Tampereen yliopisto

Tino Kaartovuori

OSTOPROSESSIN JA TUOTETIEDON DIGITALISOINNIN ARKKITEHTUURISUUNNITELMA PK-YRITYKSESSÄ

Tapaus Rengas Turku Oy

Diplomityö
Informaatioteknologian ja viestinnän tiedekunta
Tarkastaja: Kati Kuusinen
Tarkastaja: Timo Poranen
Joulukuu 2025

TIIVISTELMÄ

Tino Kaartovuori: Ostoprosessin ja tuotetiedon digitalisoinnin arkkitehtuurisuunnitelma PK-yrityksessä
Diplomityö
Tampereen yliopisto
Tietotekniikan DI tutkinto-ohjelma
Joulukuu 2025

Tämän työn tavoitteena oli suunnitella arkkitehtuuri, jonka avulla Rengas Turku Oy voi digitalisoida ostoprosessinsa ja tuotetiedon hallintansa PK-yrityksen resursseihin sopivalla tavalla. Taustalla oli tilanne, jossa toimittaja- ja ostodata on hajallaan, suuri osa työstä tehdään manuaalisesti ja päätöksenteko nojautuu vahvasti yksittäisten asiantuntijoiden hiljaiseen tietoon. Tavoitteena oli löytää ratkaisu, jossa järjestelmä tukee päätöksiä, tilausmääriä ja hinnoittelua, mutta lopullinen päätöksenteko säilyy ihmisellä.

Lähestymistapana oli konstruktiivinen tapaustutkimus. Ostoprosessi käytiin läpi vaihe vaiheelta ja sen pohjalta määriteltiin vaatimukset arkkitehtuurille, integraatioille ja tiedon hallinnalle. Kirjallisuuden ja empiiristen havaintojen perusteella muodostettiin muutamia ratkaisuvaihtoehtoja, joista osa esikarsittiin pois ennen monikriteeristä vertailua. Jäljelle jääneitä vaihtoehtoja, mikropalvelua ja Django-laajennusta, arvioitiin monikriteerisellä päätösmenetelmällä, ja valitun ratkaisuvaihtoehdon toteutuskelpoisuutta tarkennettiin rajatulla integraatiolla yhden toimittajan aineistoon. Tarkoitus ei ollut rakentaa valmista järjestelmää, vaan kuvata arkkitehtuuri, jonka varaan käytännön toteutus voidaan rakentaa.

Työn tuloksena syntyi kokonaisratkaisu, joka perustuu kanoniseen tuotemalliin, erilliseen ostopalveluun ja kahdella tavalla toteutettuun tiedonsiirtoon: eräsiirtoihin ja sisäiseen rajapintaan. Keskeinen ajatus on, että tuotetieto ja toimittajadata kootaan yhteen paikkaan ja oston rutiinityötä voidaan automatisoida osittain, jolloin järjestelmä tuottaa ehdotuksia ja ihminen tekee lopulliset valinnat. Arkkitehtuurivertailun perusteella erillinen ostopalvelu osoittautui pitkän aikavälin tarpeisiin sopivammaksi kuin nykyisen modulaarisen monoliitin laajentaminen.

Työ tuottaa kolme keskeistä kokonaisuutta: arkkitehtuurisuosituksen PK-yrityksen käyttöön, rakenteen kanonisen tuotemallin ja integraatioiden toteuttamiselle sekä vaiheittaisen etenemismallin tulevaa kehitystä varten. Työ osoittaa, että ostoprosessin digitalisointi voidaan toteuttaa vaiheittain ilman raskasta järjestelmäuudistusta ja että ratkaisu voi vähentää manuaalista työtä sekä parantaa tuotetiedon yhtenäisyyttä yrityksen työssä.

Avainsanat: ostoprosessi, tuotetiedon hallinta, integraatiot, kanoninen tuotemalli, PK-yritys, mikropalvelu, modulaarinen monoliitti

Tämän julkaisun alkuperäisyys on tarkastettu Turnitin Originality Check -ohjelmalla.

ABSTRACT

Tino Kaartovuori: Architectural Design for Digitalizing the Purchasing Process and Product Information in an SME
Master's Thesis
Tampere University
Master's Programme in Information Technology
December 2025

The aim of this thesis was to design an architecture that enables Rengas Turku Oy to digitalize its purchasing process and product information management in a way that fits the resource constraints of an SME. The starting point was a situation where supplier and purchasing data are scattered across multiple sources, a large share of the work is done manually, and decisions rely heavily on the tacit knowledge of individual experts. The goal was to outline a solution in which the system supports decisions, order quantities and pricing, while the final responsibility for decisions remains with the human user.

The chosen research approach was a constructive case study. The purchasing process was examined step by step, and the findings were used to derive requirements for the system architecture, integrations and data management. Based on literature and empirical observations, several solution alternatives were formed, some of which were screened out before a multi-criteria comparison. The remaining options, a microservice and an extension to the existing Django system, were evaluated using a multi-criteria decision-making method, and the feasibility of the selected direction was clarified through a limited integration test using data from a single supplier. The intent was not to build a production-ready system, but to define an architecture that can serve as the foundation for implementation.

The outcome is a holistic solution built around a canonical product model, a dedicated purchasing service and two complementary modes of data exchange: batch transfers and an internal API. The core idea is to consolidate product and supplier data into a single structure and to automate routine purchasing tasks to a degree that allows the system to generate suggestions while the human user makes the final choices. Based on the architectural comparison, an independent microservice was found to be better suited to the company's long-term needs than extending the existing modular monolith.

The thesis provides three main contributions: an architecture recommendation for SME use, a structural foundation for implementing a canonical product model and system integrations, and a phased adoption model to guide future development. The work demonstrates that digitalizing the purchasing process can be achieved incrementally without a large-scale system overhaul, and that such a solution can reduce manual work and improve the consistency of product information in day-to-day operations.

Keywords: purchasing process, product information management, integrations, canonical product model, SME, microservice, modular monolith

The originality of this thesis has been checked using the Turnitin Originality Check service.

TEKOÄLYN KÄYTTÖ OPINNÄYTTEESSÄ

Opinnäytteessäni on käytetty tekoälysovelluksia:

☐ Ei

☒ Kyllä

Ilmoitukseni mukaan olen käyttänyt opinnäytteessäni tutkielmaprosessin aikana seuraavia tekoälysovelluksia:

Tekoälysovellusten nimi ja versio:

ChatGPT (GPT-5), OpenAI

Käyttötarkoitus:

Tekoälyä on käytetty kirjoitusprosessissa apuna, erityisesti tekstin jäsentelyssä, kulun selkeyttämisessä sekä ilmaisun ja sanavalintojen parantamisessa. Tekoälyn avulla on hahmoteltu työn rakennetta isommassa kuvassa, ja tekstiä on pyritty muotoilemaan asialliseksi. Lisäksi tekoälyltä on usein pyydetty aiheeseen tai kappaleeseen liittyviä huomioita ja kysymyksiä, jotta työn keskeiset ajatukset ja näkökulmat saadaan paremmin esiin. Tekoälyä on hyödynnetty ikään kuin “haastattelijana” aiheiden, havaintojen ja kokemusten käsittelyssä, jonka pohjalta varsinainen analyysi ja tulkinnat on saatu helpommin ja laadukkaammin tehtyä itse. Tekoälyä ei ole käytetty itse analyysin tekemiseen, vaihtoehtojen arviointiin tai ratkaisujen valintaan, vaan ainoastaan lisänäkökulmien ja omien ajatusten esiin nostamiseen. Tekoälyltä on pyydetty myös yksittäisiä lähde-ehdotuksia, mutta ne on tarkistettu ja luettu aina erikseen käyttäen alkuperäistä teosta. Työn englannin kielisen tiivistelmän kääntämiseen on käytetty apuna tekoälyä. Järjestelmään ei ole syötetty yrityksen luottamuksellisia tietoja tai henkilötietoja.

Osiot, joissa tekoälyä on käytetty:

Tekoälyä on käytetty koko työn laajuudessa aiemmalla kuvatulla tavalla.

Olen tietoinen siitä, että olen täysin vastuussa koko opinnäytteeni sisällöstä, mukaan lukien osat, joissa on hyödynnetty tekoälyä, ja hyväksyn vastuun mahdollisista eettisten ohjeiden rikkomuksista.

ALKUSANAT

Tämä työ on tehty Rengas Turku Oy:lle vuoden 2025 aikana. Aihe oli alusta asti mielenkiintoinen, ja sen tutkiminen ja tekeminen tuntui luontevalta. Opintojeni varrelle on mahtunut takaiskuja, mutta siitä huolimatta onnistuin tässä työssä ja sain sen valmiiksi ja olen ylpeä siitä, millainen siitä lopulta tuli. Työn viimeiset kaksi viikkoa ovat olleet elämäni stressaavinta aikaa, mutta samalla myös palkitsevimpia. Toivon, että tästä työstä on jatkossa konkreettista hyötyä ja että työn pohjalta saamme Rengas Turku Oy:n ostoprosessin uudistettua järkevälle tasolle.

Kiitos Rengas Turulle ja heidän ammattilaisilleen mielenkiintoisesta aiheesta ja kaikesta työn aikana saamastani avusta. Suuri kiitos kuuluu myös tarkastajille, joiden palautteen ja tuen avulla tämä työ saatiin lopulta maaliin.

Erityiskiitos vaimolleni, perheelleni ja ystävilleni, jotka ovat tukeneet minua diplomityön ja opiskeluideni eri vaiheissa. Ilman heidän apuaan on hyvin mahdollista, että tämä työ ei olisi koskaan valmistunut.

Salossa, 14.12.2025

Tino Kaartovuori

SISÄLLYSLUETTELO

1. JOHDANTO 1
2. TEOREETTINEN VIITEKEHYS JA RATKAISUJEN TAUSTAT 5
2.1 Tuotetiedon hallinta (PIM) 5
2.2 Ostohinnan muodostus ja hinnoittelun haasteet 7
2.3 Prosessien digitalisointi PK-yrityksessä 8
2.4 Päätöksenteon tukijärjestelmät (DSS) 10
2.5 Ohjelmistoarkkitehtuuri ja skaalautuvat ratkaisut PK-ympäristössä 11
2.6 B2B-integraatiokaavat 14
3. TUTKIMUSMENETELMÄT 17
3.1 Tutkimusasetelma ja lähestymistapa 17
3.2 Tutkimuskohde ja aineistonkeruu 18
3.3 Aineiston analyysi 20
3.4 Ratkaisuvaihtoehtojen kartoitus ja SAW-vertailun valmistelu 22
3.5 Ratkaisun tekninen kokeilu ja alustava arviointi 23
3.6 Yhteenveto 24
4. EMPIIRISET HAVAINNOT JA RATKAISUN VAATIMUKSET 25
4.1 Empiirisen tarkastelun tavoite ja rajaus 25
4.2 Ostoprosessin kuvaus 25
4.3 Kipupisteet ja niiden ilmeneminen 28
4.4 Toimittajaintegraatioprofiilit ja EDIWheel-havainnot 31
4.5 Yhteys teoreettiseen viitekehykseen ja ratkaisun vaatimukset 32
4.6 Yhteenveto 37
5. RATKAISUVAIHTOEHTOJEN VERTAILU JA VALINTA 39
5.1 Vaihtoehdot 40
5.2 Vertailukriteerit ja painot 44
5.3 Pisteytys ja tulokset 45
5.4 Valinta ja johtopäätökset 48
5.5 Riskit ja varasuunnitelma 49
6. VALITUN RATKAISUN SUUNNITTELU JA TESTAUS 51
6.1 Tavoitteet, rajaus ja oletukset 51
6.2 Kokonaisarkkitehtuuri ja integraatioperiaatteet 52
6.3 Kanoninen tuotemalli ja tiedonhallinta 53
6.4 Rajapinnat, taustaprosessit ja suorituskyky 56
6.5 Sääntö- ja valintamoottori 59
6.6 Käyttöliittymä ja käyttäjäpolut 63

6.7 Tietoturva, pääsynhallinta ja auditointi 64
6.8 Käyttöönotto, toimittajan lisääminen ja tuoteryhmien laajentaminen 65
6.9 Dokumentointi ja tiedonhallinnan ohjaus 68
7. TULOKSET JA ARVIOINTI 70
7.1 Vastaus tutkimuskysymyksiin 70
7.2 Menetelmän soveltuvuus ja luotettavuus 72
7.3 Ratkaisun vaikutukset ja liiketoiminnallinen potentiaali 73
7.4 Rajoitteet ja epävarmuustekijät 74
7.5 Johtopäätökset menetelmän soveltuvuudesta PK-ympäristössä 75
8. YHTEENVETO 76
LÄHTEET 78
LIITE 1. YRITYSPALAVERIT 81

LYHENTEET JA MERKINNÄT

API	Sovellusohjelmointirajapinta; työssä erityisesti REST-periaattein suunniteltu transaktiorajapinta.
REST	Verkkoarkkitehtuurin periaatteet HTTP-rajapinnoille; mm. tilattomuus ja yhtenäinen rajapintatapa.
PIM	Tuotetiedon keskitetty hallinta ja jakelu.
MDM	Ydintiedon hallinnan kokonaisuus, johon PIM kytkeytyy.
DSS	Päätöksenteon tukijärjestelmä, joka yhdistää dataa, malleja ja näkymiä asiantuntijan tueksi.
GDSS	Ryhmäpäätöksenteon tuki, esimerkiksi yhteiset näkymät ja äänestys.
SDLC	Perinteinen järjestelmäkehitysprosessi.
GS1	Kansainvälinen standardointiorganisaatio tuotetunnisteille ja EDI-sanomille.
GTIN	Yksiselitteinen tuotetunnus arvoketjussa.
EAN	Käytännössä GTIN-tunnistepohjainen viivakoodi tuotteen yksilöintiin.
EANCOM	GS1:n EDI-sanomastandardi, esim. PRICAT ja INVRPT.
PRICAT	Tuotekatalogi- ja hinnastosanoma eräsiirtoihin.
INVRPT	Varastosaldoraportin sanoma eräsiirtoihin.
EASY XML	Toimittajakohtainen XML-pohjainen rajapinta saatavuuskyselyihin ja tilauksiin.
EDI	Sähköinen sanomaliikenne massadatan vaihtoon (esim. hinnastot, varastosaldot).
Idempotentti	Toisto ei muuta tilaa enempää kuin kerran; tärkeä periaate erä- ja API-käsittelyssä.
Korrelaatio-ID	Korrelaatiotunniste, jonka avulla sama erä/tapahtuma voidaan tunnistaa ja välttää kaksoiskäsittely.
Django REST Framework	Djangoon perustuva REST-rajapintojen toteutuskehys (DRF).

MTV (Django)	Djangon arkkitehtuurimalli, joka vastaa käytännössä MVC-mallia.
Mikropalvelu	Itsenäisesti julkaistava palvelu omalla tietovarastolla ja rajapinnalla; skaalautuva mutta operointina raskaampi.
CI/CD	Jatkuvan integroinnin ja toimituksen käytännöt ja putket.
TCO	Elinkaarikustannusten kokonaisuus päätöksenteossa.
Adapteri	Toimittajakohtainen sovitin, joka muuntaa sanomat kanoniseen malliin ja takaisin.
Kanoninen	Yhtenäinen sisäinen tietomalli, johon eri lähteiden toimittajadata normalisoidaan.
Push	Toimittajan eräsiirto massadatana.
Pull	Vastaanottajan transaktio-API-kutsu ajantasaisen tiedon hakemiseen.
ERP	Toiminnanohjausjärjestelmä, jota käytetään mm. hintojen ja varastosaatavuuden lähteenä.
UI	Käyttöliittymä, jonka kautta käyttäjä käyttää järjestelmää.
TLS	Salausprotokolla, joka suojaa verkkoyhteyden (esim. Nettivanne ↔ ostotyökalu).

1. JOHDANTO

Rengas Turku Oy on vähittäis- ja palveluliiketoiminnassa toimiva yritys, jonka toiminta on sesonkivetoista ja kysyntä vaihtelee kausittain. Yrityksen ostoprosessille ja tuotteiden hallinnalle asettuu korkeat vaatimukset laajan tuotekirjon ja nopean reagointitarpeen vuoksi. Yrityksellä on käytössä digitaalinen ydinjärjestelmä, joka palvelee esimerkiksi myyntiä, varastonhallintaa ja laskutusta. Ostoprosessi on kuitenkin edelleen ydinjärjestelmän ulkopuolella: keskeiset työvaiheet hoidetaan taulukko-ohjelmalla, tilaukset lähetetään sähköpostitse ja päivitykset kirjataan manuaalisesti tuote kerrallaan järjestelmään. Lisäksi ostot perustuvat yhden ammattilaisen hiljaiseen tietoon. Tämä ei skaalaudu PK-yrityksen nykyiseen volyymiin: läpimenoajat ovat pitkiä, virheiden määrä lisääntyy ja päätöksentekoa ei voida jakaa useamman asiantuntijan kesken. Ratkaisua haetaan digitalisoimalla ja uudistamalla yrityksen ostoprosessi.

Kirjallisuudesta löytyy tukea edellä kuvattujen ongelmien jäsentämiseen. Ostoprosessin digitalisoinnin kannalta keskeisiä teemoja ovat tuotetiedon keskittäminen tuotetiedon hallinnan (Product Information Management, PIM) periaatteiden mukaisesti, useilta toimittajilta ostettavien tuotteiden harmonisointi kanoniseen eli yhtenäiseen sisäiseen tuotemalliin sekä toimittajien integraatiot. Tämän kaltaisissa integraatioissa hyödynnetään yleensä kaksikanavaista polkua, jossa yhdistetään eräsiirrot ja sovellusohjelmointirajapinnan (Application Programming Interface, API) kutsut. Tuotteiden hinnoittelun kannalta luonteva tulevaisuuden toimintatapa on malli, jossa järjestelmä laskee ja valvoo ja ostosta vastaava asiantuntija tekee lopulliset päätökset. Tämä vastaa käytännössä ihminen silmukassa -periaatetta (Subramania & Khare, 2011), jossa päätöksenteon tukijärjestelmä (Decision Support System, DSS) tukee päätöstä eikä korvaa päätöksentekijää. Lisäksi eteneminen on järkevää vaiheistaa yrityksen digitaalisen kypsyystason perusteella, sillä irralliset sovellushankinnat tai ylisuurten projektien käynnistäminen voivat johtaa kustannusylityksiin ja viivästyksiin (Vial, 2019).

Tutkimuskysymys ja rajaus

Tässä työssä tarkastellaan, miten Rengas Turku Oy:n ostoprosessia voidaan digitalisoida PK-yrityksen reunaehdot huomioiden siten, että manuaalityö ja virheriski vähenevät, läpimenoaika lyhenee ja tuotetiedon laatu sekä hinnoittelun tuki paranevat. Päätutkimuskysymys ja sitä täydentävät tukikysymykset on koottu luvun lopussa esitettyyn kysymyslistaan.

Työn tarkastelu rajataan ostoprosessin niihin osiin, joissa työmäärä ja virheriski ovat suurimmat. Painopiste on massatilauksissa ja sesonkivalmistelussa, koska ne vievät eniten aikaa, sisältävät suurimmat määrät tuotteita ja yhdistävät useita eri työvaiheita, taulukoita ja työkaluja. Päivittäistä ostoa ja yksittäisiä täydennystilauksia tarkastellaan yhtenä kokonaisuutena, koska ne ovat käytännössä sama prosessi ja sisältävät pääasiassa samoja työvaiheita ja ongelmia. Näissä korostuvat erityisesti hinnoittelu, tuotetietojen hallinta ja ostotilausten hallinnointi.

Tarkastelu rajataan renkaiden ostoon, koska ne muodostavat liiketoiminnan suurimman osan ja renkaisiin liittyvä muutos on myöhemmin sovellettavissa muihin tuoteryhmiin. Arkkitehtuurin ja integraatioiden osalta työssä vertaillaan kahta päävaihtoehtoa: nykyisen modulaarisen monoliitin laajentamista ja erillisen mikropalvelun (microservice) toteuttamista. Lisäksi painopiste on kanonisen tuotemallin ja toimittajaintegraatioiden kokonaisuudessa. Tavoitteena ei ole käydä läpi kaikkia mahdollisia teknisiä variaatioita, vaan löytää PK-yrityksen arkeen sopiva etenemispolku.

Työn luonne ja lähestymistapa

Työ on luonteeltaan ensisijaisesti suunnittelua. Lähestymistapana on konstruktiivinen tapaustutkimus, jossa yhdistyvät käytännön kehittämistyö ja teoreettinen perustelu. Yrityksen nykytilaa on arvioitu havainnoinnin ja asiantuntijakeskustelujen avulla, ja kirjallisuudesta on johdettu suunnittelun periaatteet arkkitehtuurille, datamallille ja integraatioille. Ratkaisun uskottavuutta on alustavasti testattu prototyypillä ja rajatuilla integraatiokokeiluilla yhtä toimittajaprofiilia vasten. Näiden menetelmien tarkempi kuvaus esitetään luvussa 3, ja tekninen kokeilu kuvataan erikseen luvussa 3.5. Tässä työssä keskitytään ratkaisun suunnitteluperiaatteisiin. Toteutuksen yksityiskohtia käsitellään vain siltä osin kuin ne ovat välttämättömiä johtopäätösten kannalta, ja tavoitteena on esittää yritykselle perusteltu etenemispolku, ei täydellinen tekninen toteutusdokumentaatio.

Työn kontribuutiot

1. Arkkitehtuurisuositus PK-ympäristöön, joka valitaan yritykselle sopivien arvojen ja strategisen suunnan pohjalta.
2. Kanonisen tuotemallin ja integraatioperiaatteiden kokonaisuus, joka mahdollistaa toimittajakohtaisen datan yhdenmukaisen käsittelyn ja jäljitettävyyden sekä sovellettavuuden kaikkiin toimittajiin.
3. Suuntaviivat jatkokehitykselle toteutuksen eri osa-alueille. Näitä on esimerkiksi taustaprosessit, käyttöliittymä, tietoturva ja testaus. Lisäksi luonnos päätöksenteon tukijärjestelmästä, joka noudattaa ihminen silmukassa -periaatetta.

