soveltunut kovin hyvin kaksisuuntaiseen liikehdintään, jossa raportilla tehdyt valinnat suorittavat koodia ja palvelimen ulkoisia kyselyitä taustalla. Tämä havaittiin heikkoudeksi, koska kohdeyrityksessä tehtävät datan visualisoinnit vaativat usein taustalla olevan datan käsittelyä tai muun koodin suorittamista. Myös Tableaun markkinoitu yhteensopivuus Pythonille oli osoittautunut rajatuksi. Pythonkoodia pystyi ajamaan lokaalisti ja se pystyi ajamaan ulos joko syötettyjä arvoja vastaavan listan olioita tai yhden luvun. Tämä yhteensopivuus soveltuisi laskutoimituksiin, kuten summauksiin ja keskiarvoihin, mutta esimerkiksi koneoppimismallien ajaminen tapahtuisi muistissa ja olisi hidasta. Koneoppimismalli tulisi siis ajaa omana erillisenä palvelunaan. Kuitenkin suurimmaksi heikkoudeksi haastattelussa tunnistettiin Tableun rajoitteisuus hyvin kustomoitujen visualisointiratkaisujen tekemisessä. BI-työkaluista löytyy toki omat sisäiset kielensä, mutta niiden sekä monimutkaisten raporttien luominen nähtiin epäedulliseksi, koska esimerkiksi koodia ja kokonaisuuden toteutusta ei pysty helposti löytämään sekä tarkastelemaan.

Tableaun heikkoudet nähtiin perinteisenä ”low code ongelmana”, jossa helpot asiat ovat helppoja, mutta kun mentiin vähinkin sivumpaan niin toteutuksesta tuli vaikeampaa. Esimerkiksi annettiin eräs korrelaatiomatriisi, jonka tekeminen vaati monia työvaiheita Tableaulla, kun saman olisi voinut toteuttaa yhdellä metodilla Pythonin Pandas-taulussa. Yhteenvetokommentiksi Tablaun soveltuvuudesta visualisointiin kuitenkin kerrottiin:

> *”Hyvä työkalu tietokannan päällä, joka voi myös olla reaaliaikaisesti päivittyvä. Ei mahdollista suoraan reaaliaikaisia kyselyitä, jossa lasketaan haastavampia juttuja (algoritmeja). Toimiva kunhan (loppukäyttäjälle)riittää (Tableaussa jo valmiina) käytössä olevat kuvaajat. Oli huono (semanttiseen) tekstihakuun. Pääsynhallinnat löytyvät työkalusta jo valmiiksi.”*
>
> *– Data Scientist A*

4.5.2 Python Dash haastattelut

Pythonin haastatteluissa haastateltiin kahta eri yrityksen sisäistä kehittäjää, joilla molemmilla oli kokemusta visualisoinnin toteutuksesta Pythonilla ja etenkin diplomityössä tarkasteluun otetusta Pythonin kirjasto Dashilla. Haastateltavat ovat toteuttaneet useita eri kokoluokan datan visualisointeja Dashilla. Lisäksi haastateltavilla on vaihteleva määrä kokemusta muista datan visualisonnin työkaluista, kuten Matlabista sekä Tableausta.

Haastatteluissa nousi esiin se, että Dash on vahvimmillaan juuri keskitason visualisointityökaluna. Sillä voi luoda reaktiivisia web-sovelluksia ilman syvällistä tuntemusta nykypäivän fullstack web-kehityksestä ja siten se onkin enemmän suunnattu data scientistin työkaluksi. Dash on 100 % Pythonia, joten kehittäjä ei joudu miettimään muita kieliä. Esimerkiksi web-sovellusten HTML rakenne luodaan Pythonin luokilla, joten sillä voidaan liikutella komponentteja kätevästi. Visualisointien kehittämistä kuvailtiin muutekin kokonaisuudessaan näppäräksi sekä sen käyttöä mukavaksi. Vahvimmillaan Dashin mainittiin olevan, kun luodaan kuvaajapohjaisia Dashboard -ratkaisuja, jotka keskustelevat palvelimen kanssa ja listailevat asioita. Lisäksi esimerkiksi Tableauhin nähden suurin etu oli saatavilla oleva tuki sekä muokattavuus, ja koneoppimisratkaisujen kannalta ei ollut nähtävissä mitään suoria esteitä.

Heikkoutena Dashille nähtiin se, että se ei sovellu hyvin web-sivuston tyyliseen ratkaisuun. Enemmän web-sivuston suuntaan menevää työkalua luodessa Dashista tulee yllättävän hajoamisherkkä ja siitä on laajennettaessa helppo tehdä bugista. Koska Dashilla kehitetty visualisointityökalu tehdään kokonaan Pythonilla, se voi tehdä koodin organisoinnista haastavampaa. Perinteisesti tehdyssä toteutuksessa, jossa JavaScript tekee frontendin, HTML määrittelee rakenteen, CSS tyylin ja Python vastaa backendista, osa-alueet voivat olla selkeämmin jaettu. Dashilla etenkin pidempi kehittäminen voi olla työlästä, mikäli koodin rakennetta ei pidä kurissa. Lisäksi pienet visualisointien viilaukset ja tyylittelyt saattavat olla haastavia ja aikaa vieviä, jos ei ole pitkää kokemusta Dashin käytöstä. Mikäli halutaan vain nopeasti esittää dataa kuvaajilla ja vain tarkastella dataa, Dashin visualisoinnin pystyttäminen nähtiin jonkin verran työläämpänä kuin esimerkiksi Matlabin käyttäminen.

> ”*Jos haluaa visualisoida yhden kuvaajan ei ehkä ole oikea työkalu. Itsestä tuntuu, että visualisoinnin pystyyn saaminen on jo hieman haastavaa. Pitää olla enemmän sisältöä (mitä halutaan visualisoida) ja enemmän kuin yksi kuvaaja.” - Data Scientist B*

Dashin rajoitteina oli havaittu se, että yksinkertaisetkin asiat saattavat kestää kauan, koska data pitää hakea usein backendistä. Tämä korostuu suurien datasettien kanssa, joissa pahimmillaan käyttäjä voi laukaista takaisinkutsun (Callback) useasti klikkailemalla nappia. Lisäksi monimutkaisissa sovelluksissa haasteeksi saattaa muodostua Dashin takaisinkutsujen toteutustapa, jossa eri takaisinkutsuilla ei voi olla samaa tulostetta. Käytännössä siis jos käyttäjän syöttämät arvot päivittävät useita muita kuvaajia, voi toteutustavasta tulla monimutkainen ja haastava. Kuitenkin yhteenvetokommentti Dashista oli:

> *”Dash on 100 % Pythonia, eli etenkin jos haluaa tehdä jotain yksinkertaista, niin on hyvä. Ei välttämättä tue monimutkaisempia sovelluksia. Soveltuu parhaiten kuvaajapohjaisiin Dashboard ratkaisuihin, mutta tavanomaisten web-sivun suuntaisissa ratkaisuissa oli yllättävän hajoamisherkkä. Laajennettaessa herkkä bugeille (vaatii tarkkuutta).” – Data Scientist A*

4.5.3 Javascript haastattelut

JavaScriptin haastatteluissa haastateltiin kahta yrityksen sisäistä kehittäjää, joilla molemmilla oli kokemusta visualisoinnin toteutuksesta JavaScriptillä. Haastateltavilla oli useamman vuoden kokemus JavaScriptin käytöstä datan visualisoinnissa. Haastateltavat ovat käyttäneet useita JavaScript-kehyksiä, kuten Vuea, Reactia, Angularia sekä D3.js -kirjastoa.

Haastatteluissa alkuun nousi esiin eri Javascript-kehysten erot. Tärkeimmäksi asiaksi eri kehysten eroissa nousi se, ettei ole olemassa selkeästi parasta vaihtoehtoa vaan valinnassa painottuu kehittäjien omat mieltymykset sekä kokemukset. Kuitenkin tärkeä huomioitava asia oli siinä, että valitun kehyksen tulee tukea haluttuja visualisointikirjastoja. Esimerkiksi Apex Charts on visualisointikirjasto, joka on toiminut hyvin eri kehysten kanssa. Toisaalta esimerkiksi Plotlyn visualisointikirjastoa ei ole onnistuttu saamaan toimimaan täysin interaktiivisesti Vuessa, joka on valitettavaa Plotlyn hyvien visualisointien takia. Tähän diplomityöhön tarkempaan tarkasteluun otetusta Vuesta ei haastateltavilla ollut kehysten vertailua ajatellen kritisoitavaa. Vue antaa kehittäjälle mahdollisuuden valita, millä alikirjastolla toteuttaa asioita, eikä se liiaksi rajoita käytettäviä kirjastoja. Vue on kehittynyt ohjelmointikehys, jolla on nopea tehdä kehitystyötä. Toisaalta samat asiat ovat myös toteutettavissa muilla kehyksillä. Lisäksi kohdeyritysten toteutusten näkökulmasta etenkin edellä mainittu Plotlyn yhdistäminen toimivasti sekä interaktiivisesti JavaScript-toteutukseen oli herännyt kiinnostavaksi selvitettäväksi asiaksi.