Rengas Turku Oy toimii rengas- ja vannemyynnissä sekä muissa autoalan palveluissa. Yrityksellä on useita kivijalkaliikkeitä, ja myynti tapahtuu sekä myymälöissä että verkkokaupassa. Asiakaskunta koostuu pääosin kuluttajista. Merkittäviä asiakasryhmiä ovat lisäksi autoliikkeet ja yritysasiakkaat. Tässä työssä tarkastelun kohteena ovat ennen kaikkea renkaat, koska ne muodostavat liiketoiminnan suurimman osan. Lisäksi ostoon liittyvä muutos renkaiden osalta on myöhemmin sovellettavissa muihin valikoiman tuotteisiin. Vanne- ja tarvikeostot huomioidaan rajatusti.

Kysyntä on selvästi sesonkivetoista. Kevät- ja syyskausina myynti voi jopa moninkertaistua, mikä korostaa ennakoinnin merkitystä valikoiman suunnittelussa ja tilauksissa. Tuotevalikoima on pysynyt kaudesta toiseen melko samanlaisena, mutta yrityksen saamien ostotarjousten mukaan myynti yleensä keskittyy kausittain hieman eri tuotemerkkeihin ja malleihin. Varastomäärät ja koostumus vaihtelevat sesongin vaiheen mukaan. Yrityksellä on useita myyntipisteitä, ja rengasostoista vastaava henkilö hoitaa tilaukset kaikkiin varastoihin. Tämä korostaa valikoiman ja määrien suunnittelua toimipistekohtaisesti. Ostoprosessin kannalta tarvitaan yhtenäinen näkymä toimipisteiden myyntiin ja menekkiin riittävällä tarkkuudella ja ajantasaisesti, mikä asettaa vaatimuksia myös päätöksenteon tuelle.

Renkaiden hankintaverkosto koostuu useasta päätoimittajasta. Tässä diplomityössä syvennytään yhteen toimittajaprofiiliin esimerkinomaisesti, jotta integraatio- ja tietomalliratkaisut voidaan kuvata ilman, että yrityssalaisuuksiin tai toimittajakohtaisiin sopimusyksityiskohtiin tarvitsee paneutua. Toimitusmallit kattavat sekä keskitetyn vastaanoton että suorat toimipistekohtaiset toimitukset. Molempien mallien tukeminen on käytännössä välttämätöntä sesonkien kuormitushuippujen ja toimitusikkunoiden vuoksi. Näin ratkaisut pysyvät toteutuskelpoisina arjessa.

Yrityksen järjestelmäkokonaisuuden ydin on Python–Django-pohjainen yrityksen oma järjestelmä ja sen kanssa tiiviisti integroitunut Nettivanne-verkkokauppa. Kokonaisuus on rakennettu yrityksen tarpeisiin, ja se palvelee myyntiä, varastoa, rengashotellia ja laskutusta. Ostotoiminnan digitalisointi kytketään tähän järjestelmään siten, että nykyiset tietomallit ja integraatioperiaatteet säilyvät johdonmukaisina. Tavoitteena on välttää päällekkäistä tietoa ja mahdollistaa vaiheittainen käyttöönotto ilman häiriöitä päivittäisessä työssä. Tällä tavalla minimoidaan riskit ja tuetaan jatkuvaa kehittämistä.

Tutkimuskysymykset ovat seuraavat:

Pääkysymys

”Miten Rengas Turku Oy:n ostoprosessia voidaan digitalisoida siten, että manuaalityö, virheriski ja läpimenoaika vähenevät, tuotetiedon laatu paranee ja hinnoittelua voidaan tukea päätöksenteon tukijärjestelmällä PK-yrityksen reunaehdot huomioiden?”

Tukikysymykset

1. Mitkä ovat nykyisen ostoprosessin keskeiset kipupisteet ja digitalisoinnin tarpeet
2. Millä kriteereillä arkkitehtuurivaihtoehtoja verrataan ja valitaan PK-yrityksen näkökulmasta
3. Miten kanoninen tuotemalli suunnitellaan niin, että toimittajakohtainen tieto voidaan käsitellä yhdenmukaisesti, tuotteiden jäljitettävyys säilyy ja sama rakenne tukee sekä ostoprosessin digitalisointia, että hinnoittelua
4. Miten integraatiot (eräsiirrot ja rajapintakutsut) toteutetaan siten, että ne tukevat ostoprosessin digitalisointia, siten että tuotetiedon käsittely on yhdenmukaista ja muutokset ovat jäljitettävissä
5. Miten käyttöönotto vaiheistetaan, jotta riski pysyy hallittuna ja ratkaisu skaalautuu ilman häiriöitä

Työn rakenne on seuraava:

Luku 2 kokoaa teoreettisen viitekehyksen, eli tiivistelmän keskeisestä kirjallisuudesta (tuotetiedon hallinta, hinnoittelun erityiskysymykset, prosessien digitalisointi PK-yrityksessä, päätöksenteon tuki, arkkitehtuurivaihtoehdot ja push- ja pull-virrat). Luku 3 kertoo, miten aineisto kerättiin ja analysoitiin. Luvussa 4 raportoidaan empiiriset havainnot ostoprosessista ja niiden yhteys teoriaan. Luvussa 5 vertaillaan ratkaisuvaihtoehdot ja esitetään valinnat. Luku 6 kuvaa valitun ratkaisun kokonaisuuden ja suuntaviivat kehitykselle: kokonaisarkkitehtuuri, tietomalli, rajapinnat, taustaprosessit, käyttöliittymä, tietoturva, suorituskyky, käyttöönotto ja dokumentointi. Luvussa 7 arvioidaan tulokset ja luvussa 8 esitetään yhteenveto, jossa kootaan päätulokset ja jatkotoimet. Rakenne on valittu tukemaan vaiheittain etenevää toteutusta PK-yrityksen reunaehdoissa.

2. TEOREETTINEN VIITEKEHYS JA RATKAISUJEN TAUSTAT

Seuraavaksi kootaan työn kannalta olennainen teoria ja käsitteet. Tavoitteena on, että työssä esitetyt valinnat ja väittämät saavat tukea tässä luvussa kootusta kirjallisuudesta. Ensin kuvataan tuotetiedon hallinnan ja tiedonlaadun periaatteet, sitten ostohinnan muodostuminen ja ostettujen tuotteiden hinnoittelun keskeiset kysymykset. Tämän jälkeen käsitellään PK-yrityksen prosessien digitalisointia sekä päätöksenteon tukea. Samalla avataan ajattelutapa, jossa järjestelmä tuottaa perusteltuja ehdotuksia ja ihminen tekee lopulliset päätökset. Lopuksi vertaillaan modulaarisen monoliitin ja mikropalveluratkaisun vahvuuksia ja heikkouksia sekä käydään läpi ostoprosessille olennaiset integraatiokuviot.

2.1 Tuotetiedon hallinta (PIM)

Abraham (2014) määrittelee tuotetiedon hallinnan (Product Information Management, PIM) prosesseiksi ja järjestelmiksi, joilla yritys keskittää, hallinnoi ja rikastaa tuotteisiin liittyvää tietoa. PIM-järjestelmä toimii keskeisenä lähteenä, jonka avulla tuotetiedot esitetään johdonmukaisesti eri kanavissa ja jonka kautta tiedon jakelu ohjataan yhdenmukaiseksi (Abraham, 2014).

Abrahamin (2014) mukaan tuotetieto kattaa sekä tekniset ominaisuudet ja mitat että kaupalliset materiaalit, kuten kuvat ja tuotekuvaukset, joita PIM auttaa rikastamaan ja hallitsemaan. Hinnat ja varastosaatavuus eivät tyypillisesti synny PIM:ssä, vaan ne tuodaan lähdejärjestelmistä, kuten toiminnanohjauksesta (Enterprise Resource Planning, ERP) ja varasto- tai myymäläjärjestelmistä. Tiedon eheys, saatavuus ja luotettavuus ovat keskeisiä, ja PIM toimii keskitettynä “yhtenä totuuden lähteenä” (single source of truth) (Abraham, 2014).

PIM-järjestelmien keskeisiä toimintoja ovat (Abraham, 2014):

1. **Keskitetty tietovarasto**, johon tuotedata tallennetaan.
2. **Tietojen rikastaminen**, kuten kuvien ja markkinointitekstien lisääminen tuotetietoon.
3. **Laadunvarmistus ja validointi**, eli automaattiset tarkastukset ja korjaukset sekä yhtenäinen tiedon esitystapa.

4. **Monikanavainen tuotetiedon jakelu** esimerkiksi verkkokauppaan, kivijalkaliikkeille ja jälleenmyyjille.
5. **Integraatiot keskeisiin lähdejärjestelmiin** (esim. ERP, varasto, hankinta) ja ulkoisiin toimijoihin.

PIM kytkeytyy tiiviisti tiedonhallinnan ohjaukseen (data governance), jossa päätetään, mitä tuotetietoa otetaan mukaan, mihin tarkoituksiin sitä käytetään, kuka omistaa ja ylläpitää sitä, millaista laatua siltä vaaditaan, miten se nimetään, kuka saa siihen pääsyn sekä kuinka kauan tietoa säilytetään ja milloin se poistetaan. Ohjaus (governance) tarkoittaa näiden päätösten ja vastuiden jakamista, kun taas hallinta (management) on päätösten käytännön toimeenpanoa (Khatri & Brown, 2010).

Khatri ja Brown (2010) jakavat ohjauksen viiteen alueeseen: dataperiaatteet (data principles), datan laatu (data quality), metatieto (metadata), tietoon pääsy (data access) ja tiedon elinkaari (data lifecycle). Käytännössä organisaatio sopii ensin, mihin tuotetietoa tarvitaan ja millainen laatu on kunkin tarkoituksen kannalta riittävä. Sen jälkeen nimetään omistajat ja vastuuhenkilöt, kuvataan yhteinen käsitteistö ja perusrakenne, sovitaan pääsy- ja tietoturvasäännöistä sekä kirjataan, miten tieto tuotetaan, julkaistaan, arkistoidaan ja poistetaan. Tällöin PIM ei jää pelkäksi tekniseksi järjestelmäksi, vaan kiinnittyy ohjaukseen ja osaksi kokonaisprosessia. PIM varmistaa, että toimintatapa toistuu samankaltaisena eri tilanteissa ja että se voidaan tarvittaessa myös todentaa (Battistello et al., 2021; Khatri & Brown, 2010).

Wang ja Strong (1996) esittävät, että tiedon laatua voidaan tarkastella neljän osa-alueen avulla. Nämä ovat sisäinen laatu, kontekstinmukainen laatu, esitystavan laatu ja saatavuus (Wang & Strong, 1996). Tässä työssä tulkittuna sisäisellä laadulla tarkoitetaan, että tieto on oikein ja eheää. Käytännössä yksilöivät tunnisteet, tekniset mitat ja perusominaisuudet eivät sisällä virheitä. Kontekstinmukainen laatu korostaa relevanssia ja ajantasaisuutta. Esimerkiksi hinnan tulee vastata voimassa olevia kampanjoita ja toimitusaikoja. Esitystavan laadussa tärkeää on, että tieto on esitetty kaikissa kanavissa samalla tavalla, jolloin nimikkeet, yksiköt ja kenttien merkitykset pysyvät yhtenäisinä ja tietoa on helppoa ymmärtää ja myös vertailla. Saatavuus tarkoittaa, että tieto löytyy silloin kun sitä tarvitaan ja siihen on oikeille tahoille käyttöoikeudet (Wang & Strong, 1996).

GS1 on kansainvälinen standardointijärjestö, joka ylläpitää GTIN- ja EAN-standardeja sekä muita toimitusketjun tunnisteita ja viitekehyksiä. Kanonisen tuotemallin kannalta globaali tuotetunniste on keskeinen. Global Trade Item Number (GTIN) on GS1-järjestelmän käyttämä kansainvälinen tunniste, jonka avulla yksittäiset kauppayksiköt voidaan erottaa toisistaan. European Article Number (EAN) on viivakoodimuoto, joka kantaa

GTIN-tunnusta ja mahdollistaa koneluennan myynnissä ja logistiikassa. Kauppayksiköllä tarkoitetaan tuotetta tai pakkaustasoa, jota hinnoitellaan, tilataan tai laskutetaan omanaan, esimerkiksi yksittäinen rengas tai neljän renkaan sarja. GTINin tarkoitus on varmistaa, että tieto liikkuu yhdenmukaisena ja on seurattavissa koko toimitusketjun läpi (GS1, 2023).

Käytännössä yrityksen omat sisäiset tuotekoodit kytketään GTIN-tunnisteisiin, jotta eri lähteistä tuleva tieto yhdistyy samaan tuotteeseen eri integraatioissa. Tuotetiedon hallinta ja päädatan hallinta (Master Data Management, MDM) tarjoavat tälle keskitetyn toimintatavan, jossa tuotetieto kootaan, rikastetaan ja jaellaan kanaviin (Abraham, 2014; GS1, 2023). GTIN-hallintastandardi määrittelee, milloin tuotteelle annetaan uusi GTIN tai vaihdetaan olemassa oleva ja sääntöjä sovelletaan sekä kuluttajapakkaukseen että ylempiin pakkaustasoihin.

Yhteenvetona PIM toimii yrityksen tuotetiedon keskitettynä lähteenä. Se kokoaa, rikastaa ja välittää tuotetiedon eri kanaviin ja sijoittuu lähdejärjestelmien ja jakelukanavien väliin (Abraham, 2014). PIM-ympäristössä hyödynnetään GS1/GTIN-standardeja. GTIN yksilöi kauppayksiköt toimitusketjussa, ja GTIN-hallintastandardi ohjaa, milloin tunniste annetaan tai vaihdetaan eri pakkaustasoilla (GS1, 2023).

2.2 Ostohinnan muodostus ja hinnoittelun haasteet

Ostohinta ei synny pelkästä yksikköhinnasta, vaan useasta tekijästä. Kuten Ellram (1995) korostaa, pelkkä listattu hinta ei vielä kerro todellisia kustannuksia, vaan mukaan on otettava myös ostamiseen liittyvät muut kulut ja hyvitykset. Tässä työssä tämä tarkoittaa esimerkiksi alennuksia, kampanjoita, volyymibonuksia, kierrätysmaksuja sekä toimitus- ja käsittelykuluja. Kun plussat ja miinukset lasketaan yhteen, saadaan tuotekohtainen todellinen ostohinta, jota käytetään pohjana myyntihinnan määrityksessä.

Ellramin (1995) mukaan omistamisen kokonaiskustannus (Total Cost of Ownership, TCO) laajentaa pelkän hankintahinnan tarkastelun koko elinkaaren kuluihin. Mukaan lasketaan tilaus ja toimittajien kelpuutus, kuljetus, vastaanotto ja tarkastus, hylkäys- ja korvauserät, viiveistä aiheutuvat kulut sekä poistokustannukset. TCO toimii päätöksenteon kehikkona toimittajavaihtoehtojen vertailussa pelkän yksikköhinnan sijaan sekä toimittajan suoriutumisen arvioinnissa ja prosessiparannusten suunnittelussa (Ellram, 1995).

Ramakrishnan et al. (2019) osoittavat, että manuaalisesta hinnoittelusta voi aiheutua herkästi virheitä ja jo muutama väärin hinnoiteltu tuote voi heikentää kannattavuutta sekä asiakasluottamusta. Ympäristöissä, joissa on paljon tuotteita, tarvitaan kyky tunnistaa

poikkeamat nopeasti ja priorisoida korjaukset vaikutuksen mukaan, jotta kate ja hyvä asiakaskokemus säilyvät (Ramakrishnan et al., 2019).

Anta Callersten et al. (2024) kuvaavat käytännön mallin, jossa tekoäly tai sääntömoottori tuottaa tuotekohtaisia hintasuosituksia, mutta asiantuntija tekee lopullisen hyväksynnän. Raportoidut hyödyt näkyvät bruttokatteen paranemisena silloin, kun rinnalla on tähän toimintatapaan rakennetut tiimit, prosessit ja teknologia. Keskeistä ei ole pelkkä algoritmi vaan selkeä operointimalli ja hallittu käyttöönotto (Anta Callersten et al., 2024).

Gadde et al. (2024) osoittavat, että sääntöpohjaiset hinnoittelukäytännöt jäävät helposti jälkeen nopeasti muuttuvissa markkinaolosuhteissa, mikä perustelee adaptiivisempien menetelmien tarkastelua. Heidän mukaansa vahvistusoppimiseen perustuvat menetelmät voivat oppia hinnoittelupolitiikan ja säätää hintoja jatkuvan palautteen perusteella, mutta käyttöönotto on käytännössä raskas suurten datatarpeiden, mallivalvonnan, läpinäkyvyyden ja eettisten rajoitteiden sekä integraatiohaasteiden vuoksi.

PK-yrityksen hinnoittelun kannalta on käytännössä kolme vaihtoehtoa. Manuaalinen malli on altis virheille ja niiden liiketoimintavaikutuksille, mikä korostuu sesonkivaihteluissa ja laajoissa tuotekokonaisuuksissa (Ramakrishnan et al., 2019). Hybridimallissa sääntö- tai tekoälymoottori laskee ehdotukset ja asiantuntija vahvistaa ne, jolloin yhdistyvät tehokkuus, näkyvyys ja ohjattavuus sekä mahdollisuus huomioida TCO-tekijät järjestelmällisesti (Anta Callersten et al., 2024). Täysautomaattinen malli on periaatteessa mahdollinen, mutta käytännössä se edellyttää kypsää operointia, selkeitä sääntöjä, kattavia ja integroitavia tietolähteitä sekä mallien valvontaa ja tulkittavuutta, jotta ratkaisu toimii luotettavasti arjessa (Gadde et al., 2024).

TCO suuntaa tarkastelun kokonaiskustannuksiin ja tukee työn laskentasääntöjen määrittämistä. Kirjallisuus tarjoaa vaihtoehtoja hinnoittelun tehokkuuden parantamiseen ja riskien hallintaan manuaalisista malleista hybrideihin ja täysautomaatiolle. Näiden pohjalta työn ratkaisuvaihtoehdot voidaan mitoittaa yrityksen tarpeisiin.

2.3 Prosessien digitalisointi PK-yrityksessä

Digitalisaation odotetaan lisäävän tehokkuutta, parantavan läpinäkyvyyttä ja vähentävän riippuvuutta yksittäisistä ammattilaisista. Käytännössä etenemistä hidastavat usein rajalliset resurssit, osaamisvaje ja muutosvastarinta (Omowole et al., 2024).

Kirjallisuudessa tunnistetaan kaksi yleistä toimintatapaa, jotka heikentävät onnistumisen todennäköisyyttä PK-ympäristössä. Ensimmäinen on reaktiivinen ja pirstaleinen etene-

minen ilman selkeää tiekarttaa (Aliyev, 2025). Toinen on kerralla toteutettu laaja uudistus, joka kasvattaa riskejä, kustannuksia ja viivästyksiä (Aliyev, 2025). Molemmissa tapauksissa teknologia tulee ennen toimintamallia ja ohjausta.

Kirjallisuus suosittelee etenemistä vaiheittain yhden suuren kertatoteutuksen sijaan (Aliyev, 2025). Aliyevin (2025) mukaan työ alkaa siitä, että tunnistetaan eniten viivettä ja virheitä aiheuttavat kohdat ja automatisoidaan juuri ne. Pienet, näkyvät onnistumiset rakentavat luottamusta ja mahdollistavat seuraavat askeleet (Aliyev, 2025).

Omowole et al. (2024) kuvaavat käytännön toteutuksen koostuvan pilvipalveluista, analytiikasta ja automaatiosta. Pilvipalvelut tarjoavat skaalautuvan alustan integraatioille ja datan jakelulle, analytiikka muuntaa tapahtumatiedot päätöksiä tukeviksi havainnoiksi. Automaatio siirtää toistuvat ja virheherkät tehtävät sääntöjen tai mallien ohjatuiksi (Omowole et al., 2024). Nämä ajurit tukevat Aliyevin kuvaamaa vaiheittaista etenemistä, koska ne mahdollistavat kohdennetut, rajatut parannukset ilman laajaa kertatoteutusta ja luovat puitteet seuraavien askelten toistettavalle jatkokehitykselle (Aliyev, 2025; Omowole et al., 2024).

Organisaation muutos ei tapahdu kerralla. Vialin (2019) mukaan digitaalinen transformaatio etenee vaiheissa, joissa opitaan havaitsemaan muutostarpeita, tarttumaan mahdollisuuksiin ja uudistamaan tekemistä. Tavoite on, että teknologia, prosessit ja rakenteet tukevat arjen jatkuvaa parantamista (Vial, 2019). Ennen isompaa automaatiota on järkevää katsoa, missä kypsyystasossa nyt ollaan, ja valita seuraava realistinen askel sen perusteella (Vial, 2019). Tämä ajattelu on linjassa myös Aliyevin (2025) kanssa, joka korostaa pieniä, näkyviä edistysaskeleita ja vaiheittaista etenemistä.

Tämän työn kannalta PK-yritysten digitalisoinnin suurimmat riskit liittyvät pirstaleisuuteen ja ylisuuriin kertatoteutuksiin (Aliyev, 2025). Etenemisen on syytä olla hallittua ja nojata selkeään tiekarttaan, joka sitoo kehitysaskeleet liiketoiminnan tavoitteisiin ja aikatauluihin (Omowole et al., 2024). Yrityksen kypsyystaso tulee tunnistaa ennen laajempaa automaatiota, eli ensin arvioidaan nykytilanne ja valitaan seuraava realistinen askel (Vial, 2019). Tämä ajattelutapa on perusta PK-yrityksen onnistuneelle digitalisoinnille, jossa pienet näkyvät onnistumiset rakentavat luottamusta ja pilvipalvelut, analytiikka sekä automaatio tarjoavat käytännön työkalut eri vaiheiden toteuttamiseen (Aliyev, 2025; Omowole et al., 2024). Ratkaisut ja etenemissuunta valitaan vaikutuksen, riskin ja toteutettavuuden perusteella, jotta vältetään irralliset hankkeet ja varmistetaan, että teknologia, prosessit ja rakenteet etenevät oikeaan suuntaan (Vial, 2019; Omowole et al., 2024).

2.4 Päätöksenteon tukijärjestelmät (DSS)

Päätöksenteon tukijärjestelmä (Decision Support System, DSS) on vuorovaikutteinen järjestelmä, joka kokoaa dataa asiantuntijan harkinnan tueksi ilman, että se automatisoi päätöksentekoa (Kulkarni et al., 2007; Power, 2002). Powerin mukaan hyvä DSS tukee päätöksentekoa ja reagoi muuttuviin tarpeisiin esimerkiksi nopean prototypoinnin (rapid prototyping) kautta (Power, 2002). Kirjallisuudessa DSS-ratkaisut jaotellaan viiteen tyyppiin sen mukaan, mihin tiedonlähteeseen ja analyysiin ne nojaavat: viestintäpohjaisiin (communications-driven), datapohjaisiin (data-driven), asiakirjapohjaisiin (document-driven), tietopohjaisiin (knowledge-driven) ja mallipohjaisiin (model-driven) ratkaisuihin (Kulkarni et al., 2007; Power, 2002).

Viestintäpohjainen DSS tukee reaaliaikaista vuorovaikutusta organisaation jäsenten välillä (Power, 2002), esimerkiksi ostajan vuorovaikutusta toimittajien ja tiimin kanssa chat-sovelluksen avulla. Tietopohjainen DSS voi tarjota sääntöihin tai tietopohjiin perustuvia varoituksia, jos tilausehdot rikkovat hinnoittelu- tai kampanjasääntöjä (Power, 2002). Datapohjainen (data-driven) DSS kokoaa myynti-, varasto- ja toimittajadatan yhteen lähteeseen ja tarjoaa yhtenäisen näkymän päätöksenteon tueksi (Kulkarni et al., 2007; Power, 2002). Tämä tarkoittaa, että ostaja voi yhdestä näkymästä havaita esimerkiksi, että tietty tuotemalli on loppumassa yhdessä varastossa, sitä on edelleen saatavilla toisessa varastossa ja toimittaja tarjoaa lisäeriä kilpailukykyiseen hintaan. Mallipohjainen (model-driven) DSS voi hyödyntää optimointi- ja simulointimalleja päätöksenteossa (Power, 2002). Tällöin järjestelmä voi esimerkiksi laskea suositellun tilausmäärän huomioiden varastokapasiteetin ja kampanja-alennukset sekä näyttää, miten eri toimitukset vaikuttavat varaston kiertoon ja katteeseen, mutta lopullinen päätös jää ostajan vastuulle.

Lisäksi ryhmäpäätöksenteon tukijärjestelmä (Group Decision Support System, GDSS) tarjoaa mahdollisuuden jakaa päätöksentekoa useamman asiantuntijan kesken. DeSanctis ja Gallupe määrittelevät GDSS:n järjestelmäksi, joka yhdistää keskustelukanavan, laskentamoduulin ja päätöslokin helpottaakseen ryhmän työskentelyä (DeSanctis & Gallupe, 1987). Empiirinen näyttö GDSS:n käytöstä viittaa siihen, että anonyymi ideointi, jaettu näkymä ja äänestystyökalut lisäävät vaihtoehtojen määrää ja voivat nopeuttaa ratkaisun löytymistä (DeSanctis & Gallupe, 1987; Power, 2002).

Kehitymalliksi sopii nopea prototypointi (rapid prototyping), jota kirjallisuudessa kuvataan iteratiivisena ja joustavana lähestymistapana DSS-kehityksessä (Power, 2002). Tässä lähestymistavassa vaatimuksia tarkennetaan prototyyppien suunnittelun, testauksen ja käyttöönoton edetessä, ja ratkaisuja voidaan kokeilla pieninä osina ja hienosäätää niitä

muuttuvien tarpeiden mukaan. Power (2002) vertaa prototypointia perinteiseen järjestelmäkehityksen elinkaarimalliin (Systems Development Life Cycle, SDLC) ja korostaa prototypoinnin etuna joustavuutta DSS-vaatimusten muuttuessa.

Käytännössä DSS tukee oston suunnittelua ja päätöksentekoa neljällä toisiaan täydentävällä toiminnolla. Mallipohjaisessa DSS:ssä varasto- ja myyntihistoriaa voidaan hyödyntää tilausmäärien optimointiin (Kulkarni et al., 2007; Power, 2002). Tällainen suosittelumoduuli voi esimerkiksi ehdottaa, milloin tietyt tuotteet kannattaa tilata ja mihin varastoon. Datapohjainen DSS esittää historiallista dataa ja sen analyysia päätöksenteon tueksi (Power, 2002), jolloin esimerkiksi varasto- ja myyntigrafiikat voivat tehdä sesonkipiikit näkyviksi ja auttaa hahmottamaan, mitä tuotteita tulee tilata mihinkin aikaan. Mallipohjaisessa DSS:ssä skenaariolaskentaa (what-if-analyysiä) voidaan käyttää esimerkiksi arvioimaan ostohinnan muutoksen vaikutusta katteeseen ennen päätöksen tekemistä (Power, 2002). Tarvittaessa kevyt GDSS-kerros, jossa on anonyymit kommentit ja äänestys, nopeuttaa yhteisen näkemyksen saavuttamista useamman päätöksentekijän kesken (DeSanctis & Gallupe, 1987; Power, 2002). Yhdessä nämä toiminnot toteuttavat mallin, jossa ihminen on silmukassa: järjestelmä hoitaa datan jalostamisen ja suositusten laskennan, mutta lopullinen ratkaisu perustuu asiantuntijan harkintaan.

Yhteenvetona päätöksenteon tukijärjestelmät yhdistävät data-, malli- ja ryhmätyökerrokset yhdeksi kokonaisuudeksi, jonka avulla päätöksentekijä saa ajantasaisen tilannekuvan, suosituksia ja anonyymin keskustelukanavan päätösten viimeistelyyn. Näiden pohjalta voidaan puhua periaatteesta, jossa ihminen on silmukassa, koska järjestelmä jalostaa tiedon ja tuottaa suosituksia, mutta lopullinen päätös jää silti asiantuntijalle. Powerin (2002) kuvaama nopea prototypisointi tukee tällaista vaiheittaista toteutusta ja joustavaa kehittämistä DSS-ratkaisuissa. Tässä työssä lähestymistapa mahdollistaa ratkaisun säätämisen sesonkien ja liiketoimintatarpeiden muuttuessa.

2.5 Ohjelmistoarkkitehtuuri ja skaalautuvat ratkaisut PK-ympäristössä

Tässä työssä tarkasteltu ohjelmisto perustuu Python–Django-pinoon. Django hyödyntää malli–pohja–näkymä-mallia (Model–Template–View, MTV), joka vastaa käytännössä perinteistä malli–näkymä–kontrolleri-mallia (Model–View–Controller, MVC). Malli kuvaa tietorakenteet, näkymä käsittelee sovelluslogiikkaa ja pohja vastaa käyttöliittymän esityksestä (Holovaty & Kaplan-Moss, 2009). Käytännössä sovellus voidaan pilkkoa omiksi Django-sovelluksiksi (apps), joista jokainen vastaa yhdestä osa-alueesta, kuten ostosta, varastosta tai raportoinnista (Holovaty & Kaplan-Moss, 2009). Tällainen rakenne muo-

dostaa modulaarisen monoliitin, eli yhden sovelluskokonaisuuden, jossa osat on sisäisesti erotettu toisistaan ja koodi on loogisesti jaettu komponentteihin (Richards & Ford, 2020).

Kirjallisuudessa modulaarinen monoliitti ajatellaan PK-yritykselle usein sopivana alun arkkitehtuurivalintana, koska se pienentää alkuvaiheen monimutkaisuutta hallinnon ja infrastruktuurin näkökulmasta verrattuna mikropalveluarkkitehtuuriin (Richards & Ford, 2020; Newman, 2021). Modulaarisessa monoliitissa sovellus pyörii yhtenä prosessina, eikä kehittäjien tarvitse heti alkuvaiheessa käyttää aikaa erillisten palvelin- ja orkestrointikerrosten suunnitteluun ja ylläpitoon, joita hajautettu arkkitehtuuri vaatii. Selkeä jako Django-sovelluksiin helpottaa myös käytännön ylläpitoa ja jättää samalla tarkoituksella oven auki myöhemmälle jaolle erillisiksi mikropalveluiksi, jos tarve kasvaa (Holovaty & Kaplan-Moss, 2009; Richards & Ford, 2020). Toisaalta modulaarinen monoliitti ei mikropalveluarkkitehtuuriin verrattuna tarjoa yhtä hienojakoista palvelukohtaista skaalautuvuutta eikä yhtä suurta vapautta valita eri teknologioita, koska järjestelmä skaalautuu pääasiassa yhtenä kokonaisuutena (Richards & Ford, 2020). Newman (2021) tarkastelee kirjassaan ennen kaikkea mikropalveluarkkitehtuuria, mutta korostaa silti, että monoliitti on käytännössä “sensible default”. Hänen mukaansa monoliittinen ratkaisu on monessa tilanteessa järkevä aloituspiste, josta poiketaan vasta, kun pilkkomiselle on selkeä peruste ja mikropalvelujen lisäarvo ylittää hajauttamisen kustannukset (Newman, 2021).

Mikropalveluarkkitehtuuri tarjoaa toisenlaisen tavan kehittää ja skaalata järjestelmää. Sovellus jaetaan pieniksi, itsenäisesti julkaistaviksi palveluiksi, joilla on selkeä ja teknologisesti riippumaton rajapinta. Mikropalveluilla on usein myös oma tietovarasto, jolloin tiedot eriytyvät niin sanotun “data isolation” -periaatteen mukaisesti. Systemaattiset katsaukset ja koosteartikkelit, kuten Pahl ja Jamshidi (2016) sekä Richards ja Ford (2020), kuvaavat, että mikropalvelut voivat lisätä skaalautuvuutta, vikaeristystä ja teknologiavalintojen vapautta erityisesti pilvi- ja konttiympäristöissä. Pilvi- ja konttikontekstissa korostuvat ennen kaikkea mahdollisuus skaalata vain kuormittuneimpia palveluja, eristää viat yksittäisiin palveluihin sekä hyödyntää eri teknologioita eri osissa järjestelmää (Pahl & Jamshidi, 2016; Richards & Ford, 2020). Lisäksi kirjallisuudessa kuvataan vaiheittainen etenemismalli, jossa uusia ominaisuuksia toteutetaan mikropalveluina olemassa olevan monoliitin rinnalle. Näin järjestelmän osa-alueita voidaan uudistaa ja vaihtaa asteittain ilman, että koko ydinjärjestelmää täytyy kirjoittaa kerralla uudestaan, ja legacy-koodia voidaan purkaa ja uudistaa osissa (Newman, 2021; Taibi et al., 2017). Käytännössä tämä tarkoittaa sitä, että vanhan ja uuden arkkitehtuurin on elettävä jonkin aikaa rinnakkain.

Mikropalveluarkkitehtuuri hyödyt eivät kuitenkaan toteudu ilman alkuvaiheen lisäponnistelua. Taibi et al. (2017) tarkastelevat yrityksiä, jotka ovat siirtäneet monoliittisia järjestelmiään mikropalveluiksi, ja raportoivat, että kehitys- ja operointityö kasvaa siirtymävaiheessa. Jokainen palvelu pitää liittää erikseen jatkuvan integroinnin ja jatkuvan toimituksen putkiin, testaukseen ja seurantaan. Organisaation on usein otettava käyttöön uusi kehitys- ja operointimalli, kuten DevOps, sekä rakennettava palvelukohtaiset CI/CD-putket (Continuous Integration / Continuous Delivery) (Taibi et al., 2017). Pahl ja Jamshidi (2016) yhdistävät mikropalvelut nimenomaan pilvipohjaiseen, jatkuvaan kehittämiseen, jossa automaattinen testaus, käyttöönotto ja valvonta ovat keskeisiä osia koko arkkitehtuuria. Sama suunta näkyy myös Newmanilla (2021), joka korostaa, että mikropalveluista saatava hyöty perustuu pitkälti siihen, että yksittäiset palvelut voidaan julkaista usein ja pienissä erissä automatisoitujen putkien kautta. Käytännössä tämä tarkoittaa sitä, että mikropalveluarkkitehtuuri lisää työtä ja säätöä erityisesti alkuvaiheessa, mutta rakentaa samalla ylläpitoa helpottavan pohjan. Kun CI/CD-putket, valvonta ja lokitus ovat kunnossa ja toiminnassa, muutoksia voidaan tehdä turvallisesti ja palvelukohtaisesti, mikä tukee pitkän aikavälin ylläpidettävyyttä (Pahl & Jamshidi, 2016; Newman, 2021; Taibi et al., 2017).

Arkkitehtuuriin on yhdistettävissä myös ulkoisia hankittavia ratkaisuja. Sommerville (2016) jakaa kaikki hankittavat ohjelmistot kolmeen pääryhmään: valmiisiin valmisohjelmistoihin, konfiguroitaviin sovelluksiin, sekä erikseen räätälöityihin ratkaisuihin. Valmisohjelmistot ja säädettävät ratkaisut voivat lyhentää käyttöönottoa, mutta vaativat usein sitä, että vaatimuksia säädetään hankittavan ratkaisun mukaan. Lisäksi ulkopuoliset ratkaisut tuovat mukanaan toimittajariippuvuuden ja lisenssi- sekä palvelumaksuihin liittyvät kustannukset (Sommerville, 2016). Räätälöity ratkaisu tarjoaa enemmän joustavuutta ja paremman istuvuuden prosesseihin, mutta kasvattaa kehitystyötä ja ylläpitovastuuta. Tässä työssä arkkitehtuurivalinnan ohella tarkastellaan myöhemmin myös sitä, onko luontevampaa kehittää oma ratkaisu vai hyödyntää ulkoista työkalua.

Kustannus–hyötynäkökulmasta kirjallisuus tukee kaksivaiheista etenemismallia PK-yritykselle. Ensimmäisessä vaiheessa olemassa oleva monoliittinen sovellus selkeytetään siten, että se on aidosti modulaarinen, eli datamallit ja logiikka kootaan selkeisiin sovellusmoduuleihin ja tärkeimmät toiminnot avataan rajapintojen kautta (Richards & Ford, 2020). Tämä luo pohjan, josta korkeimman kuorman tai nopeimmin kehittyvät osa-alueet voidaan myöhemmin muuntaa mikropalveluiksi ilman jättimäistä kertamuutosta. Ajattelu on linjassa Richardsin ja Fordin (2020) sekä Newmanin (2021) esittämien näkemysten kanssa modulaarisesta monoliitista luontevana lähtökohtana. Toisessa vaiheessa, kun

kuormapiikit tietyille monoliitin osa-alueille kasvavat liian suuriksi tai tarve nopealle, itsenäiselle kehitykselle korostuu, kyseiset toiminnot voidaan irrottaa konttipohjaisiksi mikropalveluiksi ja varustaa automaattisella skaalauksella PK-yrityksen tarpeiden mukaan (Pahl & Jamshidi, 2016; Irufaan et al., 2021). Taibi et al. (2017) kuvaavat, että tällainen vaiheittainen eteneminen vähentää riskiä ja mahdollistaa sen, että legacy-monoliittia ja uusia mikropalveluja voidaan käyttää rinnakkain niin kauan kuin se on liiketoiminnan kannalta järkevää.

Yhteenvetona tämä osio kokoaa ohjelmistoarkkitehtuurin ja hankintamallien kannalta keskeiset teoriat, jotka toimivat pohjana työn myöhemmälle pohdinnalle. Modulaarisen monoliitin ja mikropalveluarkkitehtuurin vertailu auttaa jäsentämään, millaisia etuja ja rajoitteita eri vaihtoehtoihin liittyy erityisesti PK-yrityksen näkökulmasta. Sommervillen (2016) esittämä katsaus valmisohjelmistoihin täydentää kuvaa hankinnan näkökulmasta. Kirjallisuuden perusteella modulaarinen monoliitti näyttäytyy usein PK-yritykselle järkevänä lähtökohtana, joka pitää arkkitehtuurin hallinnan ja infrastruktuurin yksinkertaisempana. Mikropalvelut tuovat lisäarvoa tilanteissa, joissa skaalautuvuus, vikaeristys, teknologinen joustavuus sekä vaiheittainen legacy-järjestelmän uudistaminen nousevat tärkeiksi. Näiden pohjalta myöhemmin työssä voidaan tarkastella, missä tilanteissa monoliittinen ratkaisu riittää, milloin mikropalveluihin siirtyminen on perusteltua ja milloin ulkoisen työkalun hyödyntäminen on realistinen vaihtoehto.

2.6 B2B-integraatiokaavat

B2B-integraatioilla tarkoitetaan tässä työssä toimittajan ja ostajan välistä tiedonsiirtoa, jossa välitetään kaksisuuntaista tuote- ja tilaustietoa. Käytännössä integraatioissa voidaan erottaa kaksi toisiaan täydentävää virtaa: push-virta ja pull-virta. Push-virralla tarkoitetaan massadatan siirtoa, jossa toimittaja välittää suuremman tietomäärän vastaanottajalle sovitussa tahdissa, tyypillisesti eräajona. Tämä massadata kattaa esimerkiksi tuoteluettelot, hinnastot ja varastosaldot ja se voidaan välittää esimerkiksi XML- tai taulukkomuodossa.

Tätä tarkoitusta varten käytetään GS1 EAN Communication -standardin (EANCOM) hinnastosanomaa (Price/Sales Catalogue, PRICAT) ja varastosaldoraporttia (Inventory Report, INVRPT), jotka määrittelevät katalogi- ja saldotietojen rakenteen ja kentät yhdenmukaisesti (GS1, 2016/2021a; GS1, 2016/2021b). Rengasalalla tätä perusrakennetta sovelletaan EDIWheel-viitekehyksessä, joka määrittelee rengasvalmistajien ja jakelijoiden välisen sähköisen tiedonsiirron, ja jota EDIWheelin dokumentaation mukaan tukevat moni rengasvalmistaja ja rengaskaupan toimija (EDIWheel, n.d.). Käytännössä EDIWheel nojaa GS1:n tunnisteisiin ja sanomarakenteisiin, kuten PRICAT- ja INVRPT-

sanomiin, ja määrittelee niistä rengasalan erityistarpeisiin soveltuvan profiilin. Toinen virta on transaktio-API eli vastaanottajan tekemä pull-kutsu, joka sopii yksittäisiin kyselyihin ja tapahtumiin, kuten saatavuus- ja hintakyselyihin, tilauksen tekemiseen ja tilausstatuksen hakuun. REST-periaatteet (Representational State Transfer) korostavat yhtenäistä rajapintaa ja tilattomuutta, mikä auttaa pitämään API-operaatiot ennakoitavina ja laajennettavina (Fielding, 2000). Käytännössä transaktio-API täydentää eräsiirtoa siten, että massadata päivitetään säännöllisesti ja yksittäiset tapahtumat toteutetaan erillisinä API-kutsuina.

Yleisesti ottaen integraatiotavat kannattaa yhdistää luotettavalla tavalla. Eräsiirroissa keskeinen periaate on kertakäsittely (idempotenssi), jotta sama erätiedosto tai tapahtuma ei vahingossa päivitä tietoja useaan kertaan. Tämän toteuttamiseen käytetään usein idempotenttia vastaanotinta (Idempotent Receiver) ja korrelaatiotunnistetta (Correlation Identifier), joiden avulla sama erätiedosto tai tapahtuma voidaan tunnistaa ja varmistaa, että se ajetaan ainoastaan kerran. Lisäksi tapahtumaketju pystytään jäljittämään tarpeen mukaan (Hohpe & Woolf, 2004). EANCOM-sanomien käyttö tukeutuu lisäksi GS1:n Syntax 4 -määrittelyyn, joka kuvaa sanomien syntaksin ja käyttöympäristön (GS1, 2022).

Push-tiedostot ja pull-API eivät ole ainoa tapa toteuttaa tämänkaltaisia integraatioita, mutta ne muodostavat käytännössä käyttökelpoisen toimintatavan. Hohpe ja Woolf (2004) kuvaavat tapahtumapohjaisia viestikanavia, joissa järjestelmät eivät kutsu toisiaan suoraan, vaan lähettävät sanomia, jotka tähän tarkoitettu palvelu jakaa eteenpäin oikeille vastaanottajille. Esimerkiksi julkaisu–tilaus-tyyppinen kanava mahdollistaa sen, että yksi tapahtuma voidaan toimittaa useille vastaanottajille ilman, että julkaisijan tarvitsee erikseen tietää kohdejärjestelmät (Hohpe & Woolf, 2004). Eugster et al. (2003) korostavat samaa periaatetta publish/subscribe-paradigmassa, jossa tilaajat rekisteröivät kiinnostuksensa tiettyihin tapahtumiin ja saavat ilmoituksia, kun kanavassa julkaistaan niitä vastaavia tapahtumia.