> *”Ei pahaa sanottavaa Vuesta. Saa itse valita millä alikirjastolla toteuttaa asioita versus se, että framework sanelee millä mennään. Kehittynyt framework jolla on nopea tehdä kaikenlaista.” - Software Architect*

Etuna esimerkiksi Pythonin Dashiin verrattuna nähtiin se, että käyttämällä JavaScriptiä poistetaan ylimääräinen kerros visualisoinnin toteutuksesta. JavaScript on web-käyttöliittymien toteutuksen alkuperäinen ympäristö ja Dash tuottaa HTML/JavaScript -koodia toteutuksessa. Kirjoittamalla JavaScriptiä suoraan eduksi nousee ainakin haastateltaville itselleen helpompi tyylittely sekä valmiit kehittäjien työkalut. JavaScriptiä pystyy helpommin kirjoittamaan itse, eikä se ole riippuvainen Dashin kehittäjien luomista yhteensopivuuksista. JavaScript on frontend -kehityksessä suurin hallitseva kieli. Interaktiivisten web-käyttöliittymien rakentaminen voi olla täten mielekkäänpää. Teknisesti Pythonilla voi tehdä samat asiat kuin JavaScriptillä, mutta toisaalta se voi vaatia suoraan JavaScriptin ja HTML kirjoittamista Pythonin sisällä. Käyttöliittymän muutostoiveiden toteuttaminen nähdään helppona, kunhan toteutuksen koodi on modularisoitua sekä rakenteeltaan toimivaa. JavaScriptillä toteutettu käyttöliittymä nähtiin toimivaksi tavaksi

toimittaa loppukäyttäjälle muun sovelluksen kanssa. Lisäksi mikäli loppukäyttäjä käyttää sovellusta verkkoyhteyden kautta, eivät muutokset tarvitse erityistä päivitystä, vaan ne ovat nähtävissä suoraan.

JavaScriptillä tehdyissä toteutuksissa etuna nähtiin myös projektinhallinnan näkökulma. Frontend-kehittäjiä löytyy työmarkkinoilta helposti, jos tulee sairastumisia tai muusta syystä pulaa työvoimasta. Pääsääntöisesti frontend-kehittäjät pystyvät myös työskentelemään eri JavaScriptin ohjelmistokehysten välillä, kunhan JavaScript -osaaminen on pohjalla. Esimerkiksi Python on frontend-kehityksessä huomattavasti pienemmässä roolissa, joten Pythonin Dashilla tehdyn visualisoinnin kanssa ei ole niin paljoa valinnanvaraa resurssien osalta mahdollisen ylimääräisen työvoimatarpeen tilanteessa. Kuitenkin toteutusta suunnitellessa tulisi katsoa oman kehitystiimin kokoonpanoa; jos tiimissä on JavaScriptiä osaavia ihmisiä niin Pythonin käyttö voisi olla turha välikerros visualisoinnissa. Toisaalta taas, jos projekti on pienehkö sekä osaaminen painottuu Data Scientisteihin, niin Pythonilla tehdyt käyttöliittymät voivat olla kustannustehokkaampi ratkaisu.

Haastattelujen mukaan JavaScriptin heikkoutena visualisoinnin kontekstissa on suorituskyky laskutoimituksissa. Sen laskentakirjastot eivät ole yhtä tehokkaita kuin esimerkiksi Pythonin Numpy. Kuitenkin JavaScriptin pitäisi oikein toteutettuna riittää hyvin, etenkin kun sitä käytetään vain visualisointiin ja datan käsittely sekä laskennat suoritetaan backendissä. Datapisteiden määrä tulisi pitää kohtuullisena ja pitäisi huomioida, että ei voida piirtää mitä vain. Esimerkiksi miljoonien aikasarjapisteiden piirtäminen vaatii oikean toteutustavan, jotta data voidaan visualisoida. Suorituskykyasioissa etenkin suurien datamäärien kanssa korostuu siis etenkin applikaation suunnittelun älykkyys. Suuria datamääriä ei pitäisi siirtää selaimen päähän kerralla, vaan esimerkiksi paloittain taustakyselyinä. Suunnitelmallisuus siinä mitä asioita prosessoidaan selaimessa ja mitä backendissä korostuu JavaScript -visualisointien kehityksessä.

4.6 Haastattelujen yhteenveto ja visualisointityökalun valinta diplomityöhön

Tiivistettynä haastatteluista tuli ilmi, että sama asia voidaan toteuttaa lähtökohtaisesti monella eri työkalulla ja sopiva työkalu riippuu hyvin paljon loppukäyttäjän antamista vaatimuksista. Lentotietoaineiston kaltaisen datasetin tehokas, skaalattava sekä interaktiivinen visualisointi vaatii kuitenkin todennäköisesti BI-työkaluja vaativamman visualisointityökalun käyttöä. Siinä missä BI-työkalut olivat helpoin ja eniten valmis pakettiratkaisu, Pythonin Dash antoi jo enemmän vapautta visualisoinnin toteutustavalle ja JavaScriptin Vue puolestaan oli kaikkein eniten toteutuksen vapauksia tarjoava ratkaisu. Toisaalta visualisoinnin toteutustavan vapaus tekee JavaScriptin Vuesta haastavimman työkalun opetella, mikäli visualisoinnin kehittäjällä ei ole aiempaa taustaa. Työkalua valittaessa tulisi myös muistaa tarkistaa mahdolliset lisenssiehdot sekä palvelujen kustannukset. Haastateltuihin työkaluihin liittyen vain BI-työkalut sisältävät maksullisia toiminnallisuuksia. Python sekä JavaScript ovat ilmaisia, mutta myös koodattuihin ratkaisuihin voidaan joutua sisällyttämään maksullisia palveluja esimerkiksi karttapalvelujen muodossa. Lisäksi lisenssiehdot saattavat kieltää joidenkin kirjastojen käytön sotilaskäytössä, mutta tätä ei ole kohdattu nyt läpi käydyissä työkaluissa. Haastattelujen kokemusten sekä kirjallisuudesta tehtyjen havaintojen perusteella eri visualisointityökalujen heikkoudet ja vahvuudet koostettiin taulukkoon 5.

Taulukko 5. Eri visualisointityökalujen vahvuudet ja heikkoudet laajan lentotietoaineiston visualisoinnissa. Lähteenä haastattelut sekä kirjallisuus.

	Vahvuudet	**Heikkoudet**
BI-työkalut	• Erittäin helppo käyttöönottaa, mikäli käytössä on selkeää, strukturoitua dataa • Self-service BI, eli loppukäyttäjä voi lyhyellä opettelulla luoda myös omia visualisointeja • Vahvimmillaan dashboardeissa, joissa tautalla staattinen tietokanta, jota interaktiivisesti selataan sekä suodatetaan • Pitää usein sisällään jakamisen sekä pääsynhallinnan työkalut (Power BI Service/Tableau Server). Niillä pystyy luomaan, jakamaan sekä esittämään raportteja sekä dashboardeja tarvittaessa suoraan pilveen	• Rajoitettu BI-työkalun toiminnallisuuksiin, tietyt visualisointiratkaisut voivat olla mahdottomia toteuttaa • Kaksisuuntainen datan liikehdintä puutteellista. Visualisointinäkymästä ei pysty helposti lähettämään tietoa tietokantaan tai backendissa olevaan analytiikkaratkaisuun • BI-työkalujen mainostetut koodikielien tuet Pythonille ja R:lle eivät mahdollista koodikielien täydellistä käyttöä. Ei suoraa kattavaa tukea tilastollisille tai ennustaville mallinnuksille (Chawla et al 2018) * • Sitoutuminen tarjotun työkalun toimivuuteen tulevaisuudessa • Käyttöliittymä tehdään Excelmäisillä kaavoilla → koodi ei ole yhdessä paikassa selkeästi • Maksullinen
Python Dash	• Pystyy luomaan reaktiivisia web-sovelluksia ilman syvällistä fullstack web-kehitysosaamista • Pystyy tekemään vapaammin visualisointeja kuin BI-työkaluilla • Pelkkä Python kirjoittaminen riittää, ei tarvitse JavaScriptiä tai HTML:ää • Vahvimmillaan pienehköissä projekteissa** • Avoimet rajapinnat helposti käytettävissä** • Ilmainen	• Monisivuiset sovellukset ovat haastavia ** • Tyylittely ja visuaalinen viimeistely voi olla haastavaa ** • Suuret ja monimutkaiset datan visualisoinnit voivat tehdä koodista sotkuista ** • Ei yhtä kätevä kuin BI-työkalut nopeaan plottaukseen, mutta toisaalta ei yhtä monipuolinen kuin JavaScript monimutkaisiin visualisointeihin **
Javascript Vue	• Web-teknologialla luotu sovellus on helppo toimittaa tuotteena ja ei ole riippuvainen käyttöjärjestelmästä** • Frontend-kehittäjiä löytyy työmarkkinoilta helposti** • Toteutus on perinteisellä jaottelulla, jossa selkeä frontend ja backend -> pitää lähdekoodin organisoituneena • Avoimet rajapinnat helposti käytettävissä** • Ilmainen	• Visualisoinnissa tapahtuvissa laskennoissa on useita kertoja hitaampi kuin Python. Mikäli laskentaa ei tehdä backendissa, JS laskentakirjastot eivät ole yhtä tehokkaita kuin esim. Pythonin Numpy ** • Oppimiskäyrä on suurempi kuin BI-työkaluilla tai Dashilla jos ei ole aiempaa JS-taustaa **

* Tieto peräisin vain kirjallisuuslähteestä ** Tieto/kokemus peräisin vain haastatteluista