Käytännössä samaa ajattelutapaa voidaan soveltaa eri teknologioilla, kuten viestijonoilla ja sanomavälityspalveluilla, tai HTTP-pohjaisilla webhook-ilmoituksilla, joissa toimittaja kutsuu vastaanottajan tarjoamaa ilmoitusrajapintaa aina, kun tietty tapahtuma toteutuu. Monitoimittajaympäristössä vastaanottavan järjestelmän kannattaa joka tapauksessa varautua useampaan integraatiomalliin, vaikka rengasalalla EDIWheel-viitekehys yhtenäistääkin sanomarakenteita suurten toimittajien osalta (EDIWheel, n.d.). Näin arkkitehtuuri pysyy toimittajariippumattomana ja tukee erilaisia integraatioprofiileja, kunhan da-

tamallit ja tunnisteet pidetään yhtenäisinä eri kanavien välillä (PRICAT/INVRPT) ja sanomien syntaksi GS1 EANCOM Syntax 4 -määrittelyn mukaisena (GS1, 2016/2021a; GS1, 2016/2021b; GS1, 2022).

Yhteenvetona tämä alaluku kokoaa eri integraatioiden peruskaavan, jossa eräsiirrot hoitavat massadatan päivitykset ja API-pyynnöt tukevat toimintaa yksittäisinä tapahtumina. Tämä jaottelu tukee työn kokonaisuutta tarjoamalla rakenteen, jonka päälle myöhemmissä luvuissa tarkasteltavat ratkaisut voidaan sovittaa. Samalla se auttaa jäsentämään, mitä tietoa siirretään eräajona ja mitä kysytään API:n tai webhookin kautta. Näin integraatioarkkitehtuuri voidaan rakentaa siten, että se ottaa huomioon eri integraatiotavat, joita toimittajat voivat mahdollisesti käyttää. Rengasalalla tämä kehys konkretisoituu EDIWheel-viitekehyksen kautta, jota hyödynnetään myöhemmin empiirisissä havainnoissa ja ratkaisun suunnittelussa (EDIWheel, n.d.). Lisäksi se tekee näkyväksi, millaiset GS1-standardeihin, idempotenssiin ja sanomapohjaisiin integraatiomalleihin liittyvät perusvaatimukset PK-yrityksen on käytännössä otettava huomioon.

3. TUTKIMUSMENETELMÄT

3.1 Tutkimusasetelma ja lähestymistapa

Tämä työ toteutettiin konstruktiivisena tapaustutkimuksena, jossa kehitetään Rengas Turku Oy:n ostoprosessia tukeva ratkaisu ja arkkitehtuuri. Konstruktiivinen tutkimustapa sopii tilanteisiin, joissa tavoitteena on käytännön ongelman ratkaiseminen, ratkaisun teoreettinen perustelu ja sen toimivuuden arviointi todellisessa toimintaympäristössä (Lukka, 2003). Tässä työssä konstruktio tarkoittaa ostoprosessia tukevaa arkkitehtuurista ratkaisua ja siihen liittyviä periaatteita, ei valmista tuotantovalmista järjestelmää.

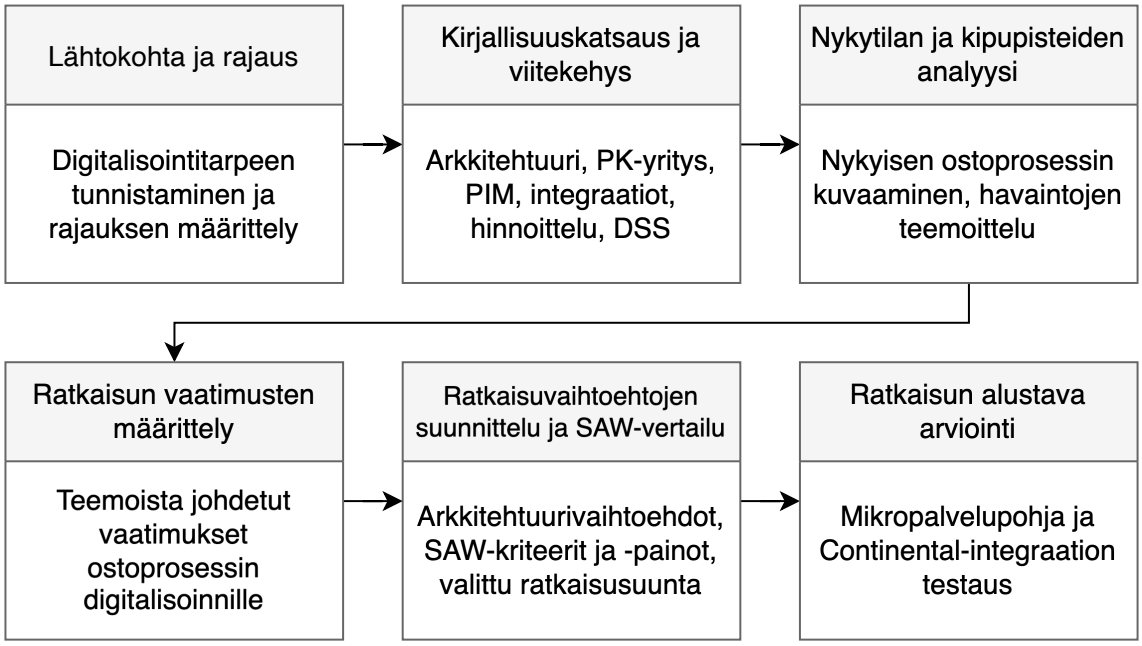
Tutkimus on samalla myös tapaustutkimus, koska tässä tarkastellaan yhden organisaation ostoprosessia ja sitä tukevaa Django-pohjaista järjestelmää omassa ympäristössään. Tapaustutkimus sopii erityisesti tilanteisiin, joissa ilmiötä ja sen kontekstia ei voi erottaa toisistaan, vaan tutkimuksen on otettava huomioon koko toimintaympäristö (Díaz Andrade, 2009). Aineistoa käytettiin ratkaisun suunnittelun tukena, eikä tavoitteena ollut tehdä tilastollisia yleistyksiä, vaan tarkastella ilmiötä tapauskohtaisesti toimintaympäristöön sidottuna.

Tutkija on käyttänyt Rengas Turku Oy:n järjestelmää ensimmäisen kerran jo vuonna 2016 ja tehnyt myyntityötä kivijalkaliikkeessä harvakseltaan vuosina 2020–2025. Tämä aikaisempi käytännön kokemus muodosti esiymmärryksen siitä, miten järjestelmä ja ostoprosessi näkyvät arjessa. Varsinainen aineistonkeruu ja konstruktion kehittäminen rajattiin kuitenkin vuoden 2025 aikana tehtyihin toimiin, joita kuvataan seuraavassa osiossa.

Konstruktiivisessa tutkimuksessa keskeistä on ratkaisun kehittäminen tiiviissä yhteistyössä käytännön toimijoiden kanssa sekä ratkaisun toimivuuden arviointi käytännön kokeiluilla (Lukka, 2003). Tässä työssä yhteistyö toteutui ostopäällikön, ohjelmistovastaavan, pääkehittäjän ja yhden rengastoimittajan yhteyshenkilön kanssa. Tutkija arvioi ratkaisun toteutettavuutta lisäksi rajatulla teknisellä kokeilulla Continentalin rajapintaa vasten.

Tutkimuksen kokonaiskulku on koottu kuvaan 1. Kuvan tarkoitus on näyttää työn pääasiallinen eteneminen, mutta käytännössä työ ei edennyt suorana putkena, vaan vaiheiden välillä palattiin myös taaksepäin. Nykytilan ja kipupisteiden kartoitus ohjasi sitä, mihin teoriaan perehdyttiin seuraavaksi, ja alustavat vaatimukset tarkensivat edelleen, mitä kirjallisuutta oli järkevää lukea. Kirjallisuus ei siis ollut vain työn alkuvaihe, vaan sitä täy-

dennettiin työn aikana aina kun aineistosta nousi uusia kysymyksiä. Kuvassa 1 näkyvässä ratkaisuvaihtoehtojen vertailussa hyödynnettiin painotettua summamallia (Simple Additive Weighting, SAW), eli yksinkertaista monikriteeristä menetelmää. Menetelmä ja sen käyttö tässä työssä kuvataan tarkemmin luvussa 3.4.



Kuva 1: *Tutkimuksen kokonaiskulku.*

3.2 Tutkimuskohde ja aineistonkeruu

Tutkimuskohteena on Rengas Turku Oy:n ostoprosessi ja sitä tukeva Django-pohjainen järjestelmä. Tutkija teki aineistonkeruun ja konstruktion kehitystyön pääosin vuoden 2025 aikana. Aineisto kerättiin neljällä tavalla:

1. osallistuvalla havainnoinnilla ostopäällikön työssä,
2. palavereilla ja epämuodollisilla keskusteluilla yrityksen avainhenkilöiden kanssa,
3. teknisellä dokumentaatiolla ja toimittajien aineistolla sekä
4. järjestelmän koodin tarkastelulla.

Seuraavissa alaluvuissa kuvataan kunkin tavan käytännön toteutus, osapuolet ja tutkijan rooli.

Osallistuva havainnointi ostopäällikön työssä

Tammi–helmikuussa 2025 tutkija toteutti kolme havainnointikertaa ostopäällikön kanssa ja seurasi ostopäällikön työtä käytännön arjessa. Havainnointi oli keskustelevaa, ja jokainen kerta täydensi edellistä. Ostopäällikkö esitteli, miten järjestelmää käytetään, ja

kuvasi samalla, miten ostoprosessi etenee eri tilanteissa, mitä tietoja eri vaiheissa hyödynnetään ja milloin hän tekee keskeiset päätökset. Havainnointikertoja oli kolme, koska ostoprosessin vaihe vaihteli eri kerroilla. Kolmannen kerran jälkeen tutkija ja ostopäällikkö arvioivat, että prosessista oli muodostunut riittävän laaja kokonaiskuva. Tutkija vastasi muistiinpanoista ja kokosi havainnot Obsidian-muistiinpanoihin. Muistiinpanoihin kirjattiin prosessin vaiheet, käytetyt työkalut ja tietolähteet sekä se, millaista työtä ostopäällikkö tekee eri tilanteissa. Erillistä kenttäpäiväkirjaa ei laadittu.

Palaverit ja epämuodolliset keskustelut

Tutkimuksen aikana tutkija järjesti useampia yrityksen sisäisiä palavereita, joissa täsmennettiin ostoprosessin nykytilaa, kehitystarpeita ja ratkaisuvaihtoehtoja. Palavereihin osallistui tilanteen mukaan ostopäällikkö, ohjelmistovastaava ja pääkehittäjä. Tutkija kokosi palavereista muistiinpanot ja hyödynsi niitä sekä prosessikuvauksen tarkentamisessa että ratkaisuvaihtoehtojen vertailun valmistelussa. Menetelmän kannalta keskeiset palaverit olivat:

1. aloituspalaveri, jossa käytiin läpi ostoprosessin perusrunko, tarpeet ja keskeiset haasteet,
2. tutkijan ja ostopäällikön palaveri, jossa ostopäällikkö kommentoi havainnoinnin pohjalta laadittua prosessikuvausta ja sen vastaavuutta arjen ostotyöhön,
3. tutkijan ja pääkehittäjän palaveri, jossa pääkehittäjä esitteli järjestelmää kooditasolla ja jossa käsiteltiin toteutusvaihtoehtoja sekä teknisiä reunaehtoja,
4. palaveri, jossa tutkija esitteli arkkitehtuurivaihtoehdot ja jossa vaihtoehtoja tarkasteltiin yhdessä; keskustelua hyödynnettiin SAW-vertailun kriteerien ja painojen valmistelussa, sekä
5. palaveri, jossa käytiin läpi SAW-pisteytyksen tulokset ja arkkitehtuurisuuntien vahvuuksia ja heikkouksia arvioitiin; palaveri tuki ratkaisusuunnan valinnan valmistelua.

Listan kahdesta viimeisestä palaverista on kooste liitteessä (ks. Liite 1, ”Yrityspalaverit”), koska niissä valmisteltiin vertailussa käytettäviä kriteerejä ja painoja ja tuettiin työn kannalta keskeistä ratkaisusuunnan valintaa. Varsinainen vaihtoehtojen vertailu ja valinnan perustelut esitetään luvussa 5.

Tekninen dokumentaatio ja toimittajien aineisto

Tutkija täydensi aineistoa järjestelmän tietomallia ja arkkitehtuuria kuvaavilla dokumenteilla sekä toimittajien rajapintakuvauksilla. Valmiita teknisiä dokumentteja oli rajallisesti,

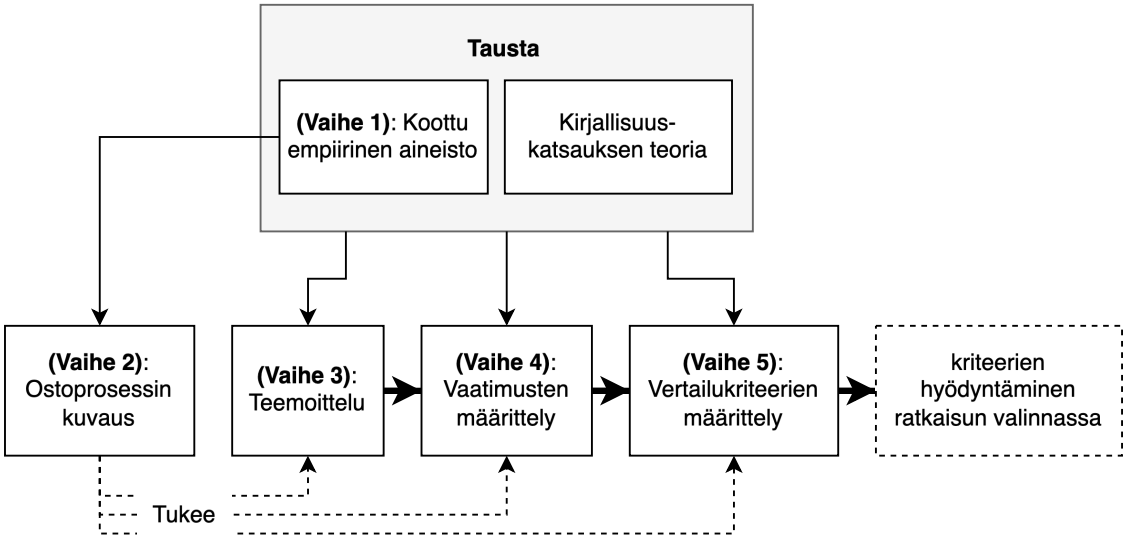
joten tutkija täydensi niitä pääkehittäjän tuella. Continental toimitti tutkijalle rajapintakuvauksia, esimerkkiaineistoja ja erätiedostoesimerkkejä sähköpostitse. Syyskuussa 2025 järjestettiin erillinen palaveri Continentalin asiantuntijan kanssa, jossa käytiin läpi integraatiomahdollisuuksia ja käytännön toteutusta Rengas Turku Oy:n järjestelmäympäristössä.

Ohjelmiston kooditason tarkastelu

Huhtikuusta 2025 alkaen tutkija perehtyi järjestelmään kooditasolla. Tarkastelu kohdistui ostoprosessin kannalta keskeisiin toimintoihin, tietorakenteisiin, integraatiopisteisiin ja järjestelmän modulaariseen rakenteeseen (Django-sovellukset). Tutkija kävi koodia läpi pääkehittäjän kanssa ja esitti epäselvissä kohdissa tarkentavia kysymyksiä. Tarkastelun tarkoituksena oli varmistaa, että myöhemmin kehitettävät ratkaisuvaihtoehdot ovat teknisesti realistisia nykyisessä ympäristössä ja yhteensopivia olemassa olevan arkkitehtuurin kanssa.

3.3 Aineiston analyysi

Tässä luvussa aineistolla tarkoitetaan kerättyä empiiristä aineistoa, eli muistiinpanoja ja materiaaleja havainnoinneista, palavereista, dokumenteista ja koodin tarkastelusta. Kuvassa 2 kirjallisuuskatsauksen teoriapohja ja työn aineisto kulkevat mukana analyysin taustana koko ketjun ajan. Tarkoituksena ei ollut tehdä tilastollisia yleistyksiä, vaan käyttää aineistoa konstruktion suunnittelun tukena, mikä on tyypillistä konstruktiiviselle tapaustutkimukselle ja tapaustutkimukselle yleisemmin (Lukka, 2003; Díaz Andrade, 2009).



Kuva 2: *Aineiston analyysin kulku.*

Seuraavaksi kuvataan vaihe vaiheelta tarkemmin, miten empiirinen aineisto käsiteltiin analyysissä. Vaiheiden eteneminen on koottu kuvaan 2.

Vaiheessa 1 (kuva 2) tutkija kokosi havainnoinnin, palaverien ja epämuodollisten keskustelujen muistiinpanot samaan Obsidian-vaultiin. Samalle alustalle koottiin myös keskeiset huomiot teknisestä dokumentaatiosta ja koodin tarkastelusta. Tarkoitus oli, että kaikki empiirinen aineisto löytyy yhdestä paikasta ja siihen on helppo palata myöhemmissä vaiheissa.

Vaiheessa 2 (kuva 2) tutkija muodosti empiirisen aineiston pohjalta ostoprosessin kuvauksen. Kuvan 2 mukaisesti kuvausta käytettiin tukena analyysin muissa vaiheissa. Prosessin kuvaus tehtiin siksi, että havainnot ja myöhemmät tulkinnat saatiin liitettyä oikeisiin ostoprosessin vaiheisiin. Ostoprosessin kuvaus esitetään luvussa 4.2.

Vaiheessa 3 (kuva 2) tutkija ryhmitteli empiirisen aineiston havainnot teemoiksi ja kokosi samalla kipupisteet. Käytännössä saman tyyppiset huomiot koottiin yhteen ja niille annettiin kuvaava teemanimi. Tämän jälkeen teemoista erotettiin ne kohdat, jotka kuvasivat selviä ongelmia tai kehitystarpeita. Kuvan 2 lineaarisesta esityksestä poiketen, teemoitusta tarkennettiin työn aikana useaan kertaan, ja tarvittaessa teemoja yhdistettiin, jaettiin tai täsmennettiin. Teemat ja kipupisteet raportoidaan luvussa 4.3.

Vaiheessa 4 (kuva 2) tutkija johti teemoista ja kipupisteistä ratkaisun vaatimukset ja kokosi ne vaatimuslistaan. Kuvan 2 mukaisesti vaatimuksia tarkennettiin palaamalla tarvittaessa takaisin empiiriseen aineistoon, jotta yhteys havaintoihin säilyi. Kirjallisuuskatsauksen teoriapohjaa käytettiin tässä vaiheessa apuna, jotta vaatimukset saatiin kirjoitettua selkeiksi, jäsennellyiksi ja vertailukelpoisiksi. Vaatimuslistaa tarkennettiin palaverissa ostopäällikön ja pääkehittäjän kanssa. Vaatimukset esitetään luvussa 4.5.

Vaiheessa 5 (kuva 2) tutkija muodosti vaatimuslistan pohjalta ratkaisuvaihtoehtojen vertailukriteerit. Kuvan 2 mukaisesti kriteerien tarkoitus oli kuvata, miten hyvin eri vaihtoehdot täyttävät vaatimukset. Vertailun valmistelu kuvataan luvussa 3.4, ja vertailukriteerit sekä niiden painot esitetään luvussa 5.2, jossa niitä hyödynnetään SAW-menetelmässä.

Kuvan 2 tarkoitus on sitoa yhteen se, miten empiirisestä aineistosta edettiin prosessikuvaukseen, kipupisteisiin, vaatimuksiin ja lopulta vertailukriteereihin. Kuvan avulla lukija näkee tämän ketjun, vaikka luvun 4 ja luvun 5 varsinaiset tulokset esitetään vasta myöhemmin.

3.4 Ratkaisuvaihtoehtojen kartoitus ja SAW-vertailun valmistelu

Ratkaisuvaihtoehtojen kartoitus ja SAW-vertailun valmistelu perustuivat luvussa 4.5 esitettyihin vaatimuksiin, luvussa 2.5 kuvattuun arkkitehtuuriteoriaan sekä yrityksen nykyiseen järjestelmäympäristöön, resursseihin ja toimintatapoihin. Lisäksi taustana hyödynnettiin kirjallisuuskatsauksen muuta teoriaa sekä yrityksen kanssa käytyjä keskusteluja. Työ eteni neljässä vaiheessa:

1. ratkaisuvaihtoehtojen muodostaminen,
2. vaihtoehtojen läpikäynti ja esikarsinta yrityksen kanssa,
3. vertailukriteerien ja painojen määrittely sekä
4. vaihtoehtojen pisteytys SAW-menetelmällä.

Vaiheessa 1 tutkija muodosti neljä ratkaisuvaihtoehtoa (A–D), jotka kuvasivat eri etenemistapoja: nykyisen Django-pohjaisen järjestelmän laajentaminen (A), erillinen mikropalvelu (B), ulkoisen valmiin ratkaisun integrointi (C) sekä prosessin kehittäminen ilman uutta järjestelmää (D). Tutkija laati vaihtoehdoista lyhyet kuvaukset vaatimusten ja kirjallisuuden pohjalta ja valmisti niitä varten esitykset. Lisäksi tutkija piirsi vaihtoehdoille A ja B arkkitehtuurikaaviot, joissa kuvattiin karkealla tasolla ratkaisun kytkentä nykyjärjestelmään ja tiedonkulku.

Vaiheessa 2 tutkija järjesti arkkitehtuuripalaverin ja kävi vaihtoehdot läpi yhdessä yrityksen ammattilaisten kanssa (ks. Liite 1, “Yrityspalaverit”). Palaverissa tarkasteltiin, miten kukin vaihtoehto vastaa vaatimuksiin ja miten se sopii yrityksen strategiaan ja resursseihin. Keskustelun perusteella tehtiin esikarsinta. Kaikkia vaihtoehtoja ei viety SAW-pisteytykseen, vaan osa rajautui pois esikarsinnassa tai jäi vertailupohjaksi. Varsinainen vaihtoehtojen vertailu ja valinnan perustelut esitetään luvussa 5.

Vaiheessa 3 vaiheessa tutkija määritteli SAW-vertailussa käytettävät kriteerit ja niiden painot. Lähtökohtina olivat vaatimuslista (luku 4.5), yrityksen kanssa käydyt keskustelut sekä kirjallisuudessa esitetyt monikriteerisen arvioinnin periaatteet. Tutkija valmisteli kriteerilistan ja painotuksen, ja yrityspalaveri vaikutti siihen, mitkä kriteerit korostuivat ja miten painot lopulta asetettiin. Kriteerit ja painot esitetään luvussa 5.2.

Vaiheessa 4 vaiheessa tutkija pisteytti jatkoon valitut vaihtoehdot kriteeri kerrallaan ja toteutti SAW-laskennan taulukkolaskentaohjelmassa. Laskennassa kriteeripisteet normalisoitiin ja painotettiin painotetun summamallin periaatteiden mukaisesti (Tzeng & Huang, 2011). Lisäksi toteutettiin herkkyysanalyysi painotusten vaikutuksen arvioimiseksi. Pisteytyksen toteutus, herkkyysanalyysi ja tulokset esitetään luvussa 5.3.

3.5 Ratkaisun tekninen kokeilu ja alustava arviointi

Konstruktiivisessa tutkimuksessa ratkaisun toteutettavuus testataan kevyesti käytännön ympäristössä, jotta saadaan alustavaa näyttöä sen toimivuudesta (Lukka, 2003). Tässä työssä tutkija teki rajatun teknisen kokeilun kehitysympäristössä yhden rengastoimittajan testiaineistolla. Kokeilun tarkoitus oli varmistaa, että toimittajaintegraatio toimii käytännössä, kaksivirtaisen mallin ajatus on toteutettavissa ja tuotteet ovat yhdistettävissä tunnisteiden avulla siten, että kanoninen tuotemalli on rakennettavissa.

Kokeilu eteni neljässä vaiheessa:

1. kevyen kokeilupohjan valmistelu kehitysympäristöön,
2. toimittajan pull-rajapinnan testaaminen rajatuilla kyselyillä,
3. toimittajan push-aineiston tarkastelu ja kevyt tietorakenne sekä
4. push- ja pull-virtojen yhdistettävyyden kokeilu tuotetunnisteiden avulla.

Vaiheessa 1 tutkija rakensi kehitysympäristöön kokeilua varten kevyen pohjan ja rajasi toteutuksen vain välttämättömiin osiin. Toimittaja toimitti testikäyttöön rajapintatunnukset sekä esimerkkiaineiston eräsiirron testausta varten.

Vaiheessa 2 tutkija testasi pull-tyyppistä rajapintaa rajatusti saatavuus- ja hintakyselyillä muutamalle yksittäiselle tuotteelle. Kokeilussa ei tehty ostotilauksia eikä päivitetty tietoja varsinaiseen järjestelmään, koska tarkoitus oli arvioida toteutettavuutta eikä toteuttaa tuotantotasoista integraatiota.

Vaiheessa 3 tutkija tarkasteli push-tyyppistä eräsiirtoaineistoa ja vertasi esimerkkierätiedoston kenttiä nykyjärjestelmässä käytettyihin tietoihin. Lisäksi tutkija toteutti kokeilua varten kevyen parserin, joka luki erätiedoston ja muodosti sen pohjalta yksinkertaisen tietorakenteen muutamalle tuotteelle. Tarkoitus oli testata käytännössä, millaiset kentät ja tunnisteet ovat saatavilla ja onko niiden pohjalta mahdollista rakentaa kanonisen tuotemallin kaltainen sisäinen tuoterakenne, ei rakentaa valmista integraatiota.

Vaiheessa 4 tutkija kokeili, ovatko push- ja pull-virrat yhdistettävissä käytännössä. Tämä tehtiin valitsemalla eräaineistosta muutama tuote ja tekemällä samoille tuotteille rajapintakyselyt eräaineistossa olevia tunnisteita käyttäen, jotta voitiin arvioida yhdistämisen perusedellytykset.

Kokeilu rajattiin yhden toimittajan testiaineistoon, yhteen eräaineistoon ja muutamaan kyselyyn, eikä sillä arvioitu suorituskykyä eikä poikkeustilanteita. Tulokset raportoidaan luvussa 4.4, ja niitä hyödynnetään ratkaisuvaihtoehtojen arviointiin luvussa 5.3.

3.6 Yhteenveto

Tässä luvussa kuvattiin, miten tutkimus toteutettiin konstruktiivisena tapaustutkimuksena ja miten valitut menetelmät tukivat ostoprosessia tukevan ratkaisun suunnittelua ja sen toteutettavuuden tarkistamista.

Aineistonkeruussa tutkija yhdisti käytännön työstä kertyneen empiirisen aineiston ja järjestelmän teknisen tarkastelun. Keskeiset tiedot koottiin havainnoimalla ostopäällikön työtä, käymällä läpi prosessia ja ratkaisusuuntaa yrityksen ammattilaisten kanssa sekä täydentämällä ymmärrystä toimittaja-aineistolla ja kooditason tarkastelulla. Näin aineisto kattoi sekä ostotyön arjen että nykyjärjestelmän tekniset rajoitteet ja mahdollisuudet.

Aineistoanalyysissa tutkija jäsensi havainnot ostoprosessin vaiheisiin, teemoitti ja muunsi ne vaatimuksiksi, joita käytettiin vertailun pohjana. Tulokset raportoidaan luvussa 4 (nykytila ja vaatimukset) ja luvussa 5 (vertailu ja valinta).

Ratkaisuvaihtoehtojen kartoitus ja SAW-vertailun valmistelu tehtiin vaatimusten, työn teoriapohjan sekä yrityksen arjen tarpeiden ja resurssien perusteella. Tutkija muodosti vaihtoehdot ja valmisteli vertailun kriteerit ja painot, ja niitä täsmennettiin yrityksen kanssa käytyjen keskustelujen pohjalta. Varsinainen vertailu ja valinnan perustelut esitetään luvussa 5.

Lisäksi tutkija teki rajatun teknisen kokeilun kehitysympäristössä. Kokeilulla varmistettiin, että suunniteltu integraatiotapa on teknisesti mahdollinen tässä ympäristössä ja että suunnitellulle tuotteiden yhdistämiselle on tarvittavat edellytykset. Kokeilun tarkempi kuvaus ja sitä koskevat tulokset raportoidaan myöhemmissä tulosluvuissa.

Menetelmään liittyvät rajoitteet johtuvat tapaustutkimuksen luonteesta: tutkimus on sidottu tiettyyn aikaan, paikkaan ja kontekstiin, joten tuloksia ei pyritä yleistämään tilastollisesti, vaan niiden arvo on tapauskohtaisessa ja teoreettisessa ymmärryksessä (Díaz Andrade, 2009). Työ kohdistuu yhteen organisaatioon ja yhteen järjestelmäympäristöön, ja tekninen kokeilu rajattiin yhteen toimittajaan sekä rajattuihin testitapauksiin. Tutkijan aiempi kokemus yrityksen järjestelmästä tuki työn käytännöllistä etenemistä, ja tulkintojen tasapainottamiseksi ratkaisuja ja vaatimuksia tarkennettiin yrityksen avainhenkilöiden kanssa.

Näistä lähtökohdista siirrytään luvussa 4 kuvaamaan ostoprosessin nykytilaa ja kipupisteitä, jotka toimivat vaatimusten ja myöhemmin valitun ratkaisun taustana.

4. EMPIIRISET HAVAINNOT JA RATKAISUN VAATIMUKSET

4.1 Empiirisen tarkastelun tavoite ja rajaus

Tässä luvussa kuvataan Rengas Turku Oy:n ostoprosessin nykytila: mitä prosessissa käytännössä tehdään, missä järjestyksessä työ etenee ja mitkä ovat prosessin keskeiset kipupisteet. Luvun tarkoitus on antaa lukijalle selkeä ja yhteinen lähtökuva ennen kuin työssä siirrytään ratkaisuvaihtoehtojen vertailuun myöhemmin luvussa 5. Tässä luvussa ei vielä ehdoteta ratkaisuja eikä arvioida vaihtoehtoja, vaan keskitytään nykytilan kuvaamiseen ja siihen liittyviin ongelmakohtiin.

Työssä edetään järjestyksessä **nykytila** → **kipupisteet** → **vaatimukset** → **ratkaisun suunnittelu ja arviointi**, jotta myöhemmät ratkaisuvalinnat voidaan perustella suoraan tässä luvussa kuvattuihin havaintoihin. Ratkaisun rakentamisen ja arvioinnin vaiheistus on linjassa konstruktiivisen tutkimusotteen kanssa (Lukka, 2003). Tulkinnallisessa tapaustutkimuksessa empiirinen kuvaus toimii myös myöhemmän teoreettisen tulkinnan pohjana, jolloin havaintojen selkeä ja läpinäkyvä esittäminen on olennaista (Díaz Andrade, 2009).

Luku etenee seuraavasti:

4.2 kuvaa ostoprosessin etenemisen ja keskeiset työvaiheet nykyisessä toimintaympäristössä.

4.3 kokoaa havainnot teemoihin ja tekee kipupiste et näkyviksi konkreettisten esimerkkien avulla.

4.4 tarkentaa toimittajaintegraatioihin liittyviä havaintoja Continentalin esimerkin sekä EDIWheel-havaintojen kautta.

4.5 yhdistää kipupisteet luvussa 2 esitettyyn viitekehykseen ja johtaa niiden pohjalta ratkaisun vaatimukset.

4.6 tiivistää luvun keskeiset havainnot ja vaatimukset ja kytkee ne seuraavan luvun ratkaisuvaihtoehtojen vertailun lähtökohdiksi.

4.2 Ostoprosessin kuvaus

Rengas Turku Oy:ssä ostaminen on jokapäiväistä työtä. Ostoprosessi jakautuu käytännössä kolmeen työputkeen.

1. **Massatilaukset ja sesonkivalmistelu**, joissa suunnitellaan rengasvalikoimaa sesonkia varten ja tehdään isommat varastontäydennykset kerralla toimittajakohtaisesti.
2. **Päivittäinen varastontäydennys**, jossa seurataan varastotasoja ja paikataan puutteita tarpeen mukaan.
3. **Myymälöiden asiakastilaukset**, joissa asiakaspalvelija tilaa tuotteen yksittäiselle asiakkaalle, jos sitä ei löydy varastosta.

Tässä työssä keskitytään kahteen ensimmäiseen työputkeen, koska ne vievät suurimman osan ajasta ja liittyvät suoraan tarkasteltavaan ratkaisuun. Renkaiden ostotyötä tekee tällä hetkellä ostopäällikkö.

Massatilaukset ja sesonkivalmistelu

Massatilaukset käynnistyvät toimittajilta saapuvilla toimittajataulukoilla (Excel). Jokaisella isolla toimittajalla on oma tapa rakentaa taulukot, ja ne voivat vaihdella myös saman toimittajan erien välillä. Sarakkeet, otsikot ja koodistot vaihtelevat, ja taulukoissa voi olla satoja tai jopa tuhansia rivejä. Samassa taulukossa on yrityksen kannalta kiinnostavia tuotteita, mutta suuri osa riveistä ei tule koskaan käyttöön.

Työ etenee käytännössä seuraavasti:

1. **Laskentakopion muodostaminen:** Ostopäällikkö tekee toimittajataulukosta oman kopion. Kopiota käytetään erityisesti ostohinnan laskentaan ja tilauksen kokonaisuuden hahmottamiseen.
2. **Osto- ja myyntihinnan laskenta:** Ostopäällikkö rikastaa toimittajan hintaa vakiintuneen laskentatavan mukaan huomioimalla esimerkiksi kampanjat, alennukset, volyymibonukset, kierrätysmaksut ja muut kulut. Yhtä kaikkiin toimittajiin sopivaa laskentamallia ei ole, vaan kaavat ja sarakkeet riippuvat toimittajasta ja sopimuksesta. Samassa vaiheessa ostopäällikkö määrittää kiinnostaville tuotteille myös alustavan myyntihinnan.
3. **Tuotevalinnat:** Ostopäällikkö tekee tuotevalinnat. Tuotteita verrataan aiempaan myyntiin, kysyntään, toimittajakohtaisiin linjauksiin ja ostopäällikön omaan kokemukseen. Kun tuote päätetään ottaa mukaan, se merkitään alkuperäiseen toimittajataulukkoon tilattavaksi. Tämä merkintä tarkoittaa käytännössä, että tuote tilataan.
4. **Vertailu järjestelmätietoihin:** Valintoja verrataan yrityksen järjestelmässä olevaan tietoon. Käytännössä ruudulla on auki toimittajataulukko, laskentakopio ja

yrityksen järjestelmä (Nettivanne), ja rivejä verrataan nykyisiin hintoihin, varastosaldoihin ja vastaaviin tuotteisiin.

5. **Tilauksen lähetys ja kirjaaminen:** Päätösten jälkeen alkuperäinen taulukko tuotevalintoineen toimitetaan sähköpostilla toimittajalle. Samalla ostopäällikkö vie ostotiedot manuaalisesti yrityksen järjestelmään: aiemmin myynnissä olleiden tuotteiden hinnat päivitetään ja uusille tuotteille luodaan tarvittavat tuotekortit ja tunnisteet. Lisäksi ostopäällikkö kirjaa tuotteiden myyntihinnat järjestelmään ja tarkentaa tarvittaessa efektiivisen ostohinnan perusteella automaattisesti laskettuja hintoja, jos lopputulos ei vastaa tavoiteltua hinnoittelua.

Tuotteiden yhdistämisessä ensisijainen tunniste on GTIN, jota täydennetään toimittajakohtaisilla koodeilla ja järjestelmän omilla tunnisteilla. Lopuksi järjestelmään muodostetaan toimittajakohtaiset ostotilaukset, joihin liitetään toimitusvarasto ja arvioitu saapumisaika. Tuotekuvia ja muuta täydentävää sisältöä voidaan lisätä myös myöhemmin muun henkilöstön toimesta.

Massatilauksissa tuotevalinnat lukittuvat käytännössä siinä vaiheessa, kun toimittajalle lähetettävä taulukko on toimitettu. Tämän jälkeen työ painottuu tilausten kirjaamiseen sekä hintojen ja perustietojen päivittämiseen yrityksen järjestelmässä. Käytännössä vaiheet eivät kuitenkaan aina etene täysin peräkkäin, vaan ostopäällikkö voi tehdä valintoja, kirjauksia ja päivityksiä osin rinnakkain jo ennen kuin lopullinen taulukko on lähetetty.

Päivittäinen osto ja varastontäydennys

Päivittäinen osto ja varastontäydennys ovat kevyempi, mutta jokapäiväinen osa samaa kokonaisuutta. Tavoitteena on varmistaa, että tärkeät tuotteet eivät pääse loppumaan ja että yksittäiset puutteet korjataan ajoissa. Tämä työputki poikkeaa taulukkomuotoisesta massatilauksesta erityisesti siinä, että tilaus tehdään tyypillisesti toimittajan verkkokaupan kautta yritykselle annetuilla tunnuksilla.

Tyypillinen eteneminen on seuraava:

1. **Tilanteen tarkistus:** Ostopäällikkö käy läpi myytyjä tuotteita ja varastotilannetta yrityksen järjestelmästä. Nettivanteessa on näkymä, josta näkee päivän aikana myydyt tuotteet. Näkymä ei ole varsinaisesti suunniteltu ostotyökaluksi, mutta se on vakiintunut tähän käyttöön.

2. **Täydennystarpeen arviointi**: Ostopäällikkö tarkistaa kunkin tuotteen saldon, menekin ja avoimet tilaukset ja arvioi, pitäisikö tuotetta olla jatkuvasti saatavilla, kuinka nopeasti sitä tarvitaan ja onko sitä jo tulossa.

3. **Tilauksen kokoaminen:** Ostopäällikkö pyrkii ajoittamaan ja kokoamaan tilaukset niin, että toimitusmaksut pysyvät kohtuullisina.

Kun päätös on tehty, ostopäällikkö tekee tilauksen toimittajan verkkokaupassa ja kirjaa sen yrityksen järjestelmään ostotilauksena. Päivittäisessä työssä korostuu myös yksittäisten tuotteiden hintojen ja perustietojen tarkistus sekä tarvittaessa niiden päivitys yrityksen järjestelmään, jos toimittaja on muuttanut tietoja.

Myymälän asiakastilaukset

Kolmas työputki on myymälän asiakastilaukset. Jos asiakas haluaa tuotteen, jota ei löydy varastosta, myyjä voi tilata sen asiakkaalle myynnin yhteydessä. Käytännön toteutus vaihtelee toimittajittain ja tapauskohtaisesti, ja joissain tilanteissa tilaaminen delegoidaan ostopäällikölle. Asiakastilauksia tehdään selvästi vähemmän kuin massatilauksia tai varastontäydennyksiä, joten niitä ei käsitellä tässä työssä tarkemmin.

Seuraavassa osiossa (4.3) tämän prosessikuvauksen pohjalta tunnistetut keskeiset havainnot tiivistetään teemoiksi ja havainnollistetaan esimerkeillä.

4.3 Kipupisteet ja niiden ilmeneminen

Empiirinen tarkastelu nosti esiin toistuvia käytännön ongelmia, jotka kuormittavat ostoprosessia ja vievät aikaa varsinaiselta suunnittelulta. Tässä osiossa havainnot on ryhmitelty neljään teemaan:

1. Volyymi, kausivaihtelu ja läpimenoaika
2. Kustannuslaskenta ja hinnoittelun hallinta
3. Valikoimapäätökset ja päätösten perustelut (hiljainen tieto)
4. Tuotetiedon päivitys, hajanaisuus ja jäljitettävyys

Volyymi, kausivaihtelu ja läpimenoaika

Massatilaukset ja sesonkivalmistelu muodostavat selkeän kuormituspiikin. Ostopäällikön arvion mukaan yhden sesonkikierroksen aikana saapuu tyypillisesti noin kahdeksan tai yhdeksän toimittajataulukkoa. Yksittäisen taulukon rivimäärä voi olla satoja tai tuhansia rivejä. Esimerkkitaulukoissa rivimääriä oli noin 1 200, 2 200 ja 2 400.

Työn kesto vaihtelee taulukon koon ja sisällön mukaan. Pienimmän taulukon läpikäyntiin kuluu ostopäällikön arvion mukaan yleensä noin kaksi työpäivää. Suurimman taulukon käsittely voi viedä jopa kaksi viikkoa. Kuormitusta lisää se, että taulukot eivät tule tasaisesti pitkin vuotta, vaan painottuvat samoihin ajankohtiin. Tyypillisesti kesärengastaulu-

koita käsitellään joulukuusta eteenpäin ja talvirengastaulukoita toukokuussa. Käytännössä ennen sesonkia noin kaksi kuukautta voi kulua siihen, että taulukot saadaan käytyä läpi ja valinnat tehtyä.

Aikataulupaine näkyy arjessa siinä, että viimeiset taulukot tulevat usein myöhässä. Tällöin käsittely jää kiireiseksi ja osa työstä siirtyy jo sesongin kynnykselle. Tämä kasvattaa riskiä, koska yrityksen on saatava valinnat valmiiksi ajoissa ja varmistettava halutut tuotteet ennen kilpailijoita.

Päivittäinen osto ja varastontäydennys kuormittavat eri tavalla. Sesongin ulkopuolella päivittäisiin tilauksiin kuluu ostopäällikön arvion mukaan tyypillisesti noin tunti päivässä. Sesonkien alla sama työ voi viedä yhteensä jopa kahdeksan tuntia päivässä, kun täydennyksiä tehdään useaan varastoon ja myynti on vilkkaampaa. Kuormitushuiput eivät yleensä osu täysin päällekkäin sesonkivalmistelun kanssa, mutta yksittäisinä päivinä kokonaiskuorma voi silti nousta niin suureksi, että liikkumavaraa muulle työlle jää vähän.

Kustannuslaskenta ja hinnoittelun hallinta

Toinen selkeä kipupiste liittyy tuotteiden ostohinnan laskentaan ja hinnoitteluun. Jokaisella rengastoimittajalla on omalla tavallaan rakennetut taulukot, eikä yhtä yhteistä mallia ole. Yhteen taulukkoon rakennettu laskenta ei siirry sellaisenaan seuraavaan, vaan toimittajakohtaiset käytännöt ja säännöt ohjaavat laskelmia.

Massatilauksissa ostopäällikkö tekee toimittajalta saadusta taulukosta oman version ja lisää siihen sarakkeita, joilla lasketaan tuotteelle yrityksen oma todellinen ostohinta. Toimittajan ilmoittamaa lähtöhintaa rikastetaan huomioimalla esimerkiksi kampanjat, alennukset, volyymibonukset, kierrätysmaksut ja muut sovitut kulut. Samassa yhteydessä ostopäällikkö arvioi myyntihintaa ja määrittää tarvittaessa myös efektiivisen ostohinnan.

Efektiivinen ostohinta on järjestelmän sisäinen hinnoittelun apuväline. Ostopäällikkö voi asettaa tuotteelle efektiivisen ostohinnan, jolloin järjestelmä voi käyttää sitä todellisen ostohinnan sijasta automaattisissa hinnoittelulaskelmissa. Näiden laskelmien pohjalta muodostuvat erityisesti asiakas- ja alennusryhmien hinnat. Tällaisia ryhmiä ovat esimerkiksi autoliikkeet, tukkuasiakkaat ja erilaiset sopimusasiakkaat. Käytännössä efektiivisellä ostohinnalla ohjataan sitä, miten alennusryhmien hinnat ja tuotteesta saatava kate asettuvat eri myyntitilanteissa. Samalla se antaa mahdollisuuden säätää hinnoittelua ylös tai alas ilman, että kaikkia ryhmähintoja tarvitsee laskea ja syöttää käsin.