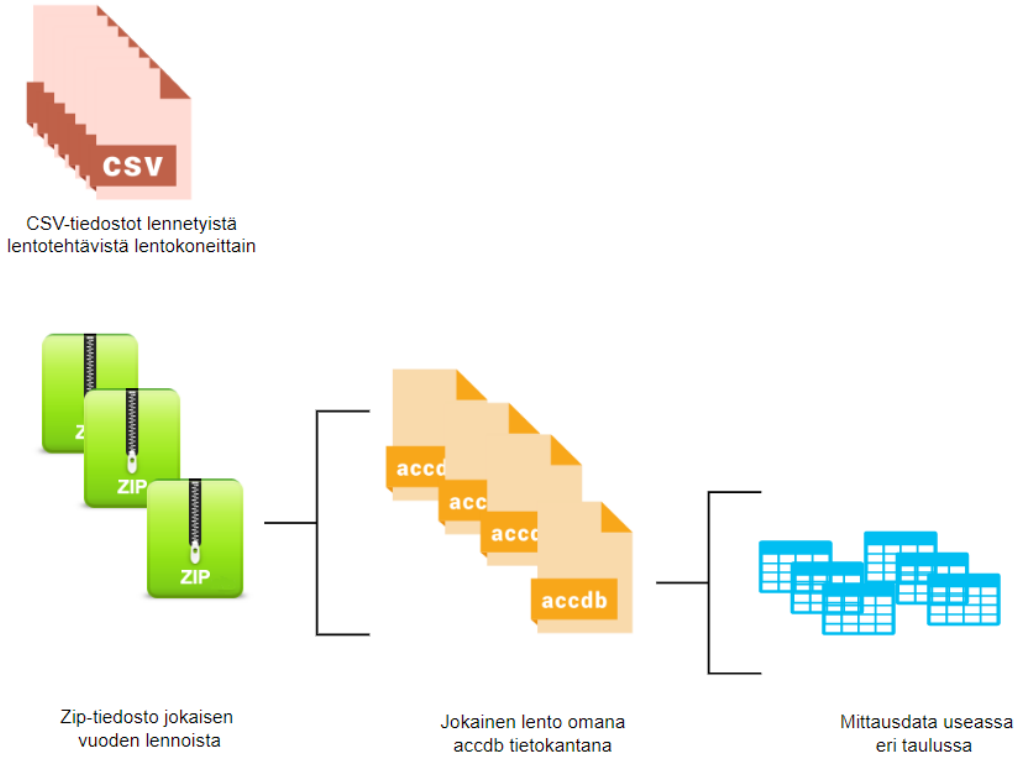
Diplomityön empiiriseen osuuteen, jossa luodaan visualisointi lentotietoaineiston datalle, valittiin **Pythonin Dash**. Valinta perustui siihen, että asiantuntijoilta saadut käyttötapaukset olivat hyvin toteutettavissa Dashilla. Lisäksi aiemman toteutuksen ollessa Dash- sovellus, voitiin diplomityön demo rakentaa sen päälle ja saman toteutuksen yhteyteen. BI-työkaluja ei valittu toteutukseen, koska loppukäyttäjä tarvitsee kustomoituja kuvaajia, joiden parametrit ovat valittavissa. Lisäksi BI-työkalulla olisi ollut hyvin haastavaa vastata toteutuksen vaatimuksiin, jossa visualisoinnin täytyy pystyä lähettämään käyttäjän syötteitä palvelimelle sekä tarjota tukea mahdollisesti toteutettaville koneoppimisratkaisuille. Valintaan vaikutti myös aineistossa olevien parametrien huomattava määrä ja se, että visualisointityökalun kehitysvaiheessa ei voida tietää tarkalleen, mitä parametreja loppukäyttäjä haluaa tarkastella. Esimerkiksi BI-työkaluissa Power BI:llä luotu dashboard käytännössä vaatii kuvaajille ennalta tiedetyt dimensiot, tai vähintäänkin pienen määrän valittavia parametreja dimensioiksi. JavaScriptin Vue puolestaan olisi ollut varmasti vähintään yhtä toimiva toteutustapa, ellei jopa toimivampi, kuin Dash. JavaScript jätettiin kuitenkin valitsematta tämän diplomityön rajoituksista johtuen. Työn toteuttajalla olisi huomattavasti pidempi oppimiskäyrä JavaScript toteutukseen, ja lopputulos olisi kuitenkin työn käyttötapaukset huomioiden käytännössä yhtä pätevä kuin Dashilla toteutettu visualisointi.

5 Visualisointityökalun käytännön toteutus

Tässä luvussa esitetään työn empiirisen osuuden käytännön toteutus. Luku esittelee lentotietoaineiston rakenteen sekä siitä toteutetun datan visualisointityökalun. Visualisointityökalun tavoitteena on vastata haastatteluista kerättyihin käyttötapauksiin.

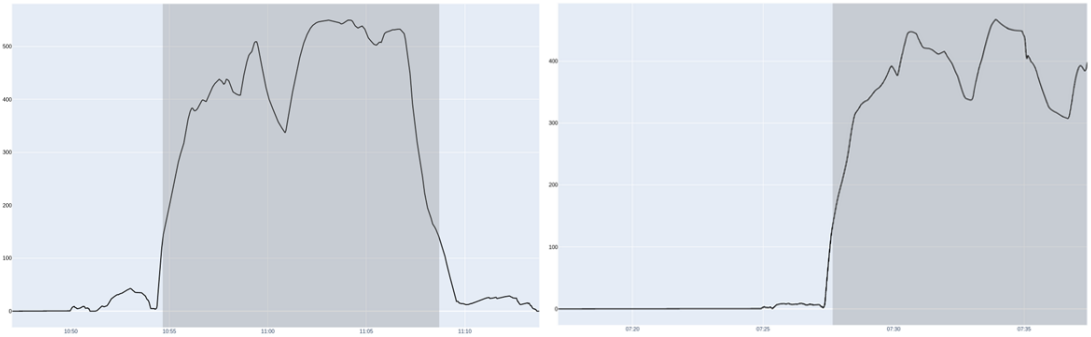
5.1 Datalähteet ja datan eheys

Data on useammalta vuodelta olevaa lentotietoaineistoa, joka on jaettu lennoittain Microsoft Access .accdb tiedostoihin sekä pakattu vuosittaisiksi .zip tiedostoiksi levytilan säästämiseksi. Jokainen yksittäinen lentotiedosto pitää sisällään tietokantataulut yksittäisen lennon eri tallennetuista osa-alueista. Tietokantatauluja on esimerkiksi lennon yhteenvedolle, moottorin tilalle, huoltotallenteille, suorituskyvylle sekä ohjelmistolle. Asejärjestelmiin liittyvät datat on poistettu aineistosta. Yhteenveto- sekä huoltotietoja tallentavia tauluja lukuun ottamatta tietokantataulut ovat aikasarjadataa, joka päivittyy enimmillään 8 kertaa sekunnissa. Tämän lisäksi lennoilta kirjatut asiat, kuten kyseisen lennon lentotehtävä ja lentokenttä ovat erillisissä csv tiedostoissa. Työn aloitushetken lentotietoaineiston rakenne on esitetty kuvassa 12.



Kuva 12. Diplomityössä käsitelty data

Data on pääosin ehyttä sekä toimivaa. Joitakin haasteita datassa kuitenkin oli, kuten puutteellisia sekä parametrirakenteeltaan erilaisia tiedostoja. Tämä vaikeutti esimerkiksi lennoista laskettavien arvojen, kuten laskuhetken lentokorkeuden muutosnopeuden laskemista. Kuvassa 13 on esimerkki rikkinäisestä lentotiedostosta, jossa tiedosto ei pidä sisällään koko lennon dataa.



Kuva 13. Vasemmalla ehjä lentotiedosto ja oikealla lentotiedosto, jossa datan tallennus on loppunut kesken lennon. Parametrina nopeus, tumma korostus on tunnistettu lentoaika.

Aineistolle oli jo aiemmin testattu poikkeavien datapisteiden tunnistusta, mutta tätä ei otettu diplomityön yhteydessä laajemmin käyttöön tai kehitetty lisää. Mikäli datassa olisi ollut merkittävästi virheellistä dataa, olisi sitä voitu siistiä esimerkiksi Augustin et. al (s. 6) esittämällä tavalla käyttäen Gaussin prosessia (Kowalska & Peel 2012). Dataa oli mahdollisesti puhdistettu jossakin aiemmassa vaiheessa prosessia, koska se oli niin ehyttä. Tästä johtuen poikkeavien datapisteiden tunnistaminen jätettiin työn ulkopuolelle.

5.2 Datan yhdistäminen

Eri lentotiedostoissa oleva data päädyttiin yhdistämään yhteen tietokantaan. Tämä mahdollisti helpommin luvussa 4.4 esitetyt käyttötapaukset lentotietojen suodatuksesta lennon aikaisten parametrien arvojen perusteella. Eri lentojen .accdb tiedostot yhdistettiin Pythonilla tehdyllä skriptillä. Data yhdistettiin yhteen SQLite tietokantaan. Aiemmin yksittäisten lentotieostojen aikasarjadatoja oli käsitelty Pythonin dataframeissa, mutta tämä olisi ollut mahdotonta koko aineistolla datan suuresta määrästä johtuen. Pythonin Pandasissa sekä tietokoneen muistissa on yleensä tehokasta pyörittää korkeintaan muutaman gigatavun kokoisia datasettejä kerrallaan. Lentojen yhdistämisessä yhteen tietokantaan täytyi myös lisätä kuvan 12 mukaisesti eri tiedostosta löytyneet lentotehtäville manuaalisesti ylös kirjatut parametrit. Yhdistämisen ohessa pystyttiin laskemaan myös muita lennoista mahdollisesti kiinnostavia tietoja, kuten laskeutumishetken nopeudet kaikilta lennoilta.

Haasteena datan yhdistämisessä esiin nousi se, että kaikki vuosien aikana eri lentokoneyksilöistä tulleet lentotiedostot eivät ole samanlaisia. Esimerkiksi parametri, joka tunnistaa lentokoneen olevan maassa oli eri taulussa osassa lennoista. Myöskään kaikissa lennoissa ei ollut kaikkia tauluja, vaan niissä oli hieman eroavaisuuksia. Kuitenkin tietokantojen yhdistämisen jälkeen lentojen analysointi koko laivueen mittakaavassa on huomattavasti tehokkaampaa, kun lentojen dataa ei tarvitse purkaa ja hakea erikseen. Toisaalta yhdistämisen johdosta lentotietoaineisto vie huomattavasti enemmän levytilaa kuin aiemmin pakattuna olleet yksittäiset lentotiedostot. Datan määrä ei kuitenkaan osoittautunut liian suureksi haasteeksi demon toteutukselle.

5.3 Luodun käyttöliittymän esittäminen

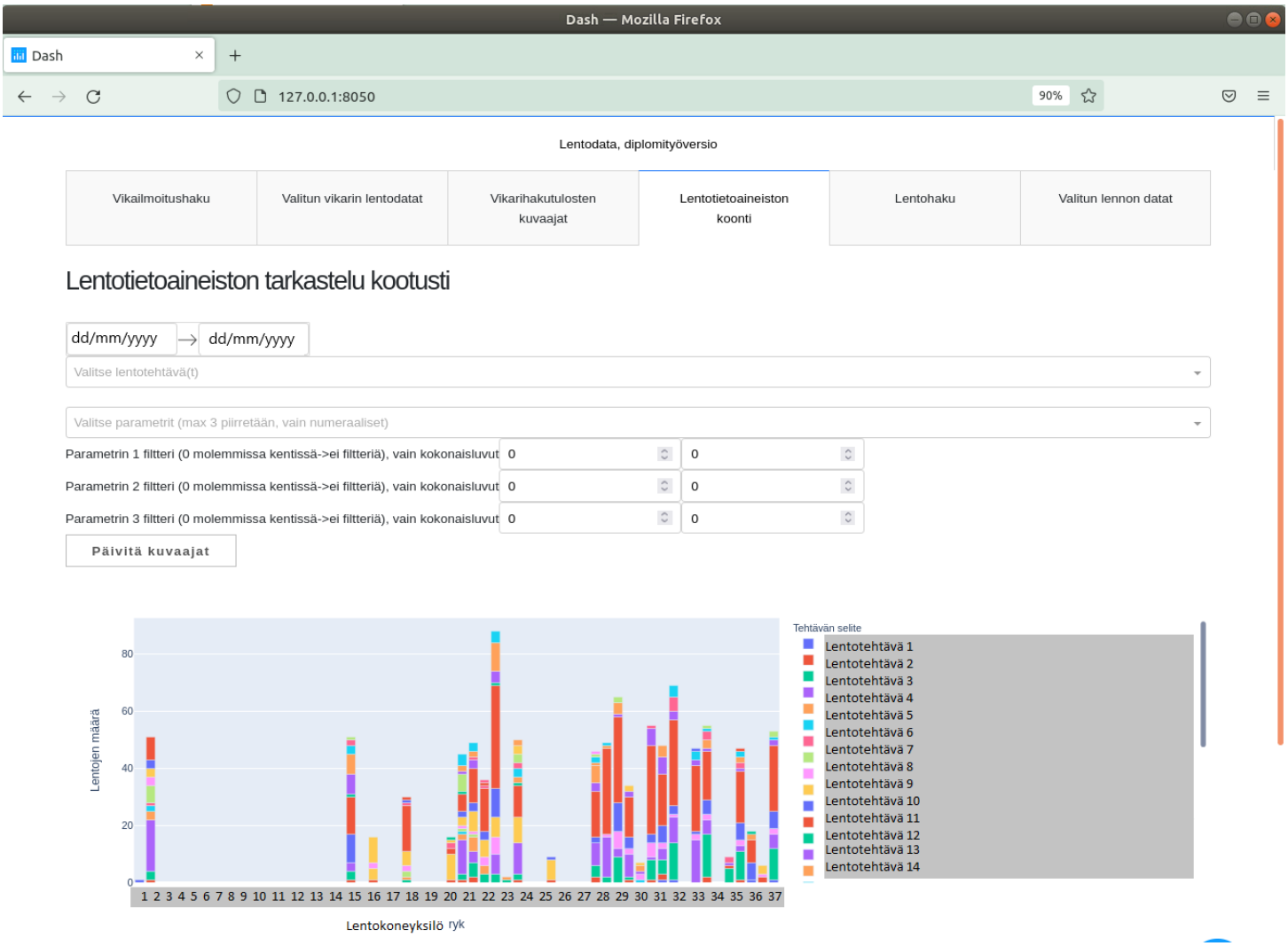
Käyttöliittymään luotiin kaikki luvussa 4.4 valitut käyttötapaukset. Tässä luvussa esitetään kyseisten käyttötapausten toteutukset. Visualisoinneista otetuista kuvakaappauksista on peitetty osa lentotietoaineiston sisältöön liittyvistä tiedoista. Käyttöliittymä on jatkokehitystä diplomityötä aiemmin toteutettuun Dash -sovellukseen. Aiemmassa toteutuksessa ei ollut yhdistettyä tietokantaa eikä siihen tehtävien kyselyiden generointia.

Toiminnaltaan käyttöliittymä toimii siten, että Dash tunnistaa käyttäjän valitsemat syötteet selaimesta ja palauttaa nämä syötteet takaisinkutsuina olevalle Python koodille. Syötteiden perusteella luodaan SQLite -tietokantaan kysely, jolla haetaan tietokannasta valittu data visualisoitavaksi. Kyselyillä haetaan pelkästään oleellinen tieto, kuten suodatusehdot täyttävien lentokoneyksilöiden tunnisteet. Näin visualisoitava datamäärä pysyy pienenä, vaikka taustalla voi olla satoja gigatavuja dataa. Kaikki koodi suoritetaan Docker-konteissa, joten sovellus voidaan ottaa käyttöön helpommin riippumatta missä käyttöjärjestelmässä sitä halutaan suorittaa.

Käyttötapaus 1:

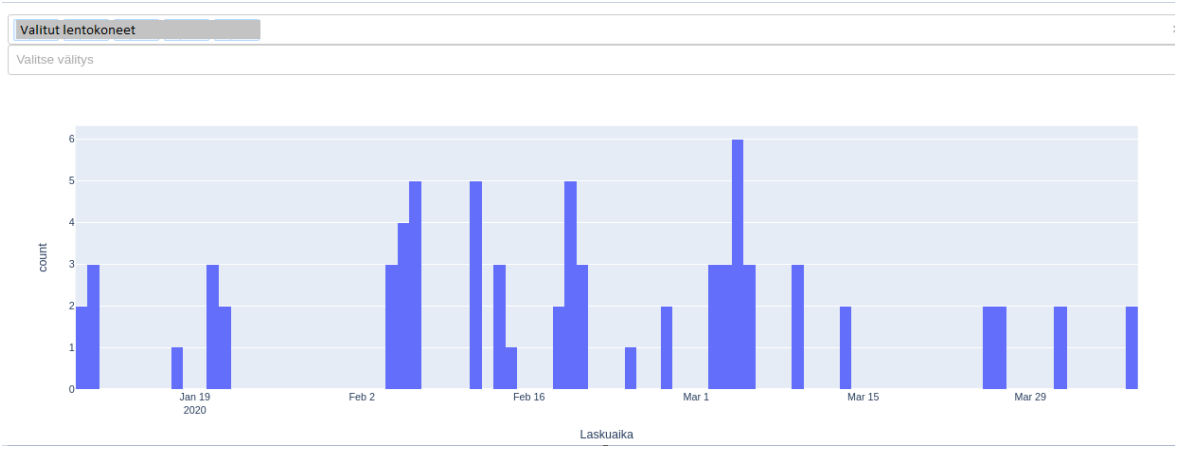
Mahdollisuus skaalata ja siirtyä helposti yhdestä koneesta koko laivueen historiaan, ja mahdollisuus selvittää koko laivueen osalta tapahtumia sekä niiden frekvenssiä.

Kuvassa 14 on esitetty käyttöliittymän etusivu sekä käyttötapaus 1. Käyttäjä voi hakea lentoja koko lentotietoaineistosta ajan, lentotehtävän, lentoyksilön sekä lennolla esiintyneiden parametrien arvojen perusteella. Visualisointiin on mahdollisuus valita samanaikaisesti kolme eri parametria, joiden perusteella valittuja lentoja voidaan suodattaa. Tämä mahdollistaa kyseisten parametrien kannalta kiinnostavien lentojen tehokkaan hakemisen.



Kuva 14. Käyttöliittymän etusivu sekä käyttötapaus 1. Parametrien mukaisten lentojen valinta.

Kuvassa 15 on esitetty toinen käyttötapaukseen 1 liittyvä samalla selainnäkymällä oleva kuvaaja. Siinä pystytään tarkastelemaan aiemmin valittujen lentojen ajankohtia tarkemmin. Tämä mahdollistaa esimerkiksi trendien tunnistamisen datasta, eli onko tarkasteltava tapahtuma kuinka yleinen ja onko se lisääntynyt tai vähentynyt historian aikana. Kuvaajassa palkkien välitystä voi vaihtaa esimerkiksi päiväksi, viikoksi, kuukaudeksi tai vuodeksi riippuen kuinka tarkasti lentoja halutaan tarkastella.