Efektiivinen ostohinta helpottaa hinnoittelun ohjausta, mutta lisää samalla tarkistettavia kohtia. Kun uusi tuote otetaan valikoimaan tai tuotetietoja päivitetään, ostopäällikkö tarkistaa, että myyntihinta, efektiivinen ostohinta ja eri ryhmille muodostuvat hinnat ovat keskenään järkevässä suhteessa. Pelkkä todellisen ostohinnan laskeminen ei siis riitä.

Lisäksi ostopäällikkö varmistaa käytännössä, ettei yksittäinen alennusryhmä jää liian korkealle tai liian matalalle tasolle, mikä näkyy työssä lisätarkistuksina ja tarvittaessa käsin tehtyinä korjauksina.

Hinnoitteluvirheitä syntyy ostopäällikön arvion mukaan harvakseltaan, mutta ne liittyvät usein juuri efektiiviseen ostohintaan ja alennusryhmiin. Virhe voi syntyä siitä, että efektiivinen ostohinta arvioidaan väärin tai jokin asiakasryhmä jää tarkistamatta. Ongelma huomataan usein vasta myynnin tai raportoinnin kautta, joskus asiakkaan palautteesta. Korjaus tarkoittaa tyypillisesti sitä, että efektiivistä ostohintaa säädetään tai jonkin ryhmän hinnat irrotetaan automatiikasta ja määritellään kiinteiksi.

Valikoimapäätökset ja päätösten perustelut (hiljainen tieto)

Kolmas teema liittyy valikoiman rajaukseen ja siihen, miten tuotteet lopulta valitaan ja hinnoitellaan. Kaikkia toimittajataulukon rivejä ei ole tarkoitus tilata, vaan suuri osa tuotteista jää pois jo alkuvaiheessa. Esimerkkitapauksessa noin 1 500 rivin taulukosta päätyi käyttöön arviolta 200–300 tuotetta.

Karsinta ja hinnoittelu nojaavat vahvasti ostopäällikön hiljaiseen tietoon. Päätöksissä yhdistyy aiempi myynti, kysyntä, autokanta, toimittajakohtaiset linjaukset, varastojen kapasiteetti ja kokemus siitä, mikä toimii eri asiakasryhmille. Taustalla hyödynnetään myös dataa, kuten myyntiraportteja ja aiempia taulukoita, mutta lopullinen johtopäätös on usein kokemusperäinen. Tämä näkyy erityisesti harvinaisemmissa kokoluokissa ja erikoismalleissa, joissa yksittäinen virhe voi tarkoittaa pitkään hyllyssä seisovia tuotteita.

Tässä haaste ei ole se, että päätökset perustuvat asiantuntijan näkemykseen. Se on monessa kohdassa käytännössä välttämätöntä. Haaste syntyy siitä, että perustelut eivät tallennu näkyvään ja siirrettävään muotoon. Taulukossa ja järjestelmässä näkyy lopputulos, mutta syyt hylkäämiselle, poikkeavalle hinnoittelulle tai varastokohdistukselle jäävät pitkälti ostopäällikön muistin varaan. Tämä vaikeuttaa työn jakamista useammalle henkilölle ja tekee sijaisuuksista hankalampia, erityisesti massatilausten ja valikoiman kehittämisen osalta.

Tuotetiedon päivitys, hajanaisuus ja jäljitettävyys

Neljäs teema liittyy siihen, miten tieto siirtyy toimittajataulukoista yrityksen järjestelmään ja miten se löytyy myöhemmin uudestaan. Kun massatilauksen valinnat on tehty, tuotteiden hinnat ja muut perustiedot kirjataan järjestelmään rivi kerrallaan. Uusille tuotteille luodaan tuotekortit ja vanhoille päivitetään vähintään hinta ja tarvittaessa muita tietoja.

Koska järjestelmä ei lue toimittajataulukoita suoraan, sama tieto elää jonkin aikaa sekä taulukossa että järjestelmässä. Valinnat ja laskenta tehdään kopiotaulukossa, lopputulos

kirjataan järjestelmään ja toimittajan tilausvahvistus jää sähköpostiin. Jos myöhemmin halutaan selvittää, miksi jokin malli hinnoiteltiin tietyllä tavalla tai miksi varastosta puuttuu tietty tuote, tieto joudutaan kokoamaan vanhoista taulukoista, sähköposteista ja muista muistijäljistä.

Tämä näkyy erityisesti massatilausten yhteydessä, joissa pitäisi samalla arvioida varastokohdistusta useaan toimipisteeseen. Kohdistus tehdään yhdistämällä myyntiraportteja, varastosaldoja ja ostopäällikön näkemystä. Yhtenäistä näkymää päätöksille ei ole, vaan tieto jakautuu useisiin lähteisiin.

Seuraavassa osiossa (4.4) tarkennetaan toimittajaintegraatioihin liittyviä havaintoja Continentalin esimerkin ja EDIWheel-havaintojen kautta. Luvussa 4.5 nämä kipupisteet kytketään luvussa 2 esitettyyn viitekehykseen ja niiden pohjalta johdetaan ratkaisun vaatimukset.

4.4 Toimittajaintegraatioprofiilit ja EDIWheel-havainnot

Continentalin integraatiodokumentaatio osoittaa, että yhden toimittajan integraatio rakentuu käytännössä useasta rinnakkaisesta tavasta välittää tuotetietoa ja tilauksiin liittyviä tapahtumia (Continental, n.d.). Continental hyödyntää sähköistä tiedonsiirtoa (Electronic Data Interchange, EDI) massamuotoiseen dataan, kuten hinnastoihin ja varastosaldoihin. Yksittäisiin pyyntöihin ja tilauksiin se käyttää erillistä EASY XML -rajapintaa, joka on Continentalin XML-pohjainen rajapinta (Continental, n.d.).

Continentalin EDI-sanomat vastaavat tässä työssä aiemmin esiteltyjä GS1 EANCOM PRICAT- ja INVRPT-sanomia. Näissä tuotetiedot ja saldot välitetään eräaineistona, jossa tuotteet esitetään rakenteisesti taulukkomaisessa muodossa (GS1, 2016/2021a; GS1, 2016/2021b; Continental, n.d.). Sanomien avulla toimittajan hinnastot, tuoteluettelot ja varastosaldot voidaan siirtää ostavan yrityksen järjestelmään. EASY XML toimii tämän massakerroksen rinnalla rajapintana, jonka kautta voidaan tehdä yksittäisiä saatavuus- ja hintakyselyjä, lähettää tilauksia ja hakea tilausstatuksia (Continental, n.d.). Sama tuotetieto voi siten näkyä sekä EDI-sanomissa että EASY XML -rajapinnan vastauksissa, mutta eri tarkkuudella ja eri käyttötarkoituksessa.

Continental viittaa dokumentaatiossaan rengasalan yhteiseen EDIWheel-standardiin (Continental, n.d.; EDIWheel, n.d.). EDIWheel on rengaskauppaa varten kehitetty EDI-standardi, joka määrittelee, miten rengasteollisuuden ja rengaskaupan väliset prosessit, kuten varastokyselyt, tilaukset, toimitukset ja laskutus, esitetään sähköisinä sanomina (EDIWheel, n.d.). Standardi pohjautuu GS1 EANCOM -määrityksiin. Tämä tarkoittaa, että hinnasto- ja varastotiedot kuvataan rakenteisesti samaan tapaan kuin PRICAT- ja

INVRPT-sanomissa, joissa tuotetunnisteet ja keskeiset peruskentät on määritelty yhtenäiseen muotoon (GS1, 2016/2021a; GS1, 2016/2021b; EDIWheel, n.d.). EDIWheelin oman kuvauksen mukaan standardi on laajassa käytössä rengasteollisuuden ja rengaskaupan välisessä liiketoiminnassa (EDIWheel, n.d.).

Näiden lähteiden perusteella tässä työssä oletetaan, että useiden rengasvalmistajien integraation peruslogiikka on pitkälti samankaltainen, vaikka toteutus ei ole täysin identtinen toimittajasta toiseen. Continentalin tapauksessa massadata siirtyy EDIWheel-profiilin mukaisina sanomina, kun taas yksittäiset transaktiot kulkevat EASY XML -rajapinnan kautta (Continental, n.d.; EDIWheel, n.d.). EDIWheel määrittelee sanomien rakenteen ja kentät rengasalalle sopiviksi (EDIWheel, n.d.). Käytännössä toimittajalle jää silti liikkumavaraa, koska se voi rajata, mitä kenttiä se käyttää ja millä integraatiotekniikoilla sanomat välitetään.

Näiden havaintojen pohjalta tässä työssä suunniteltu ratkaisu pohjautuu malliin, jossa EDIWheel- tai muu vastaava aineisto luetaan ensin toimittajakohtaisen integraation kautta ja vasta sen jälkeen muunnetaan yhteiseen kanoniseen tuotemalliin. Näin sanomien perusrakenne voidaan hyödyntää uudelleen eri toimittajien välillä, mutta integraatioarkkitehtuuri säilyttää joustavuuden erilaisten formaattien, kenttävalintojen ja kanavien suhteen (Continental, n.d.; EDIWheel, n.d.)

4.5 Yhteys teoreettiseen viitekehykseen ja ratkaisun vaatimukset

Luvussa 4.3 empiiriset havainnot ryhmiteltiin neljään teemaan. Tässä alaluvussa teemat kytketään luvussa 2 esitettyyn teoriaan ja niiden pohjalta johdetaan ratkaisulle vaatimukset. Teema kuvaa, mihin kokonaisuuteen havainto liittyy. Kipupiste kuvaa, mikä siinä kokonaisuudessa kuormittaa työtä tai aiheuttaa riskin. Vaatimus määrittää, mitä ratkaisun on pystyttävä tekemään, jotta kipupisteeseen voidaan vaikuttaa.

Teema 1: Volyymi, kausivaihtelu ja läpimenoaika

Massatilauksissa rivimäärät nousevat tuhansiin ja läpivienti voi kestää jopa viikkoja. Suurin osa työstä on sesonkitaulukoiden manuaalista käsittelyä ja saman perusdatan toistuvaa läpikäyntiä useiden toimittajien taulukoista. Prosessi on osittain Nettivanne-järjestelmän sisällä, mutta sesonkivaiheessa työ siirtyy käytännössä taulukoihin ja sähköposteihin.

Kirjallisuudessa PK-yritysten digitalisointia kuvataan vaiheittaisena etenemisenä, jossa kehitys kohdistetaan ensin kohtiin, joissa manuaalinen työ aiheuttaa eniten viivettä ja virheriskiä (Aliyev, 2025; Vial, 2019). Omowole et al. (2024) nostavat esiin pilvipalvelut,

analytiikan ja automaation käytännön työkaluina, joilla voidaan tehdä rajattuja parannuksia ilman koko arkkitehtuurin uudistamista kerralla. Tämän työn kannalta tämä tukee mallia, jossa sesonkityön raskain vaihe otetaan ensin hallintaan, ja vasta sen jälkeen laajennetaan ratkaisua muualle ostoprosessiin.

Rengas Turun tapauksessa massataulukkotyöskentely ja siihen liittyvä manuaalityö osuvat juuri sellaiseen korkean kuorman ja virheriskin kohtaan, josta digitalisointi on perusteltua aloittaa (Aliyev, 2025; Vial, 2019). Koko ostoprosessin digitalisointi kerralla olisi laaja prosessiuudistus, joka kasvattaa sekä kuormaa että toteutusriskiä. Siksi eteneminen kannattaa toteuttaa vaiheittain. Samalla havaintojen pohjalta tunnistetaan, että pelkät väliaikaiset helpotukset eivät muuta ongelmien perustilannetta, jos varsinainen sesonkityö jatkuu taulukoiden varassa. Tämän vuoksi ratkaisun on vietävä prosessia aidosti kohti digitaalista toimintamallia, eikä jäädä vain nykyisen työskentelyn kevyeksi uudistamiseksi.

Kipupiste 1: Sesonkitaulukoiden käsittelyyn liittyvä manuaalityö ja aikataulupaine kasaantuvat lyhyelle ajalle ja aiheuttavat sekä viivettä että virheriskiä.

Vaatimus 1: Ratkaisun pitää vähentää sesonkitaulukoiden manuaalityötä ja mahdollisuuksien mukaan korvata taulukkoprosessi digitaalisella ratkaisulla, joka käsittelee nykyisen tuotemäärän paremmin ja säilyttää päätösvallan ostajalla. Laajempi uudistus toteutetaan vaiheittain siten, että ensimmäisenä kehitetään eniten kuormittava vaihe (Aliyev, 2025; Vial, 2019).

Teema 2: Kustannuslaskenta ja hinnoittelun hallinta

Toinen teema liittyy ostohinnan laskentaan ja hinnoitteluun. Laskenta tehdään toimittajakohtaisilla taulukoilla, eikä yhtä yhteistä toimintatapaa ole. Ostohinta muodostuu useista osista, kuten myyntibonuksista, kampanjoista, viranomaismaksuista ja muista kuluista. Hinnoittelussa käytetään lisäksi yrityksen sisäistä efektiivistä ostohintaa, jota järjestelmä voi hyödyntää automaattisissa hinnoittelulaskelmissa, jos näin on määritetty. Tällöin esimerkiksi asiakas- ja alennusryhmien hinnat voivat muodostua efektiivisen ostohinnan perusteella.

Ellram (1995) korostaa, että pelkkä listattu yksikköhinta ei kuvaa todellisia kustannuksia, vaan tarkasteluun on sisällytettävä myös alennukset, lisämaksut ja muut ostamiseen liittyvät erät. Tätä taustaa vasten efektiivisen ostohinnan käyttö muistuttaa yrityksen omaa kevyttä omistamisen kokonaiskustannuksen (Total Cost of Ownership, TCO) ajattelua, mutta malli ja säännöt eivät ole kuvattuja eikä toimintatapa ole yhtenäinen (Ellram, 1995). Tämä näkyy käytännössä tilanteina, joissa ostohinta, kampanjat ja alennusryhmät muodostavat risteäviä hintapisteitä, ja lopputulos varmistetaan manuaalisesti.

Ramakrishnan et al. (2019) osoittavat, että manuaalisesti ylläpidetty hinnoittelu on virheherkkää ympäristöissä, joissa tuotteita ja hintapisteitä on paljon. Heidän mukaansa jo yksittäiset virheet voivat vaikuttaa kannattavuuteen, jos poikkeamia ei tunnisteta ajoissa (Ramakrishnan et al., 2019). Tässä työssä havaittu ilmiö on sama. Virheet eivät yleensä synny yhdestä selkeästä laskukaavasta, vaan useista risteävistä säännöistä ja monista hintapisteistä. Tämän vuoksi ratkaisun pitää tuoda kustannusrakenne näkyväksi ennen päätöstä ja tukea poikkeamien tunnistamista, jotta manuaalinen tarkistustyö vähenee.

Kipupiste 2: Ostohinnan ja efektiivisen ostohinnan laskenta sekä alennusryhmien hinnoittelu sisältävät paljon manuaalista tarkistusta ja virheriskejä.

Vaatimus 2: Ratkaisun pitää tukea ostohinnan ja efektiivisen ostohinnan laskentaa tai tarvittaessa korvata nykyinen efektiivisen hinnan malli TCO-ajatteluun sopivalla paremmalla ratkaisulla. Kustannusten tulee näkyä selkeästi ennen päätöstä, ja järjestelmän tulee auttaa tunnistamaan virheet (Ellram, 1995; Ramakrishnan et al., 2019).

Teema 3: Valikoimapäätökset ja päätösten perustelut (hiljainen tieto)

Kolmas teema liittyy valikoiman rajaukseen, korvattavuuden arviointiin ja päätöksenteon tukemiseen. Empiiristen havaintojen perusteella suuri osa toimittajataulukoiden riveistä jätetään tilaamatta, ja karsinta nojaa pitkälti ostopäällikön kokemukseen kysynnästä, tuotekorvattavuudesta ja autokohtaisista tiedoista. Perustelut eivät tallennu rakenteisesti mihinkään, vaan näkyvissä on käytännössä vain lopullinen valinta. Tämä lisää henkilöriippuvuutta ja tekee sijaisuuksista hankalia, koska päätösten taustaa ei voi tarkistaa jälkikäteen.

Power (2002) määrittelee päätöksenteon tukijärjestelmän järjestelmäksi, joka kokoaa dataa, malleja ja asiakirjoja asiantuntijan harkinnan tueksi ilman, että päätöstä automatisoidaan (Power, 2002). Kulkarni et al. (2007) kuvaavat DSS-ratkaisuja datapohjaisina ja mallipohjaisina lähestymistapoina. Datapohjainen DSS kokoaa päätöksen kannalta oleellista dataa yhteen näkymään, ja mallipohjainen DSS tukee vaihtoehtojen vertailua esimerkiksi skenaarioiden kautta (Kulkarni et al., 2007). Nykytilassa perustelut ja päätöspolku eivät ole myöhemmin selkeästi nähtävissä, eikä käyttäjä saa niiden avulla tukea päätöksentekoon. Tällöin päätöksenteon tuki jää käytännössä yksittäisen henkilön muistin varaan.

Kipupiste 3: Valikoiman rajauksen ja poikkeusten perustelut jäävät hiljaiseksi tiedoksi, eikä nykytilassa ole rakennetta, jonka varaan päätöksenteon tukea ja myöhempää suosittelua voidaan rakentaa läpinäkyvästi.

Vaatimus 3: Ratkaisun pitää tallentaa valikoiman rajauksen ja hinnoittelun kannalta olennaiset perustelut digitaaliseen muotoon ja tarjota tulevaisuudessa DSS-tyyppinen

näkymä, jossa data ja suositukset ovat esillä. Järjestelmän pitää pystyä tukemaan ostajaa päätöksenteossa hintojen, tuotevalikoiman, tilausmäärien ja muiden olennaisten asioiden kanssa. Näin riippuvuus yksittäisestä asiantuntijasta pienenee ja työtä voidaan jakaa useammalle henkilölle (Kulkarni et al., 2007; Power, 2002).

Teema 4: Tuotetiedon päivitys, hajanaisuus ja jäljitettävyys

Neljäs teema liittyy siihen, miten oston kannalta olennainen tieto liikkuu toimittajataulukoiden, sähköpostien ja Nettivanne-järjestelmän välillä. Massatilauksissa valinnat ja laskenta tehdään kopiotaulukoissa, minkä jälkeen tiedot syötetään käsin järjestelmään. Sama tieto elää hetken aikaa sekä taulukossa että järjestelmässä, ja päätösten taustat jäävät sähköposteihin tai henkilön muistiin. Tällöin jäljitettävyys on heikko ja manuaalinen massapäivitys kuormittaa työtä.

Abraham (2014) määrittelee tuotetiedon hallinnan (PIM) toimintatapana, jossa tuotetieto kootaan yhteen paikkaan ja jaetaan eteenpäin yhtenäisessä muodossa. Khatri ja Brown (2010) korostavat tiedonhallinnan ohjausta vastuiden ja käytäntöjen kokonaisuutena, jolla tieto pidetään hallittavana, kun sitä käytetään useassa prosessissa ja järjestelmässä. Nykyinen tilanne, jossa oston kannalta kriittinen tieto on hajallaan taulukoissa, sähköposteissa ja Nettivanteessa, on ristiriidassa tämän periaatteen kanssa.

Nettivanne toimii käytännössä yrityksen omana tuotetietovarastona, mutta ostoprosessin näkökulmasta tiedon hallinta ja päivitys tapahtuvat edelleen pitkälti taulukoiden kautta. GS1-viitekehys ja sanomamallit tarjoavat käytännön tukea tuotetiedon yhdenmukaistamiseen, kun tietoa halutaan yhdistää useista lähteistä samaan tuotteeseen (GS1, 2016/2021a; GS1, 2016/2021b; GS1, 2022). Kun tähän yhdistetään integraatioperiaatteet, joissa massatiedostot käsitellään kerran ja virheelliset tapaukset ohjataan erilliseen käsittelyyn, siirtymä taulukoista rajapintoihin on toteutettavissa hallitummin (Hohpe & Woolf, 2004).

Kipupiste 4: Oston kannalta olennainen tuotetieto on hajallaan, sama asia elää rinnakkain useassa paikassa ja manuaalinen massapäivitys heikentää jäljitettävyyttä.

Vaatimus 4: Ratkaisun pitää keskittää oston kannalta olennainen tuotetieto yhteen kanoniseen tuotemalliin ja tukea GS1-tunnisteita. Tavoite on siirtyä taulukoista ja manuaalisesta tuotteiden siirrosta rajapintoihin, joiden kautta toimittajien tuotteita voidaan hallita ja tilata (Abraham, 2014; GS1, 2016/2021a; GS1, 2016/2021b; Khatri & Brown, 2010).

Arkkitehtuuri ja vaiheittainen toteutus

Tämä vaatimus ei perustu suoraan luvun 4.3 neljään teemaan, vaan se on ratkaisun toteutuksen reunaehto. Yrityksen nykyinen ydinjärjestelmä perustuu Python-Django-pinoon ja on jaettu omiin Django-sovelluksiin, jolloin kokonaisuus vastaa modulaarista monoliittia (Holovaty & Kaplan-Moss, 2009; Richards & Ford, 2020). Ratkaisun toteutuksen pitää olla yhteensopiva nykyjärjestelmän kanssa, jotta eteneminen voidaan tehdä hallitusti eikä ydintoimintoja tarvitse kirjoittaa laajasti uudelleen.