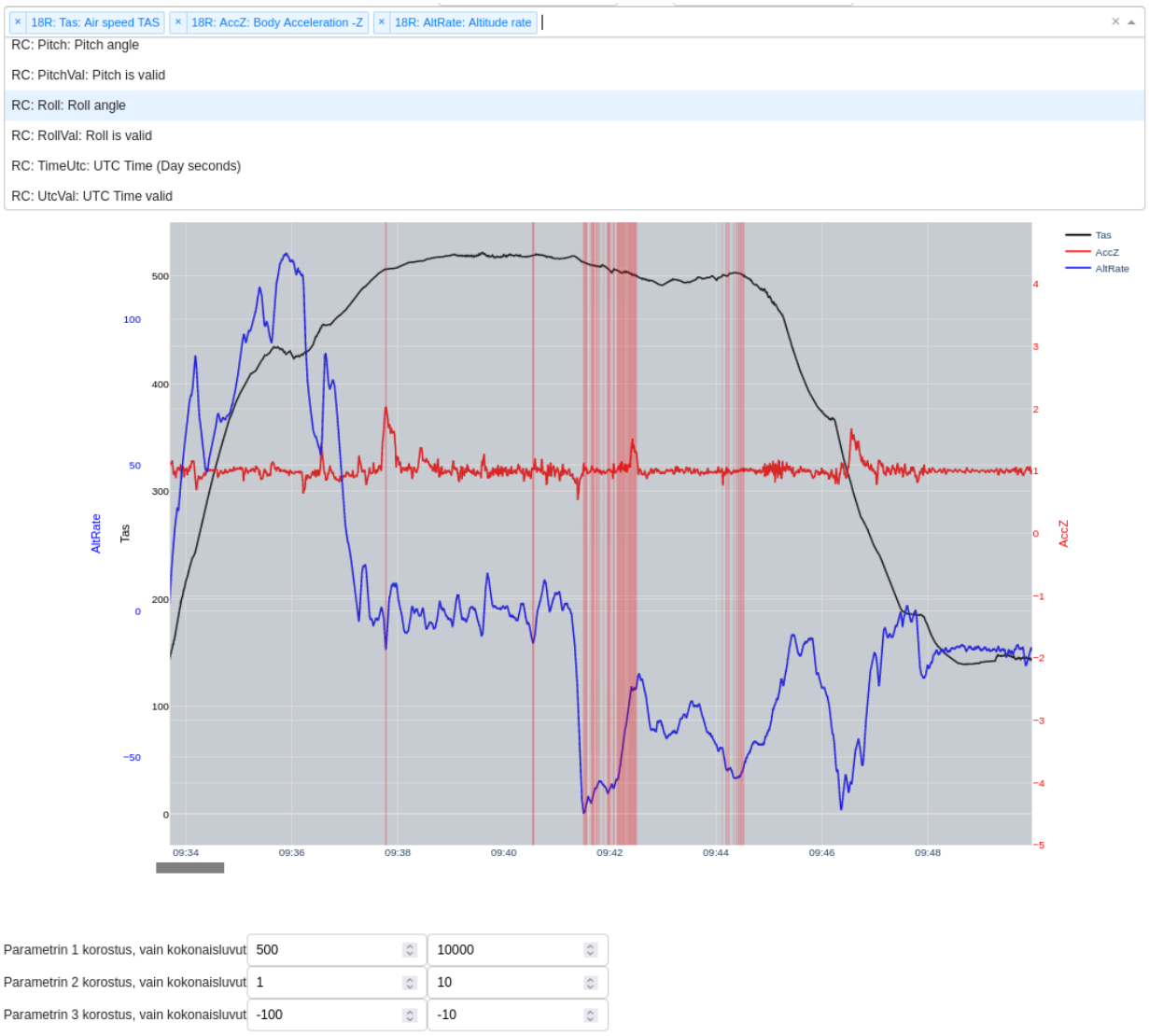


Kuva 15. Käyttötapaus 1. Valittujen parametrien mukaisten lentojen ajankohdat.

Käyttötapaus 2:

Kuvaajaan selkeästi esitettynä datapisteet, joilla valittu parametri ylittää/alittaa annetun haku- tai suodatuskriteerin. Esimerkkinä: lentokorkeus on ollut yli tietyn korkeuden tai kun kone on laskeutumassa tietyllä nopeudella.

Kuvassa 16 on esitetty visualisointi käyttötapauksesta 2. Käyttäjä voi valita lentoparametrien kuvaajalle parametrien valinnat, joiden ajanhetket korostetaan kuvaajassa. Korostetut hetket on visualisoitu punaisella taustalla. Tämä helpottaa tunnistamaan lentotietoaineistosta ne ajanhetket, milloin valitut parametrit ovat olleet syötettyjen arvojen välillä. Käyttökohteina tälle käyttötapaukselle voisi olla esimerkiksi lennon tarkempi tarkastelu koulutustilanteen jälkeen tai vian aiheuttaneen syyn etsintä. Kuvan yläreunassa on myös esitettynä parametrien valinnan valikko avattuna.

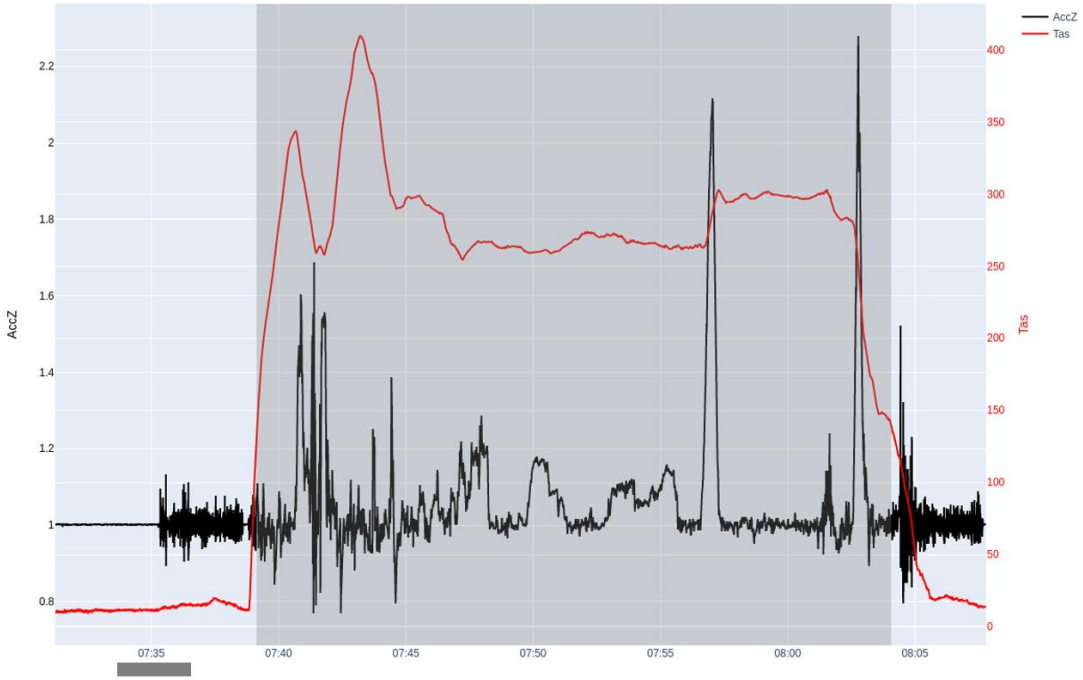


Kuva 16. Käyttötapaus 2. Lentoparametrien valinnat, joiden ajanhetket korostetaan kuvaajassa.

Käyttötapaus 3:

Säätiedon yhdistäminen ja tuominen osaksi lentotiedon analyysia nousujen ja laskujen osalta. Esimerkki: näytetään nousu ja/tai laskuhetken olennaisimmat säätiedot lentokentällä.

Säädatan esitys lisättiin yksittäisen lennon tarkastelun näkymän alapuolelle. Tämä toteutus päätettiin kuitenkin tässä vaiheessa jättää vielä ominaisuuden esittelyn asteelle, eikä näkymää ole yhdistetty tässä versiossa oikeaan tietolähteeseen. Kohdeyrityksellä on kuitenkin jo pääsy kattavaan säädataan, joten sen yhdistäminen olisi myös tarvittaessa toteutettavissa. Visualisointi toteutettiin selkeänä taulukkona, jossa lasku ja nousuhetken oleellisimmat säätiedot ovat omilla riveillään.



Parametrin 1 korostus, vain kokonaisluvut 0 0

Parametrin 2 korostus, vain kokonaisluvut 0 0

Parametrin 3 korostus, vain kokonaisluvut 0 0

Aika	Lentokenttä	Lämpötila (C)	Ilmankosteus (%)	Paine (hPa)	Tuuli (m/s)	Näkyvyys (Km)
2015-01-01 12:03	Halli	5.2	23	1100	5.2	0.8
2015-10-24 18:32	Halli	8.5	40	1050	6.1	2.4

Kuva 17. Käyttötapaus 3. Yksittäisen lennon tarkastelun sivulle on lisätty kyseisen lennon nousu- sekä laskuajankohdan keskeiset säätiedot lentokentällä.

5.4 Visualisointityökalun toimivuuden arvioiminen

Visualisointityökalun toteutuksessa onnistuttiin toteuttamaan käyttötapaukset 1 ja 2 hyvin. Käyttötapaus 3, eli säädatan yhdistäminen, jätettiin käytännön toteutuksessa pois muuten kuin visualisoinnin esimerkin tasolla. Etenkin käyttötapauksen 1 lentotietoaineiston tarkastelu koko laivueen osalta kehitti visualisointityökalua huomattavasti eteenpäin. Aiemmin aineistoa ei pystynyt käsittelemään kokonaisuutena, vaan pelkästään yksittäisinä lentoina. Käyttötapaus 2, jossa lennon aikaisista parametreista voidaan tunnistaa tapahtumia, on myös toimiva. Tarkka visualisointien tarjoama lisähyödyn taso on kuitenkin vain visualisointia käyttävän asiantuntijan määriteltävissä. Työkaluna tämä toteutus on silti hyvin varmasti tehokkaampi kuin manuaalinen Exceleiden käsittely. Työkalun käyttökohteina voi toimia esimerkiksi vianetsintä, huollon optimointi sekä lentokoneohjaajien koulutus. Lisäksi taustalle tehty prosessi, jossa eri lentotietoaineistot yhdistetään yhdeksi tietokannaksi mahdollistaa myös muiden sovellutusten, kuten koneoppivien mallien helpomman käyttöönoton lentotietoaineistolle.

Pythonin Dash oli toteutuksessa toimiva vaihtoehto. Visualisoinnit, joissa valitaan tarkasteltavat parametrit dynaamisesti, olivat helposti toteutettavissa Dashilla. Suorituskyvyltään visualisointityökalu pysyi hyvin käytettävänä. Työn alussa saatavilla ollut Dash-sovellus oli alun perin pyritty tekemään niin, että uutta dataa lisättäessä riittää koodien uudelleensuoritus ja visualisointi toimii uudella datalla. Diplomityössä on noudatettu samaa periaatetta.

Työssä osaltaan haasteena oli vähäinen yhteydenpidon mahdollisuus todellisiin loppukäyttäjiin. Nyt toteutetut visualisoinnit ovat tarkoitettu asiantuntijoiden käytettäväksi, joten toteutuksen aikana ei ollut tarkkaa tietoa mitkä parametreista ovat kaikkein kiinnostavimpia loppukäyttäjien kannalta. Toisaalta nyt visualisointeihin voidaan valita mitkä tahansa parametrit ja työkalu tukee siten erilaisia loppukäyttäjiä, joten tämä ei osoittautunut puutteeksi. Diplomityön toteutuksen loppupuolella visualisointityökalua esiteltiin potentiaalisille loppukäyttäjille, joilta tuli positiivista palautetta.

5.5 Kehitystyön toimivuuden arvioiminen

Anacondan (2020) tutkimuksen mukaan datan kanssa työskentelevät käyttävät keskimäärin lähes puolet työajastaan datan valmisteluun, eli datan keräämiseen sekä siivoamiseen. Datan visualisointiin käytettiin puolestaan aikaa vain reilu viidesosa työajasta. Tämä diplomityö voi osaltaan tukea kyseisen tutkimuksen lukuja, koska datan valmistelu vei merkittävän osan diplomityölle varatusta työajasta. Datan suuri määrä, sen tuntemattomuus sekä myös osittain sen heterogeenisyys vei aikaa pois datan visualisoinnin toteuttamiselta. Toisaalta diplomityön demossa tarkoitus olikin toteuttaa datan visualisointityökalu, eikä pelkästään visualisoida jo valmista datasettiä. Osaltaan nyt toteutettu demo siis kuvastaa paremmin reaalimaailman toteutusta, jossa hyödyllinen datan visualisointi on paljon enemmän kuin pelkästään kuvaajien valitsemista.

Yksi demoa vastaava toteutus kirjallisuudesta on Augustin et al. (2021) toteutus helikoptereiden datalle. Siinä datan yhdistämisen sekä visualisoinnin työkalua oli kuitenkin toteuttamassa yhteistyössä kaksi yritystä sekä kahdesta eri yliopistosta olevat tutkimusryhmät. Riippuen datalähteiden laadusta, voi lentotietoaineiston tehokas visualisointi siis olla huomattavasti yhtä diplomityötä laajempi kokonaisuus. Tässä työssä toki etuna oli se, että Dash-visualisointeja oli jo luotu yksittäisten lentojen osalta työn alkuvaiheessa.

6 Johtopäätökset ja yhteenveto

Tässä luvussa esitetään tutkimuksen tulokset vastaamalla tutkimuskysymyksiin sekä arvioidaan mitä saavutetut tulokset merkitsevät laajemmassa perspektiivissä. Diplomityön tavoitteena oli selvittää eri datan visualisointiteknologioiden toimivuutta lentotietoaineiston analysoinnin näkökulmasta.

Millaisilla teknologioilla voidaan toteuttaa käyttäjänäkymiä data-analytiikkajärjestelmiin?

Datan visualisointiin on olemassa erittäin kattava työkalupakki ja monilla eri työkaluilla voidaan luoda toisiaan vastaavia toteutuksia. Tässä diplomityössä tarkempaan tarkasteluun otettiin BI-työkalut, Pythonin Dash sekä JavaScriptin Vue. Tutkimuksen perusteella yhdessäkään tarkastellussa teknologiassa ei ole mitään merkittävää estettä toimia ainakaan lentotietoaineiston visualisoinnin työkaluna. BI-työkaluja on hyvin paljon markkinoilla ja niissä yhä selkeämmiksi markkinajohtajiksi ovat nousseet Microsoftin Power BI sekä Salesforcen Tableau. BI-työkalut ovat erittäin monikäyttöisiä ja keskeisimmät erot eri BI-työkaluissa ovat niiden hinnoittelumalleissa. Kuitenkin yrityksellä voi olla vaatimuksia esimerkiksi datan säilytykseen tai raporttipalvelimen käyttöjärjestelmiin liittyen, jotka tulee ottaa huomioon työkalua valittaessa. Pythonin avoimen lähdekoodin kirjasto Dash pyrkii tuomaan datan visualisointiteknologiat helpommin data-analyytikon työkalupakkiin. Se mahdollistaa BI-työkaluja kehittyneemmät kustomointimahdollisuudet visualisoinneissa, jotka toteutetaan verkkoselaimessa suoritettavan applikaationa. Dash voi olla kuitenkin haastava työkalu, mikäli toteutettava käyttäjänäkymä on monisivuinen sekä monimutkainen. JavaScriptin Vue puolestaan edustaa perinteisemää web-kehitystä, jossa datan visualisoinneista luodaan verkkosovellus käyttäen yleisesti käytössä olevia web-teknologioita. JavaScript-kehyksen valinta Vuen ja muiden vaihtoehtojen välillä perustuu usein eniten kehittäjätiimin omiin mieltymyksiin.

Tässä työssä data-analytiikkajärjestelmiä tarkasteltiin tarkemmin sotilasilmailun näkökulmasta. Viime vuosien kirjallisuuslähteissä lentotietoaineistoa on visualisoitu sekä analysoitu pääasiassa kustomoiduilla ohjelmistopohjaisilla ratkaisuilla. Esimerkiksi JavaScriptillä tehtyjä käyttäjänäkymiä oli toteutettu lentotietoaineistoille aiemmin.

Mitkä ovat eri datan visualisointiteknologioiden vahvuudet ja heikkoudet?

Eri datan visualisointiteknologioiden vahvuuksia sekä heikkouksia tarkasteltiin tässä diplomityössä valittujen teknologioiden osalta. Tietoa haettiin kirjallisuudesta sekä haastattelemalla yrityksen ohjelmistokehittäjiä. Yleisimmät BI-työkalut, Power BI sekä Tableau ovat toimintaperiaatteeltaan hyvin samankaltaisia. Suurimpia etuja muihin teknologioihin verrattuna ovat BI-työkalujen mahdollistama self-service-BI, eli loppukäyttäjä pystyy itse luomaan sekä muokkaamaan käyttäjänäkymiä hyvin intuitiivisesti. BI-työkalut eivät kuitenkaan aina sovellu erityistä kustomointia vaativien visualisointinäkymien toteuttamiseen, vaan käyttäjä on enemmän sidottu palveluntarjoajan valmiiksi luomiin raameihin. BI-työkalut ovat myös yleensä maksullisia. Pythonin Dash puolestaan antaa kehittäjälle enemmän mahdollisuuksia käyttöliittymän kustomointiin, mutta voi olla raskas kehittää, mikäli käyttöliittymä on laaja. JavaScriptin Vue pitää sisällään laajimmat visualisointien kustomointimahdollisuudet, mutta sitä ei ole yhtä kevyttä kehittää ilman aiempaa kokemusta. Lähtökohtaisesti työkalun valintaan vaikuttaa eniten se, kuinka kustomoiduille visualisoinneille on tarvetta, tarvitseeko loppukäyttäjän itse päästä luomaan visualisointeja, millainen on kehittäjätiimin oma tausta sekä mihin ympäristöön data-analytiikkaratkaisu tullaan asentamaan. BI-työkaluilla tehdyt visualisoinnit voivat olla usein nopeampia toteuttaa. Ne kuitenkin sitovat toteutetun käyttöliittymän omaan ympäristöönsä koodikielillä toteutettuja ratkaisuja tiukemmin.

Millainen datan visualisoinnin käytännön toteutus soveltuu parhaiten kohdeyritykselle tutkimuksen kohteena olevan lentotietoaineiston visualisointiin?

Kohdeyrityksen työntekijöille tehtyjen haastattelujen perusteella koottiin kolme toivottua käyttötapausta datan visualisoinnille. Nämä käyttötapaukset olivat lentojen visualisointi ja suodatus lennon aikana toteutuneiden parametriarvojen perusteella, tehokkaampi lennon aikaisten parametriarvojen visualisointi sekä säädatan yhdistäminen lentotietoaineistoihin. Näistä käyttötapauksista toteutettiin kaksi ensimmäistä toimivaksi ja kolmas esimerkkivisualisoinnin asteelle. Toteutetussa demossa taustalle luotiin koottu SQL-tietokanta, joka mahdollisti lentojen visualisoinnin sekä hakemisen nyt myös kootusti yksittäisen lentokoneen tai koko laivueen tasolla. Tämä voi tehostaa vianselvitystä, koska työkalulla pystytään vastaamaan kysymyksiin ”kuinka usein tätä on tapahtunut aiemmin?” tai ”Onko tapahtuma lisääntynyt historian aikana?”. Vianselvitys on oleellinen osa lentokoneiden elinkaaren hallintaa. Toteutukseen valittiin Pythonin Dash. Valintaan vaikutti vaadittujen käyttötapausten laajuus, diplomityön aikarajoitteet, tekijän tausta sekä aiempi olemassa oleva Dash-toteutus, jota pystyttiin jatkokehittämään. Dash -toteutuksessa pystyttiin BI-työkaluja kätevämmin hallita laajaa massaa visualisoitavia parametreja. Nyt visualisoitu data oli erittäin moniparametrista, eikä käyttöliittymän kehityksessä voinut olla varmuutta, mitkä parametreista ovat oleellisimpia. Käyttöliittymä toteutettiinkin asiantuntijasovelluksena, joka olettaa käyttäjän tietävän parametrien sisällön. Dash-sovellusta kehitettiin mikropalveluarkkitehtuurin mukaisesti omassa Docker-kontissaan. Tämä mahdollistaa sen, että esimerkiksi erillisiä koneoppivia malleja pystytään tarvittaessa lisäämään helposti käyttöliittymän taustalle kehittämällä niitä omissa konteissaan.

6.1 Tulosten käytännöllinen ja tieteellinen merkitys

Työn teoriaosuus kokoaa data-analytiikasta saatavilla olevaa kirjallisuutta yhteen. Näkökulmaksi on otettu viime vuosien julkaisut, joissa on käsitelty lentotietoaineiston analytiikkaa sekä esittämistä. Diplomityön teoriaosuuden kaltaista kirjallisuuskatsausta viime vuosien lentotietoaineiston analytiikkaan ei ollut tehty aiemmin. Teoriaosuuden perusteella voidaan sanoa, että ilmailualan data-analytiikassa on usein samoja haasteita kuin lähes kaikilla muillakin toimialoilla; dataa on paljon, se on hajautuneena eri järjestelmiin, se ei ole täysin puhdasta eikä sitä hyödynnetä tarpeeksi.