Kirjallisuudessa modulaarinen monoliitti kuvataan usein luontevana lähtökohtana ympäristöissä, joissa muutokset halutaan tehdä hallitusti. Mikropalvelut voivat olla perusteltuja tietyissä tilanteissa, mutta ne kasvattavat alkuvaiheen vaatimuksia toimitusputkelle ja operoinnille (Newman, 2021; Pahl & Jamshidi, 2016; Taibi et al., 2017). Siksi tässä työssä arkkitehtuurivalinnan ydin on se, että valittu malli tukee integraatioita ja vaiheittaista etenemistä ilman, että nykyjärjestelmää joudutaan kirjoittamaan laajasti uudelleen (Richards & Ford, 2020; Newman, 2021; Pahl & Jamshidi, 2016).

Vaatimus 5: Ratkaisun pitää nojata nykyiseen Django-pohjaiseen järjestelmään ja tukea vaiheittaista etenemistä. Ratkaisun on liityttävä Nettivanteeseen selkeästi rajattuna osana, ja toteutus voi perustua joko uuteen moduuliin nykyisessä järjestelmässä, erilliseen mikropalveluratkaisuun tai muuhun yhteensopivaan toteutustapaan.

Yhteenveto vaatimuksista

Tässä alaluvussa kytkettiin luvussa 4.3 tunnistetut teemat luvussa 2 esitettyyn teoriaan ja johdettiin niiden pohjalta viisi vaatimusta ratkaisulle:

1. Ratkaisun pitää vähentää sesonkitaulukoiden manuaalityötä ja mahdollisuuksien mukaan korvata taulukkoprosessi digitaalisella ratkaisulla, joka käsittelee nykyisen tuotemäärän paremmin ja säilyttää päätösvallan ostajalla.
2. Ratkaisun pitää tukea ostohinnan ja efektiivisen ostohinnan laskentaa tai tarvittaessa korvata nykyinen efektiivisen hinnan malli TCO-ajatteluun sopivalla paremmalla ratkaisulla. Kustannusten tulee näkyä selkeästi ennen päätöstä, ja järjestelmän tulee auttaa tunnistamaan virheet.
3. Ratkaisun pitää tallentaa valikoiman rajauksen ja hinnoittelun kannalta olennaiset perustelut digitaaliseen muotoon ja tarjota tulevaisuudessa DSS-tyyppinen näkymä, jossa data ja suositukset ovat esillä. Järjestelmän pitää pystyä tukemaan ostajaa päätöksenteossa hintojen, tuotevalikoiman, tilausmäärien ja muiden olennaisten asioiden kanssa.

4. Ratkaisun pitää keskittää oston kannalta olennainen tuotetieto yhteen kanoniseen tuotemalliin ja tukea GS1-tunnisteita. Tavoite on siirtyä taulukoista ja manuaalisesta tuotteiden siirrosta rajapintoihin, joiden kautta toimittajien tuotteita voidaan hallita ja tilata.
5. Ratkaisun pitää nojata nykyiseen Django-pohjaiseen järjestelmään ja tukea vaiheittaista etenemistä. Ratkaisun on liityttävä Nettivanteeseen selkeästi rajattuna osana, ja toteutus voi perustua joko uuteen moduuliin nykyisessä järjestelmässä, erilliseen mikropalveluratkaisuun tai muuhun yhteensopivaan toteutustapaan.

Yhdessä nämä vaatimukset kuvaavat sen tason, jolle ratkaisun on yllettävä, jotta ostoprosessi saadaan uudistettua riittävästi. Seuraavassa luvussa tarkastellaan ratkaisuvaihtoehtoja ja arkkitehtuurimalleja näiden vaatimusten näkökulmasta.

4.6 Yhteenveto

Tässä luvussa kuvattiin Rengas Turku Oy:n ostoprosessin nykytila rajatusti kahteen keskeiseen työputkeen: massatilaukset ja sesonkivalmistelu sekä päivittäinen varastontäydennys. Tavoite oli kuvata selkeästi, mitä työssä käytännössä tehdään, missä järjestyksessä työ etenee ja missä kohdissa työ kuormittuu tai altistuu virheille, ennen kuin työ siirtyy ratkaisuvaihtoehtojen vertailuun luvussa 5.

Empiiristen havaintojen perusteella keskeiset ongelmakohdat tiivistyivät neljään toistuvaan teemaan:

1. **Volyymi, kausivaihtelu ja läpimenoaika:** sesonkitaulukot ovat suuria, tulevat lyhyelle ajanjaksolle ja pakottavat työn kiireeseen.
2. **Kustannuslaskenta ja hinnoittelun hallinta:** ostohinnan muodostus vaatii toimittajakohtaista laskentaa, ja lisäksi hinnoitteluun liittyy useita tarkistuspisteitä (esim. efektiivinen ostohinta ja asiakas-/alennusryhmät), mikä kasvattaa virheriskiä.
3. **Valikoimapäätökset ja päätösten perustelut (hiljainen tieto):** valikoiman rajaus ja poikkeavat ratkaisut nojaavat vahvasti ostopäällikön kokemukseen, mutta perustelut eivät tallennu rakenteisesti, mikä lisää henkilöriippuvuutta ja vaikeuttaa sijaisuuksia.
4. **Tuotetiedon päivitys, hajanaisuus ja jäljitettävyys:** sama tieto elää taulukoissa, sähköposteissa ja järjestelmässä rinnakkain, ja massapäivitykset tehdään käsin, jolloin päätösten taustat ovat vaikeasti jäljitettävissä.

Luvun lopussa kipupisteet kytkettiin luvussa 2 esitettyyn viitekehykseen ja johdettiin ratkaisulle viisi vaatimusta, jotka ohjaavat toteutusta ja rajaavat vertailua luvussa 5:

1. Sesonkitaulukoiden manuaalityön selkeä vähentäminen ja eteneminen kohti digitaalista prosessia.
2. Ostohinnan ja efektiivisen ostohinnan laskennan sekä hinnoittelun virheriskin pienentäminen, ja kustannusrakenteen näkyvyys ennen päätöksiä.
3. Päätösten perustelujen tallentaminen ja päätöksenteon tuen mahdollistaminen (DSS).
4. Tuotetiedon keskittäminen kanoniseen tuotemalliin (PIM) ja GS1-tunnisteiden tukeminen sekä siirtymä taulukoista rajapintoihin.
5. Yhteensopivuus nykyisen Django-pohjaisen ydinjärjestelmän kanssa ja vaiheittainen käyttöönotto.

Seuraavassa luvussa ratkaisuvaihtoehtoja ja arkkitehtuurimalleja vertaillaan näiden vaatimusten avulla, ja valinta perustellaan suoraan tässä luvussa kuvattuun nykytilaan ja havaittuihin ongelmakohtiin.

5. RATKAISUVAIHTOEHTOJEN VERTAILU JA VALINTA

Tässä luvussa valitaan arkkitehtuuri, joka toimii tämän työn pohjana ja tukee parhaiten Rengas Turku Oy:n ostoprosessin uudistamista. Tarkastelussa on neljä vaihtoehtoa:

A. nykyjärjestelmään toteutettava Django-laajennus,

B. erillinen mikropalvelu,

C. kaupallinen työkalu sekä

D. prosessin kehittäminen ilman uutta järjestelmää.

Vertailu nojaa luvussa 2 esitettyyn teoreettiseen viitekehykseen, luvun 4 empiirisiin havaintoihin ja niiden pohjalta johdettuihin vaatimuksiin sekä yrityksen asiantuntijoiden kanssa käytyihin keskusteluihin, jotka on koottu yrityspalaverit liitteeseen (ks. Liite 1, ”Yrityspalaverit”).

Vertailu tehdään painotetun summamallin (SAW) avulla. Menetelmässä vaihtoehdot pisteytetään ennalta määritellyillä kriteereillä, ja kriteereille annetaan painot yrityksen tavoitteiden mukaan. Kriteerit määritellään siten, että suurempi arvo on parempi, ja kustannus- ja riskikriteerit käsitellään käänteisesti. Kokonaistulos muodostetaan pisteiden ja kriteeripainojen painotettuna summana, josta metodin nimikin tulee. Menettely vastaa Tzengin ja Huangin (2011) esittämää MADM-kehystä (Multi-Attribute Decision Making, monikriteerinen päätöksentekomenetelmä) ja SAW-menetelmää, ja heidän suosituksensa mukaisesti tuloksia tarkastellaan myös herkkyysanalyysin avulla (Tzeng & Huang, 2011).

Kriteerit ja painot on määritellyt työn tekijä luvussa 3 kuvatun prosessin mukaisesti. Määrittely perustuu luvussa 4 esitettyihin kipupisteisiin ja vaatimuksiin, luvun 2 teoreettisiin lähtökohtiin, teknisiin reunaehtoihin sekä arviointipalaveriin, jossa käytiin läpi painotusten ja pisteytyksen perusteita. Ennen varsinaista pisteytystä tehtiin esikarsinta projektivaatimusten pohjalta. Tämän seurauksena SAW-menetelmää sovellettiin vain kahteen päävaihtoehtoon (A: Django-laajennus ja B: mikropalvelu), ja kaupallinen työkalu sekä prosessin kehittäminen ilman uutta järjestelmää rajattiin pois.

Luvussa käydään läpi ensin eri vaihtoehdot, sen jälkeen esitetään vertailukriteerit ja niille annetut painot, raportoidaan pisteytyksen tulokset ja tehdään valinta. Lopuksi arvioidaan keskeiset riskit ja esitetään varasuunnitelma etenemiselle. Vaatimukset on koottu luvussa 4.5 luvun 4 teemoista ja kipupisteistä, ja niitä käytetään tässä luvussa vertailun lähtökohtana.

5.1 Vaihtoehdot

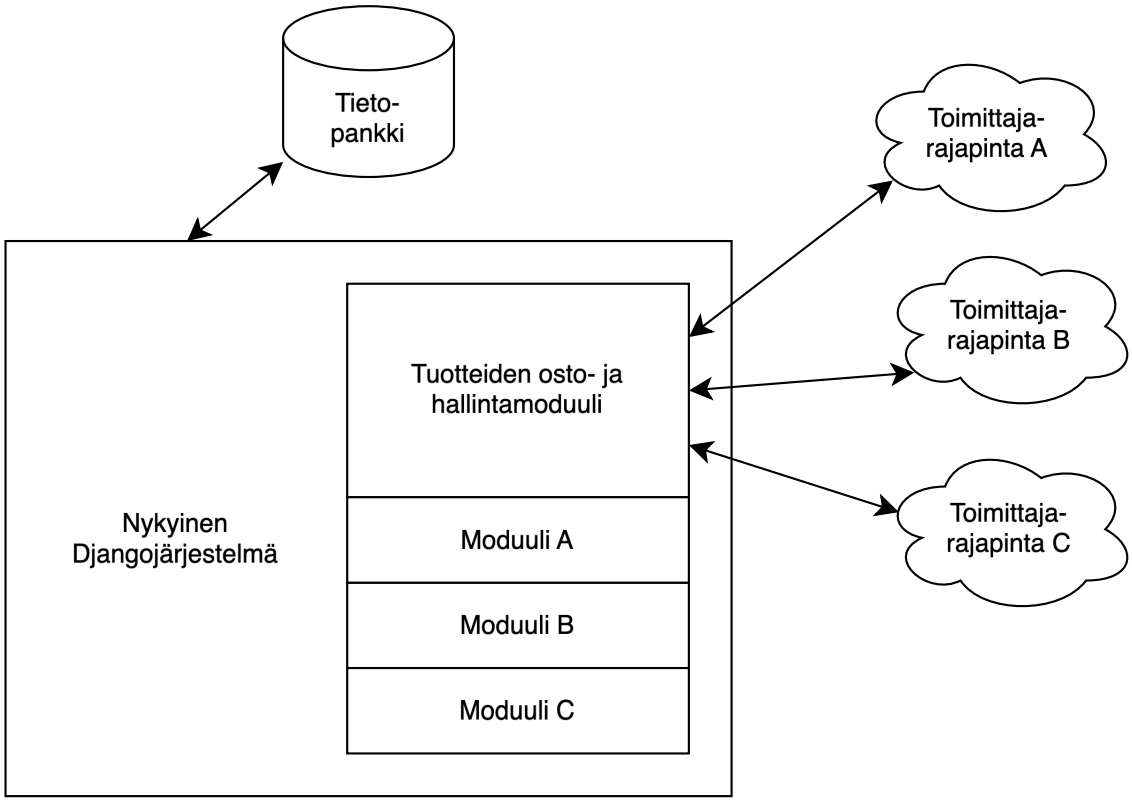
Tässä alaluvussa kuvataan neljä ratkaisuvaihtoehtoa ostoprosessin uudistamiselle. Vaihtoehdot on tunnistettu yhdistämällä luvussa 4 esitetyt empiiriset havainnot, luvun 2 teoria ja yrityksen asiantuntijoiden kanssa käydyt keskustelut, jotka on koottu yrityspalaveri-liitteeseen (ks. Liite 1, “Yrityspalaverit”). Tarkastelussa on sekä pienempiä että laajempia muutoksia, ja mukana on sekä kokonaan omaan panokseen perustuvia että ulkoisia ratkaisuja. Näin varmistetaan, että vertailu kattaa olennaiset etenemispolut ja ne voidaan arvioida selkein kriteerein painotetun summamallin (SAW) mukaisesti (Tzeng & Huang, 2011).

Vaihtoehto A – Nykyjärjestelmän (Django) laajennus

Vaihtoehdossa A nykyistä Django-pohjaista järjestelmää laajennetaan uudella tuotteiden osto- ja hallintamoduulilla. Moduuli käyttää jo olemassa olevia tietomalleja, rajapintoja ja rakennetta. Toteutustapa noudattaa modulaarisen monoliitin ajattelua, jossa sovellus jaetaan omiin Django-sovelluksiin, mutta kokonaisuus toimii edelleen yhtenä järjestelmänä (Holovaty & Kaplan-Moss, 2009; Richards & Ford, 2020). Modulaarisen monoliitin etuja PK-yritykselle on käsitelty tarkemmin luvussa 2.5.

Kirjallisuudessa modulaarinen monoliitti nähdään usein selkeänä lähtökohtana PK-yritykselle ja luontevana aloituspisteenä (Richards & Ford, 2020; Newman, 2021). Selkeä komponenttijako ja sisäiset rajapinnat helpottavat myöhempää eriyttämistä ilman, että arkkitehtuuria tarvitsee muuttaa kerralla. Käytännössä tämä tarkoittaa, että osto- ja hinnoittelumoduuli voidaan ensin rakentaa osaksi nykyistä Nettivanne-järjestelmää ja tarvittaessa irrottaa myöhemmin omaksi palvelukseen, jos kuormitus tai kehitystarpeet kasvavat.

Kuvassa 3 on esitetty vaihtoehto, jossa ostotyökalu rakennetaan osaksi olemassa olevaa Django-monoliittia. Kuva havainnollistaa, miten uusi ostomoduuli sijoittuu suhteessa nykyisiin sovelluksiin, tietokantaan ja integraatioihin. Ratkaisu hyödyntää olemassa olevaa teknistä perustaa ja lyhentää lyhyen aikavälin toteutusaikaa, mutta sitoo ostotoiminnan tiukasti muun järjestelmän rakenteeseen ja elinkaareen.



Kuva 3: *Tuotteiden osto- ja hallintamoduuli osana nykyistä järjestelmää.*

Rengas Turku Oy:n näkökulmasta vaihtoehto A:n vahvuus on, että Django-ympäristö ja nykyjärjestelmä ovat jo jokapäiväisessä käytössä ja osaamista on talossa. Aloituskynnys on matala, koska uusia teknologioita ei tarvitse ottaa käyttöön samaan aikaa prosessiuudistuksen kanssa. Samalla riskinä on, että uusi moduuli joutuu elämään nykyisen järjestelmän legacy-koodin keskellä ja skaalautuvuus on aluksi pääasiassa vertikaalista. Siksi modulaarisuuden ja sisäisten rajapintojen suunnittelu korostuu, jotta myöhempi mahdollinen irrottaminen erilliseksi palveluksi on edes mahdollista (Richards & Ford, 2020).

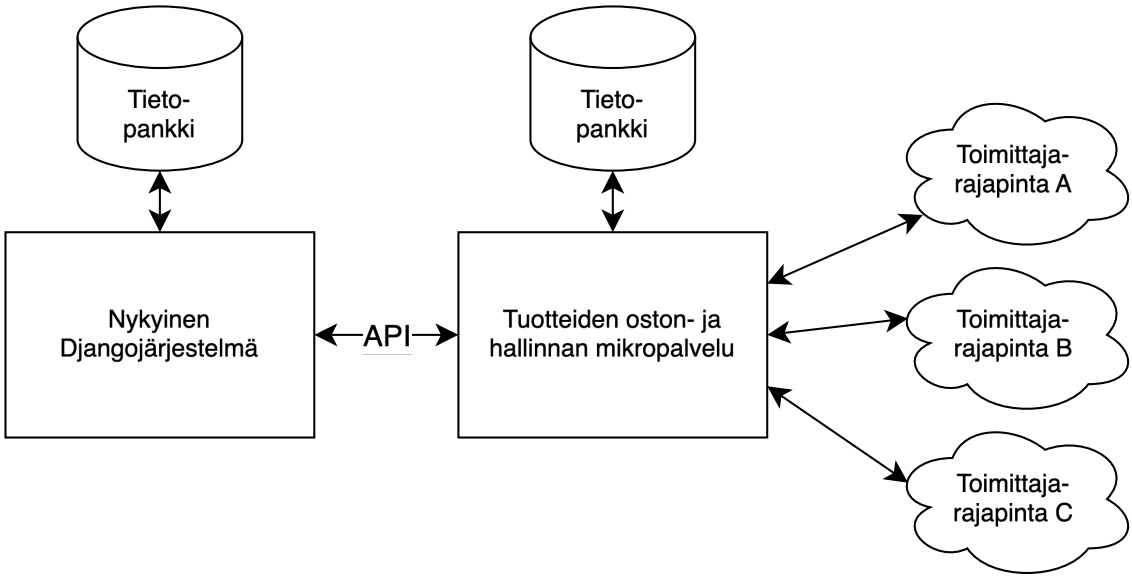
Vaihtoehto B – Erillinen mikropalvelu

Vaihtoehdossa B ostoprosessille rakennetaan itsenäinen mikropalvelu, jolla on oma tietovarasto ja rajapinta nykyjärjestelmään. Mikropalveluarkkitehtuurilta odotetaan etuja erityisesti itsenäisessä julkaistavuudessa, skaalautuvuudessa ja vikarajauksessa (Newman, 2021; Richards & Ford, 2020). Osto- ja hinnoittelulogiikka keskitetään yhteen palveluun, jota voidaan kehittää ja skaalata erikseen muusta järjestelmästä.

Hyötyjen saavuttaminen edellyttää kuitenkin, että taustalla on toimiva CI/CD-putki, valvonta ja selkeä operointimalli (Newman, 2021). Siirtymävaiheessa tämä lisää kehitys- ja ylläpitotyötä verrattuna pelkkään Django-laajennukseen. Integraatiorajapintojen suunnit-

telu ja elinkaaren hallinta nousevat tässä mallissa selkeään rooliin, koska nykyisen järjestelmän ja uuden palvelun välisen tiedonkulun pitää olla luotettavaa ja versionhallinnan hallittua.

Kuva 4 esittää vaihtoehdon, jossa ostotyökalu toteutetaan omana mikropalvelunaan, jolla on erillinen tietokanta ja rajapinnat muihin järjestelmiin. Kuva korostaa, että tämä ratkaisu vaatii enemmän suunnittelu- ja integraatiotyötä alkuvaiheessa, mutta erottaa ostotoiminnan logiikan muusta järjestelmästä ja helpottaa myöhempää laajentamista.



Kuva 4: *Erillinen mikropalvelu, joka yhdistetään nykyisen järjestelmään integraation avulla.*

Rengas Turku Oy:n kannalta vaihtoehto B tarjoaa pidemmällä aikavälillä eniten liikkumavaraa, jos ostoon ja PIM-logiikkaan liittyvä kuorma kasvaa selvästi tai toimittajaintegraatioiden määrä moninkertaistuu. Lisäksi erillinen mikropalvelu rajaa oston ja hinnoittelun koodin erilleen, mikä voi pitkällä aikavälillä helpottaa juuri tämän osa-alueen ylläpitoa ja räätälöintiä verrattuna siihen, että sama logiikka on osana laajaa monoliittia (Newman, 2021; Richards & Ford, 2020). Toisaalta nykytilanteessa CI/CD-putket, valvonta ja operointimalli eivät ole yhtä pitkällä kuin kirjallisuuden mikropalveluesimerkeissä. Käytännössä tämä tarkoittaa, että mikropalveluun siirtyminen vaatii alussa selvästi vaihtoehtoa A enemmän työtä, ja osa kehityspanoksesta menee arkkitehtuurin ja infrastruktuurin rakentamiseen itse varsinaisen ratkaisun sijaan. Lisäksi vaikka tässä työssä käsitellään pelkästään renkaita tuotekategoriana, on mahdollista, että mikropalvelurakenne helpottaa useamman tuotekategorian lisäämistä ostotyökaluun myöhemmin.

Vaihtoehto C – Kaupallinen tai ulkoinen työkalu

Vaihtoehdossa C käytetään kaupallista tai ulkoista ohjelmistoa ostoprosessin digitalisointiin. Tyypillisesti kyse on valmiista kaupallisesta ohjelmistosta (Commercial Off-The-

Shelf, COTS) tai pilvipalveluna tarjottavasta ohjelmistosta (Software as a Service, SaaS), joka liitetään nykyiseen järjestelmään rajapintojen avulla tai otetaan rinnakkaiskäyttöön. Sommerville (2016) jakaa tällaiset ratkaisut suoraan käytettäviin valmisohjelmistoihin ja säädettäviin sovelluksiin, kuten ERP-järjestelmiin.