Gartnerin analytiikan maturiteettimallin näkökulmasta diplomityössä toteutettu demo kehitti kohdeyrityksen lentotietoaineiston analytiikkaa ensimmäiseltä ”Mitä tapahtui?” -tasolta kohti toisen tason ”Miksi tapahtui?” diagnosoivaa analytiikkaa. Visualisoimalla koko laivueen data kerralla ja mahdollistamalla lentojen analysointi kootusti, voidaan helpommin löytää syitä havaittujen ilmiöiden taustalla sekä löytämään datasta oleellinen tieto. Työssä tehty demo myös kehitti yksittäisen lennon parametrien visualisointia lentojen tarkempaa analysointia varten.

6.2 Työn luotettavuuden arviointi

Puusan et al. (2020) mukaan laadullisen tutkimuksen luotettavuuden arviointiin kuuluu aineiston reliabiliteetin ja validiteetin tarkastelu. Reliaabeliusarviossa kysytään, onko tutkittavaa ilmiötä tutkittu siten, että mittaustilanne, mittaaja tai satunnaiset tekijät eivät vaikuta tutkimustuloksiin. Reliabiliteettia tarkastellessa on tarkasteltava käsittelyn sekä analyysin lisäksi myös koko tutkimusprosessia. Validiuden arvioinnissa puolestaan kysytään, että tutkitaanko oleellista ilmiötä ja onko tutkimuksessa tunnistetut kausaliteetit todenmukaisia. Lisäksi validiuden arvioinnissa kysytään, kuinka hyvin tutkimustulokset voidaan yleistää laajemmin erilaisiin tilanteisiin, henkilöihin ja organisaatioihin.

Tämän tutkimuksen reliabiliteetin näkökulmasta datan visualisointiteknologioiden haastattelujen osalta aineistojen otoskoko oli suhteellisen pieni. Lisäksi haastatteluissa korostui erityisen paljon yksittäisten haastateltavien subjektiiviset kokemukset. Toisaalta tämä oli myös tarkoituksenmukaista, koska tutkimuksessa haluttiin selvittää juuri kohdeyrityksen omia kokemuksia kirjallisuudesta löytyvän materiaalin tueksi. Lisäksi empiirisen osuuden demon käyttötapauksia varten ei pystytty tutkimuksen rajoituksista johtuen haastattelemaan kaikkia potentiaalisia loppukäyttäjiä. Optimaalisessa tilanteessa loppukäyttäjien kanssa olisi oltu jatkuvassa vuorovaikutuksessa koko demon kehitystyön ajan. Toisaalta tutkimuksessa tehty demo voidaan nähdä riittäväksi tämän tutkimuksen kontekstissa, koska nyt tehtyjä visualisointeja pystytään jatkokehittämään tarvittaessa. Tämän työn lisäksi koko data-analytiikan kirjallisuudessa sekä ohjelmistoista kirjoitetuissa vertailuissa on haasteena se, että ala on erittäin nopeasti kehittyvä. Lähteet pitävät usein

sisällään vanhentunutta tietoa ja todennäköisesti myös tämä työ vanhenee tämän vuosikymmenen aikana ainakin osittain.

Validiuden näkökulmasta tutkimustulokset voidaan lähtökohtaisesti yleistää vastaavanlaisiin datasetteihin. Kuitenkin tutkimuksen aikarajoitteista johtuen joitain oleellisiakin datan visualisointiteknologiota on voinut jäädä tutkimuksessa käsittelemättä. Tämä tulee ottaa huomioon yleistettäessä tutkimustuloksia, etenkin jos halutaan vertailla nyt käsiteltyjä teknologioita muihin markkinoilla oleviin ratkaisuihin.

6.3 Jatkotutkimus

Selkeä jatkotutkimusaihe tälle diplomityölle olisi kehittää lentotietoaineiston data-analytiikkaa kohti ennustavaa analytiikkaa. Lentämiseen liittyy lähtökohtaisesti hyvin suuret rahalliset kustannukset ja etenkin sotilasilmailussa tavoitteena on optimoida lentokoneiden käyttöä aina niiden elinkaaren loppuun asti. Huollon optimointi voisi olla merkittäviäkin kustannussäästöjä tarjoava jatkotutkimusaihe, joka voisi nostaa myös koneiden käytettävyysprosenttia korkeammaksi. Toisaalta kuten kirjallisuuskatsauksessa huomattiin, on huollon analytiikkaa ollut jo nyt hyvin keskeinen tutkimuskohde. Lentoprofiilien sekä huoltotietojen yhdistäminen voisi myös tarjota lisämahdollisuuksia vika-analytiikkaan ja diagnostiikkaan.

Lisäksi etenkin miehittämättömien ilma-alusten tuottaman data voi pitää sisällään uusia tutkimusaiheita. Kaikkiaan data-analytiikka tulee olemaan yhä enemmän automatisoitu prosessi tulevaisuudessa. Tutkituille data-analytiikkaratkaisuille, jotka toimivat erittäin suurilla datamassoilla on varmasti tulevaisuudessa kysyntää.

Lähteet

Ali, S., Gupta, N., Nayak, G., Lenka, R. 2016. Big data visualization: Tools and challenges. 2016 2nd International Conference on Contemporary Computing and Informatics (IC3I). s. 656-660.

Anaconda. 2020. The State of Data Science 2020 Moving from hype toward maturity [WWW-dokumentti]. [Viitattu 3.6.2022] Saatavissa: https://www.anaconda.com/state-of-data-science-2020?utm_medium=press&utm_source=anaconda&utm_campaign=sods-2020&utm_content=report

Anderson, T. 2020 Microservices guru warns devs that trendy architecture shouldn't be the default for every app, but 'a last resort'. [WWW-dokumentti]. [Viitattu 3.4.2022] Saatavissa: https://www.theregister.com/2020/03/04/microservices_last_resort/

Armbrust, M., Das, T., Paranjpye, S., Xin, R., Zhu, S., Ghodsi, A., Yavuz, B., Murthy, M., Torres, J., Sun, L., Boncz, P.A., Mokhtar, M., Hovell, H.V., Ionescu, A., Luszczak, A., Switakowski, M., Ueshin, T., Li, X., Szafranski, M., Senster, P., & Zaharia, M. 2020. Delta lake: high-performance ACID table storage over cloud object stores. Proceedings of the VLDB Endowment. Vol 13, s. 3411–3424.

Augustin, M., Dunaway, D., Le, D., Nixon, S. 2021. Boiling down aviation data: Development of the aviation data distillery. 77th Annual Vertical Flight Society Forum and Technology Display, FORUM 2021: The Future of Vertical Flight.

Badiru, A. 2020. Data Analytics: Handbook of Formulas and Techniques. CRC Press.

272 s.

Barchart. 2019. Comparing HTML5 Canvas vs. SVG for Charting. [WWW-dokumentti]. [Viitattu 14.4.2022] Saatavissa: https://www.barchart.com/solutions/company/blog/3852318/comparing-html5-canvas-vs-svg-for-charting

Barros, T. B. Serrano, V. Infante, V. Franco, P. Antunes. 2020. Prediction of the fatigue lifetime of the Portuguese Air Force Epsilon TB-30 aircraft. Engineering Failure Analysis. Volume 116.

Bayoumi, A. & Matthews, R. 2020. Condition-based maintenance to predictive maintenance: A use case on selected USARMY military aircraft. International Journal of COMADEM. Vol. 23, (2), s. 3-8.

Bucchiarone A., Dragoni N., Dustdar s., Lago P., Mazzara M. 2020. Microservices: Science and Engineering. Cham, Springer. 376 s.

Chawla, G., Bamal, S., Khatana, R. 2018. Big Data Analytics for Data Visualization: Review of Techniques. International Journal of Computer Applications. Vol. 182, (21), s. 37–40.

Chung, S., Ma, H., Hansen, M., Choi, T. 2020. Data science and analytics in aviation. Transportation Research Part E: Logistics and Transportation Review. Volume 134.

Columbus, L. 2019. IDC Top 10 Predictions For Worldwide IT, 2019. [WWW-dokumentti]. [Viitattu 3.2.2022] Saatavissa: https://www.forbes.com/sites/louiscolumbus/2018/11/04/idc-top-10-predictions-for-worldwide-it-2019/?sh=3ab2d54b7b96

DASC. 2022. Digital Acionics Systems Concerence. [WWW-dokumentti]. [Viitattu 13.4.2022] Saatavissa: https://2022.dasconline.org/

Efthymiou, M., McCarthy, K., Markou, C., O'Connell, J. 2022. An Exploratory Research on Blockchain in Aviation: The Case of Maintenance, Repair and Overhaul (MRO) Organizations. Sustainability. Vol 14. s. 2643.

Forrester. 2021. The Forrester Wave™: Augmented BI Platforms, Q3 2021 [WWW-dokumentti]. [Viitattu 13.4.2022] Saatavissa: https://reprints2.forrester.com/#/assets/2/108/RES176073/report

Gartner. 2014. Gartner Says Advanced Analytics Is a Top Business Priority. [WWW-dokumentti] [Viitattu 9.7.2022] Saatavissa: https://www.gartner.com/en/newsroom/press-releases/2014-10-21-gartner-says-advanced-analytics-is-a-top-business-priority

Gartner. 2022. Magic Quadrant for Analytics and Business Intelligence Platforms. [WWW-dokumentti]. [Viitattu 5.2.2022]. Saatavissa: https://info.microsoft.com/ww-landing-2022-gartner-mq-report-on-bi-and-analytics-platforms.html?lcid=en-us

Greif, S. 2022. The 2021 State of JS survey. [WWW-dokumentti]. [Viitattu 3.4.2022] Saatavissa: https://2021.stateofjs.com/en-US/about

Hao, M., Yong, X., Xi, X., Zhang T., Zhang, Y. 2020. Method for Optimising Mission-Specific Inventory of Aviation Materials. IEEE International Conference on Information Technology, Big Data and Artificial Intelligence (ICIBA). s. 506-511.

Holst, A. 2021. Total data volume worldwide 2010-2025 [WWW-dokumentti] [Viitattu 3.2.2022] Saatavissa: https://www.statista.com/statistics/871513/worldwide-data-created/

Hwang, J. 2020. Plotly Dash vs Streamlit — Which is the best library for building data dashboard web apps? [WWW-dokumentti]. [Viitattu 3.2.2022] Saatavissa: https://towardsdatascience.com/plotly-dash-vs-streamlit-which-is-the-best-library-for-building-data-dashboard-web-apps-97d7c98b938c

Indrasiri, K. & Siriwardena, P. 2018. Microservices for the Enterprise Designing, Developing, and Deploying. Berkeley, CA. Apress. 441 s.

Johnson, J., Shiff, L. 2021. What Is Microservice Architecture? Microservices Explained. BCM DevOps Blog. [WWW-dokumentti]. [Viitattu 13.7.2022] Saatavissa: https://www.bmc.com/blogs/microservices-architecture/

Kasanen, E., Lukka, K., Siitonen, A. 1991. Konstruktiivinen tutkimusote liiketaloustieteessä. Liiketaloudellinen aikakauskirja. Vol. 40, nro 3, s. 301–329.

Kowalska, K. & Peel, L. 2012. Maritime anomaly detection using Gaussian Process active learning. 15th International Conference on Information Fusion. s. 1164-1171.

Larrucea, X., Santamaria, I., Colomo-Palacios, R. & Ebert C. 2018. Microservices. IEEE software. Vol. 35, (3), s. 96–100.

Llave, M. 2018. Data lakes in business intelligence: Reporting from the trenches. Procedia Computer Science. Vol. 138, s. 516–524.

Lukka, K. 2001. Konstruktiivinen tutkimusote. Menetelmäartikkeli. [WWW-dokumentti]. [Viitattu 2.5.2022] Saatavissa: https://metodix.wordpress.com/2014/05/19/lukkakonstruktiivinen-tutkimusote/

Matlab. 2022. Products overview. [WWW-dokumentti]. [Viitattu 15.4.2022]. Saatavissa: https://se.mathworks.com/products/matlab.html

McKinney, W. 2017. Python for Data Analysis: Data wrangling with Pandas. NumPy and IPython. O'Reilly Media, Incorporated. 2. painos. 523 s.

MDN Web Docs. 2022. Introduction to client-side frameworks. [WWW-dokumentti]. [Viitattu 5.3.2022]. Saatavissa: https://developer.mozilla.org/en-US/docs/Learn/Tools_and_testing/Client-side_JavaScript_frameworks/Introduction

Microsoft. 2022a. Power BI pricing [WWW-dokumentti]. [Viitattu 10.4.2022]. Saatavissa: https://powerbi.microsoft.com/en-us/pricing/

Microsoft 2022b. Hardware and software requirements for installing Power BI Report Server. [WWW-dokumentti]. [Viitattu 5.4.2022]. Saatavissa: https://docs.microsoft.com/bs-latn-ba/power-bi/report-server/system-requirements

Microsoft. 2022c. Describe bronze, silver, and gold architecture. [WWW-dokumentti]. [Viitattu 5.4.2022]. Saatavissa: https://docs.microsoft.com/en-us/learn/modules/describe-azure-databricks-delta-lake-architecture/2-describe-bronze-silver-gold-architecture

Mikkonen T., Taivalsaari, A. 2008. Web Applications – Spaghetti Code for the 21st Century. Sixth International Conference on 74 Software Engineering Research, Management and Applications (SERA '08). IEEE Computer Society. s. 319–328.

Nakazawa, R., Ueda, T., Enoki, M., Horii, H. 2018.Visualization Tool for Designing Microservices with the Monolith-First Approach. IEEE Working Conference on Software Visualization (VISSOFT), s. 32-42.

Oliveira, A. & Bernardino, J. 2020. Evaluating Self-Service BI and Analytics Tools for SMEs. Conference: 17th International Conference on e-Business. s. 89-97.

Paluszek, M. & Thomas, S. 2021. MATLAB Recipes: A Problem-Solution Approach. Berkeley, CA. Apress. 2. painos. 415 s.

Panimalar, A., Shree, V. Katthrine, V. 2017. The 17 V's Of Big Data. International Research Journal of Engineering and Technology (IRJET). Vol. 4 (09). s. 329-335.

Parts, Ü., Töytäri, P., Forsman, K., Horn, S., Keskinen, J., Kohtamäki, M., Martinsuo, M., Kärri, T., Rajala, R., Hakanen, T., Laine, T., Talaoui, Y., Rabetino, R., Heimonen, J., Huikkola, T., Kunttu, I., Boldosova, V., Jähi, M., Mikkola, M., ... Laitinen, J. 2017. S4Fleet – Service Solutions for Fleet Management: FINAL REPORT 6/2017. DIMECC Publication series. Vol. 19. 168 s.

Plotly. 2017. Introducing Dash. [WWW-dokumentti] [Viitattu 11.2.2022] Saatavissa: https://medium.com/plotly/introducing-dash-5ecf7191b503

Polyzotis, N. & Zaharia, M. 2021. What can Data-Centric AI Learn from Data and ML Engineering? 35th Conference on Neural Information Processing Systems (NeurIPS 2021), Sydney, Australia.

Ponce F., Márquez, G., Astudillo, H. 2019. Migrating from monolithic architecture to microservices: A Rapid Review. 38th International Conference of the Chilean Computer Science Society (SCCC), s. 1-7.

Prabhu, C.S.R., Chivukula, S., Mogadala, A., Ghosh R., Livingston, J. 2019. Big Data Analytics: Systems, Algorithms, Applications. 1. painos. Springer Singapore. 412 s.

Preeth, E., Mulerickal, F. J. P., Paul, B., Sastri, Y. 2015. Evaluation of docker containers based on hardware utilization. In 2015 international conference on control communication & computing India (ICCC). IEEE. s. 697–700.

Puolustusvoimat. 2022 Lentokalusto. [WWW-dokumentti] [Viitattu 3.6.2022] Saatavissa: https://puolustusvoimat.fi/kalusto#/category/view/id/38348309

Puusa, A., Juuti, P. & Aaltio, I. 2020. Laadullisen tutkimuksen näkökulmat ja menetelmät. Helsinki. Gaudeamus. 380 s.

Richter, K. & Walther, J. 2016. Supply chain integration challenges in commercial aerospace: A comprehensive perspective on the aviation value chain. 297 s.

Runkler, T. 2016. Data Analytics Models and Algorithms for Intelligent Data Analysis. 2. painos. Wiesbaden, Springer. 137 s.

Solita. 2021. Tableau has removed minimum purchase requirement from their license policies. [WWW-dokumentti] [Viitattu 11.4.2022] Saatavissa: https://data.solita.fi/tableau-has-removed-minimum-purchase-requirement-from-their-license-policies/

Sommerville, I. 2016. Software Engineering. 10. painos. Boston. Pearson. 816 s.

Staegemann, D., Volk, M., Shakir A., Lautenschläger, E., Klaus, T. 2021. Examining the Interplay Between Big Data and Microservices – A Bibliometric Review. Complex Systems Informatics and Modeling Quarterly (CSIMQ). Article 157, (27). s. 87–118.

Strobierski, T. 2021. What's the difference between data analytics & data science? [WWW-dokumentti] [Viitattu 14.2.2022] Saatavissa: https://online.hbs.edu/blog/post/data-analytics-vs-data-science

Sulava. 2019. Power BI - kaikki mitä sinun tulee tietää aloittaaksesi. [WWW-dokumentti].

[viitattu 9.3.2022] Saatavissa: https://www.sulava.com/power-bi-kaikki-mita-sinun-tuleetietaa-aloittaaksesi/

Sun, A., Guo, D., Wang. R. 2021. A Data-based Expert System for Aero-Engine Gas Path Fault Diagnosis. 33rd Chinese Control and Decision Conference (CCDC). s. 2917-2922.

Tableau. 2022. Tableau Server for Linux. [WWW-dokumentti]. [viitattu 9.3.2022] Saatavissa: https://www.tableau.com/products/linux

TypeScript. 2022. The Basics. [WWW-dokumentti] [Viitattu 14.2.2022] Saatavissa: https://www.typescriptlang.org/docs/handbook/2/basic-types.html

Viitanen, T., & Siljander, A. 2021. A Review of Aeronautical Fatigue Investigations in Finland April 2019 – April 2021. VTT Technical Research Centre of Finland. ICAF National Review – Finland. 73 s.

Vo.T.H, P., Czygan, M. Raman, K., Kumar, A. 2017. Python: Data Analytics and Visualization. Packt Publishing. 847 s.

Wang, Y., Perry, M., Whitlock, D., Sutherland, W. 2022. Detecting anomalies in time series data from a manufacturing system using recurrent neural networks. Journal of Manufacturing Systems. Vol. 62. s. 823-834.

Widjaja, J. 2020. How analytics maturity models are stunting data science teams. [WWW-dokumentti] [Viitattu 14.3.2022] Saatavissa: https://towardsdatascience.com/how-analytics-maturity-models-are-stunting-data-science-teams-962e3c62d749

Zhang, J. & Zhang, P. 2016. Time series analysis methods and applications for flight data. Springer. 240 s.

Liitteet

Liite 1. Haastattelurunko

- Haastateltavan rooli yrityksessä?
- Mikä on oma kokemus [Visualisointityökalusta]?
- Missä [Visualisointityökalu] on hyvä?
- Missä [Visualisointityökalu] on huono?
- Kuinka [Visualisointityökalu] soveltuu Patrian data-analytiikkajärjestelmään?
    - Suorituskyvyn riittävyys suurella datalla?
    - Tuki myöhemmille kehittyneille analytiikka-/koneoppimisratkaisuille?
    - Käyttöönotto asiakkaalla? Asiakkaan muutostoiveiden toteuttamisen helppous?
- Onko jotain muuta datan visualisointityökalua mitä kannattaisi diplomityössä käsitellä?