Sommervillen (2016) mukaan valmisohjelmistot eivät yleensä täytä kaikkia vaatimuksia, ellei niitä ole alun perin kirjoitettu tietty tuote mielessä. Käytännössä ratkaisu valitaan etsimällä “tarpeeksi hyvä” paketti yrityksen tarpeiden ja ohjelmiston ominaisuuksien välillä, ja usein joko vaatimuksia tai prosesseja joudutaan taivuttamaan valitun ulkoisen ratkaisun suuntaan. Tämä vastaa tilannetta, jossa valmis PIM- tai ostoalusta pyritään sovittamaan Rengas Turku Oy:n ostomalliin ja olemassa olevaan Django-järjestelmään.

Hyötynä on, että jos markkinoilta löytyy riittävän sopiva työkalu, käyttöönotto voi olla nopeampi kuin oman ratkaisun rakentaminen ja valmiita toiminnallisuuksia voidaan hyödyntää esimerkiksi tuotetiedon hallinnassa (PIM), hinnoittelussa ja integraatioissa. Haittapuolena on toimittajariippuvuus ja se, että valmisratkaisun perusoletukset, tietomalli ja prosessit voivat poiketa selvästi Rengas Turku Oy:n tavasta tehdä ostoa, jolloin vaatimuksista joudutaan tinkimään tai ympärille syntyy “liimaohjelmistoa” ja lisälogiikkaa (Sommerville, 2016). Lisäksi asiakas ei käytännössä ohjaa tuotteen tietoturvaa tai kehityssuuntaa, vaan on sidottu toimittajan aikatauluihin ja lisenssimalliin, mikä voi johtaa pidemmällä aikavälillä kalliisiin korjausliikkeisiin tai ratkaisun vaihtotarpeeseen (Sommerville, 2016). Tässä tapauksessa vaihtoehto C on käytännössä realistinen vain, jos markkinoilta löytyy poikkeuksellisen hyvin sopiva työkalu, joka tukee myös tässä työssä kuvattua DSS- ja PIM-ajattelua.

Vaihtoehto D – Ei teknistä ratkaisua, prosessin kehittäminen

Vaihtoehto D keskittyy ostoprosessin parantamiseen ilman uutta ohjelmistoa. Käytännössä tämä tarkoittaa esimerkiksi käytäntöjen parantamista, työn selkeyttämistä ja ohjeistuksen kehittämistä. Tällä tavoin voidaan saavuttaa nopeita hyötyjä pienillä kustannuksilla, erityisesti silloin kun nykyisiä työtapoja ei ole aiemmin kuvattu tai sovittu tarkasti.

Kirjallisuudessa korostetaan, että prosessien selkeyttäminen ennen teknisiä investointeja vähentää riskiä siihen, että heikkoja käytäntöjä otetaan osaksi uutta rakennettavaa järjestelmää (Richards & Ford, 2020; Newman, 2021). Tässä vaihtoehdossa ostoprosessi pysyy kuitenkin pääosin manuaalisena. Taulukot, sähköpostit ja hajautunut tieto säilyvät käytännössä suurin piirtein ennallaan, mikä rajoittaa ratkaisun tehokkuutta. Aiempien empiiristen havaintojen perusteella nykyinen työmäärä on jo lähellä ylärajaa, joten pelkkä prosessin pieni parantaminen ilman teknistä tukea ei todennäköisesti riitä tuomaan tarvittavaa helpotusta massatilauksiin ja hinnoittelutyöhön.

5.2 Vertailukriteerit ja painot

Vertailu tehdään painotetun summamallin periaatteella (SAW). Jokaiselle kriteerille on määritetty paino, ja vertailtavat vaihtoehdot pisteytetään kaikki samalla asteikolla yhdestä viiteen. Kriteerit on laadittu siten, että suurempi arvo kuvaa kaikissa tapauksissa parempaa suoriutumista. Kokonaistulos saadaan kertomalla kriteerin pistemäärä sen painolla ja laskemalla tulokset yhteen. Menetelmä on läpinäkyvä ja helposti toistettavissa, ja se noudattaa Tzengin ja Huangin (2011) kuvaamaa monikriteerisen päätöksenteon viitekehystä ja SAW-käytäntöä.

Kriteerit ja painot johdettiin luvun 2 teoreettisesta viitekehyksestä, luvussa 4 kuvatuista kipupisteistä ja vaatimuksista sekä yrityksen asiantuntijoiden kanssa käydystä arviointipalaverista (ks. Liite 1, “Yrityspalaverit”). Painot perustuvat erityisesti projektin aikana tehtyyn vaatimusten määrittelyyn, jossa korostuivat ratkaisun vaikuttavuus, pitkän aikavälin ylläpidettävyys ja mahdollisuus rakentaa DSS- ja PIM-ajattelua tukeva ratkaisu PK-ympäristössä. Vertailu rajautuu tässä työssä renkaiden ostoprosessiin, mutta kriteerit on muotoiltu niin, että sama ratkaisu voidaan myöhemmin ottaa käyttöön myös vanne- ja tarvikeostoihin.

Käytetyt kriteerit ja painot ovat seuraavat:

1. Ratkaisun vaikuttavuus 0,18
2. Ylläpidettävyys ja elinkaari 0,14
3. Digitalisaatio ja työkalupotentiaali 0,12
4. Kustannus (käänteinen) 0,12
5. Integraation vaivattomuus 0,08
6. Tekninen vaativuus (käänteinen) 0,08
7. Riskit ja epävarmuudet (käänteinen) 0,08
8. Käyttökokemus 0,10
9. Tiedonhallinta ja siirrettävyys 0,06
10. Skaalautuvuus 0,04

Painotus on rakennettu Rengas Turku Oy:n tunnistetun nykytilan ja resurssien pohjalta. Ratkaisun vaikuttavuus, ylläpidettävyys ja digitalisaatio-/työkalupotentiaali saivat suurimmat painot, koska yksi työn keskeisistä tavoitteista on vähentää manuaalista työtä, lyhentää työhön käytettävää aikaa ja luoda pohja ostoa tukeville työkaluille ilman, että jär-

jestelmästä tulee vaikeasti ylläpidettävä. Kustannus, tekninen vaativuus ja riskit on mitoitettu käänteisinä kriteereinä, eli matala kustannus, matala tekninen vaativuus ja matala riski saavat korkeimman pistemäärän. Tämä tehdään sen vuoksi, että kaikki kriteerit ovat samansuuntaisia ja suurempi pistemäärä tarkoittaa laskennassa aina parempaa lopputulosta. Integraation vaivattomuus ja tiedon hallinta erotettiin omiksi kriteereikseen, koska ratkaisun on oltava liitettävissä olemassa olevaan Django-järjestelmään ja samalla tuettava kanonista tuotemallia myös mahdollisia tulevia tuoteryhmiä varten. Skaalautuvuus sai muita pienemmän painon, koska yrityksen arvion mukaan lyhyellä aikavälillä kuorma on selkeästi hallittavissa, mutta kriteeri haluttiin silti mukaan, jotta valittava arkkitehtuuri ei estä laajentumista vanne- ja tarvikeostoihin.

Painot mitoitettiin niin, että niiden summa on yksi. SAW-malli olettaa, että kriteerit ovat keskenään erilaisia, jotta niiden painotettu summa kuvaa kokonaisarvoa (Tzeng & Huang, 2011). Tässä tapauksessa oletusta pidetään hyväksyttävänä, koska kriteerit kuvaavat selkeästi eri näkökulmia samaan ratkaisuun. Seuraavassa luvussa käydään läpi pisteytys kriteereittäin ja esitetään taulukossa kokonaistulos, jonka avulla arkkitehtuurivalintaa voidaan myöhemmin perustella.

5.3 Pisteytys ja tulokset

Esikarsinta

Ennen pisteytystä tehtiin esikarsinta, jonka tavoitteena oli rajata vertailu sellaisiin vaihtoehtoihin, joilla on realistinen kyky ratkaista luvussa 4 kuvatut kipupisteet. Arvioinnissa painotettiin erityisesti sitä, miten se vaikuttaa manuaalityöhön, hinnoitteluun ja massatilausten läpimenoaikoihin sekä miten helppo sillä on jatkossa rakentaa oston ympärille DSS- ja PIM-tyyppisiä ominaisuuksia myös myöhemmille vanne- ja tarvikeostoille. Esikarsinta oli alun perin tarkoitus tehdä kaikille neljälle vaihtoehdolle, mutta vaatimuksia katsottuna kävi selväksi, että osa ratkaisuista ei täyttänyt projektin kannalta olennaisia ehtoja. Tämä varmistettiin arviointipalaverissa, jossa jokainen vaihtoehto esiteltiin ja käytiin lyhyesti läpi yrityksen asiantuntijoiden kanssa (ks. Liite 1, “Yrityspalaverit”).

Arviointipalaverissa todettiin, että kaupalliset ja ulkoiset ratkaisut (vaihtoehto C) eivät istu hyvin Rengas Turku Oy:n nykyiseen toimintalinjaan. Yritys on aiemmin suosinut räätälöityjä ratkaisuja ja kokenut tämän lähestymistavan hyväksi, joten myös tässä projektissa oli ajateltu ratkaisua, jota voidaan muokata omien tarpeiden mukaan. Projektin aikana määritellyt vaatimukset myös antavat viittausta siihen, että yleiskäyttöinen kaupallinen työkalu ei todennäköisesti tuota riittävää hyötyä ostoprosessin parantamisessa pidem-

mällä tähtäimellä. Keskusteluissa korostui lisäksi, että sopivan valmisratkaisun löytäminen olisi vaikeaa ja edellyttäisi merkittäviä kompromisseja uuden ostoprosessin suhteen. Näistä syistä vaihtoehto C nähtiin sekä strategisesti että käytännössä heikosti sopivana, eikä sitä viety SAW-pisteytykseen.

Vaihtoehto D, joka perustui pelkkään prosessin kehittämiseen ilman uutta teknistä ratkaisua, karsittiin, koska se ei selkeästi poista keskeisiä ongelmia, kuten tiedon hajautumista, manuaalityötä ja virhealttiutta. Luvun 4 empiiristen havaintojen perusteella nykyinen työmäärä on jo lähellä ylärajaa, ja ilman uutta tehokasta työkalua massatilausten ja hinnoittelun läpivientiä ei realistisesti saada kevennettyä. Näin varsinainen pisteytetty vertailu rajattiin vaihtoehtoihin A (Django-laajennus) ja B (mikropalvelu), jotka täyttävät vaikuttavuuden ja arkkitehtuurisen yhteensopivuuden edellytykset ja tarjoavat pohjan myöhemmälle laajentamiselle myös vanne- ja tarvikeostoihin.

Pisteytysmenetelmä ja kaava

Pisteytys tehtiin luvussa 5.2 kuvatun painotetun summamallin (SAW) mukaisesti. Jokainen kriteeri pisteytettiin asteikolla 1–5, ja käänteiset kriteerit (kustannus, tekninen vaativuus, riskit) muunnettiin niin, että suurempi pistemäärä tarkoitti kaikissa kriteereissä parempaa suoriutumista. Vaihtoehdon kokonaistulos laskettiin kaavalla

$$S_j = \sum_{i=1}^{n} w_i x_{ij}$$

, missä w_i on kriteerin paino ja x_{ij} vaihtoehdon j saama normalisoitu pistemäärä kriteerissä i. Tutkija pisteytti vaihtoehdot, minkä jälkeen tulokset käytiin läpi ja hyväksyttiin arviointipalaverissa (ks. Liite 1, “Yrityspalaverit”).

Painot, pisteet ja laskettu kokonaistulos

Käytetyt kriteerit ja painot on esitetty luvussa 5.2. Ne perustuvat renkaiden oston nykyisiin kipupisteisiin, projektissa tehtyyn vaatimusten määrittelyyn. Varsinainen vertailudata, eli painot, kriteerikohtaiset pisteet ja painotetut pisteet, on koottu taulukkoon 1. Pisteasteikko on 1–5.

Taulukko 1: *SAW-menetelmän mukainen vertailulaskenta.*

Kriteeri	Paino	A piste	A painotettu	B piste	B painotettu
Ratkaisun vaikuttavuus	0,18	5	0,90	5	0,90
Ylläpito ja elinkaari	0,14	3	0,42	5	0,70
Digitalisaatio ja työkalut	0,12	4	0,48	5	0,60
Kustannus *	0,12	3	0,36	3	0,36
Integraation vaivattomuus	0,08	4	0,32	3	0,24
Tekninen vaativuus *	0,08	4	0,32	3	0,24
Riskit ja epävarmuudet *	0,08	4	0,32	3	0,24
Käyttökokemus	0,10	4	0,40	5	0,50
Tiedonhallinta ja siirrettävyys	0,06	3	0,18	4	0,24
Skaalautuvuus	0,04	3	0,12	5	0,20
Summa:	**1,00**	**37**	**3,82**	**41**	**4,22**

* Käänteinen kriteeri

Taulukon 1 tulosten perusteella **vaihtoehto B** eli **mikropalvelu** saa **4,22** pistettä ja **vaihtoehto A** eli **Django-laajennus 3,82** pistettä. Ero syntyy erityisesti ylläpidettävyydessä, elinkaaressa sekä digitalisaatio- ja työkalupotentiaalissa. Mikropalvelu hyötyy omasta tietovarastosta, irtikytkennästä, joustavammista teknologiavalinnoista ja itsenäisestä kehitysrytmistä (vrt. luku 2.5). Mikropalvelu saa paremmat pisteet myös skaalautuvuudessa ja käyttökokemuksessa, koska rakenne mahdollistaa paremmin pitkän aikavälin laajennusta ja antaa enemmän liikkumavaraa käyttöliittymäratkaisuihin.

Django-laajennus suoriutuu odotetusti paremmin integraation vaivattomuudessa ja teknisen vaativuuden käänteisessä kriteerissä. Se hyödyntää nykyistä koodia, järjestelmää ja osaamista, mikä laskee aloituskynnystä ja tekee ensimmäisestä versiosta edullisemman toteuttaa. Kustannus arvioitiin kuitenkin molemmille vaihtoehdoille samalle tasolle, koska kriteeri kattaa myös pidemmän aikavälin näkökulman. Mikropalvelu voi vähentää legacy-riippuvuutta, helpottaa tämän kaltaisen rajatun osa-alueen ylläpitoa ja pienentää myöhempien muutosten kustannuksia verrattuna laajenevaan monoliittiin (vrt. luku 2.5). Lopputulos ei siis synny kustannuserosta, vaan siitä, että painotetuissa pitkän aikavälin kriteereissä (ylläpidettävyys, työkalupotentiaali, skaalautuvuus, tiedon hallinta) mikropalvelu saa selvästi paremmat pisteet luvussa 5.2 määritellyn painorakenteen mukaan.

Herkkyysanalyysi

Kirjallisuudessa suositellaan, että monikriteerisen päätöksenteon yhteydessä tarkastellaan myös sitä, miten kriteeripainojen muutokset vaikuttavat lopputulokseen (Tzeng &

Huang, 2011). Tässä työssä herkkyysanalyysi tehtiin taulukkolaskentaohjelmassa muuttamalla painoja useilla eri tavoilla ja pitämällä painot aina niin, että niiden summa pysyi vakiona. Yksittäisiä painoja siirrettiin tyypillisesti noin ±20 % kerrallaan, minkä jälkeen tarkasteltiin, miten muutos vaikutti kokonaistulokseen. Lisäksi testattiin tilannetta, jossa vaihtoehtoa A suosivien kriteerien (esimerkiksi integraation vaivattomuus, tekninen vaativuus ja riskit) painoja nostettiin samanaikaisesti ja vaihtoehtoa B suosivien kriteerien (ylläpidettävyys, digitalisaatio ja työkalut, tiedon hallinta ja skaalautuvuus) painoja laskettiin.

Analyysin tulokset pysyivät vakaana. Kaikissa realistisiksi arvioiduissa muutoksissa mikropalvelu säilyi pisteiden perusteella parempana vaihtoehtona. Järjestys ei kääntynyt edes tapauksessa, jossa painoja muutettiin selvästi A:ta suosivaan suuntaan, vaan mikropalvelu jäi edelleen niukasti edelle. Herkkyysanalyysin perusteella mikropalveluun päätyvä valinta ei ole herkkä pienemmille painomuutoksille, mikä tukee tämän pisteytyksen tuloksen luotettavuutta.

5.4 Valinta ja johtopäätökset

Valituksi ratkaisuksi valittiin **vaihtoehto B**, **erillinen mikropalvelu**. Painotetun summamallin pisteytys osoitti mikropalvelulle korkeammat kokonaispisteet kuin Django-laajennukselle, ja luvussa 5.3 tehty herkkyysanalyysi vahvisti, että järjestys säilyy kaikissa realistisiksi arvioiduissa painomuutoksissa (Tzeng & Huang, 2011). SAW-pisteytys toimi kuitenkin ennen kaikkea päätöksenteon tukena. Se teki vertailusta läpinäkyvän ja auttoi hahmottamaan, missä kriteereissä erot syntyvät, mutta lopullinen valinta tehtiin todellisuudessa pistetuloksen, vaatimusten ja asiantuntijaharkinnan yhdistelmänä.

Valinta pohjautuu siihen, miten hyvin vaihtoehdot vastaavat luvussa 4.5 koottuja vaatimuksia. Mikropalvelu mahdollistaa laajemman työkalujen käytön, joka tukee massapäivityksiä, ostohinnan laskentaa, kanonisen mallin toteuttamista sekä myöhempää DSS-tyyppisten ominaisuuksien rakentamista. Se erottaa oston ja hinnoittelun logiikan omaksi kokonaisuudekseen, jolloin ylläpito ja jatkokehitys on helpompi rajata selkeämmin kuin ajan mittaan paisuvassa monoliitissa. Tämä menee samaan suuntaan myös ajatuksen kanssa uudesta koodipohjasta ja pitkästä elinkaaresta, jossa arkkitehtuuri ei rajoita myöhempää laajentamista vanne- ja tarvikeostoihin (ks. Liite 1, “Yrityspalaverit”).

Arviointipalaverissa pisteytyksen tulokset, kriteerien painot ja käytännön riskit käytiin yhdessä läpi. Palaverin muistion mukaan mikropalvelu katsottiin parhaana ratkaisuna, koska se tukee yrityksen strategista suuntaa, tarjoaa paremman perustan ostajaa tukevien työkalujen kehittämiselle ja mahdollistaa käyttöliittymäkehityksen laajemmin kuin

monoliitin sisälle rakennettu moduuli (ks. Liite 1, “Yrityspalaverit”). Django-laajennus arvioitiin silti erittäin käyttökelpoiseksi etenemistavaksi, ja se on määritelty varasuunnitelmaksi tilanteisiin, joissa mikropalvelun toteutuksen riskit tai resurssirajoitteet osoittautuvat arvioitua suuremmiksi.

Esikarsinnassa pois rajatut vaihtoehdot C ja D eivät täyttäneet osion 4.5 määritettyjä vaatimuksia. Kaupallinen tai ulkoinen työkalu ei todennäköisesti tarjoa riittävää räätälöitävyyttä renkaiden ostoon, eikä prosessin kehittäminen ilman uutta järjestelmää poista keskeisiä ongelmia, kuten manuaalityötä ja tiedon hajautumista. Kokonaisuutena valinta perustuu luvussa 5 esitettyyn pistevertailuun, herkkyystarkasteluun ja yrityksen asiantuntijoiden kanssa tehtyyn harkintaan. Menettely on läpinäkyvä ja linjassa Tzengin ja Huangin (2011) kuvaaman monikriteerisen päätöksenteon perusperiaatteiden kanssa.

5.5 Riskit ja varasuunnitelma

Mikropalveluratkaisussa on käytännössä kahta päätyyppiä riskejä: arjen tekemiseen liittyvät riskit ostoprosessissa ja tekniset riskit, jotka liittyvät uuden ratkaisun rakentamiseen ja ylläpitoon.

Toimittajien rajapintojen kohdalla riski on, että aineisto on puutteellista, virheellistä tai väärässä muodossa. Jos virheitä ei huomata ajoissa, järjestelmään voi päätyä vääriä tuotteita tai hintoja, mikä näkyy suoraan myynnissä ja sesonkitilauksissa. Siksi mikropalvelun on tarkistettava aineisto ennen kuin se viedään eteenpäin ja merkittävä virheelliset rivit erikseen tarkastettaviksi. Käytännössä tämä tarkoittaa virhelokia tai erillistä näkymää, josta ostosta vastaava työntekijä voi käydä toimittajakohtaiset virheet läpi ennen erien hyväksymistä. Prosessiin kuuluu lisäksi muutenkin se, että tuotteet käydään yksitellen läpi ennen lopullista hyväksyntää, joten tällaiset virheet pitäisi saada kiinni jo ennen kuin ne pääsevät vaikuttamaan käytännön työhön.

GTIN-koodeihin liittyvä riski syntyy siitä, että kaikilla tuotteilla ei ole vielä Nettivanteessa GTIN-tunnistetta tai toimittajilla on samoista tuotteista eri koodeja. Pahimmillaan väärä GTIN yhdistää renkaan väärään tuotekorttiin tai jättää tuotteen kokonaan kanonisen tuotemallin ulkopuolelle. Seurauksena järjestelmään päätyy virheellistä tietoa, mikä voi aiheuttaa uusia virheitä myöhemmissä vaiheissa. Riskin hallinta edellyttää, että mikropalvelu tukee useita tunnistetapoja rinnakkain, tarjoaa työkalut GTIN-tietojen täydentämiseen ja mahdollistaa epäselvien tapausten merkitsemisen manuaalista tarkistusta varten.

Varastoon liittyvä riski kytkeytyy siihen, että virheelliset varastosaldot Nettivanne-järjestelmässä heijastuvat sellaisenaan mikropalveluun. Mikropalvelu käyttää varastosaldoja

taustatietona ostopäätöksiin, joten ero fyysisen varaston ja järjestelmän välillä voi ohjata oston väärään suuntaan. Tämän vuoksi ratkaisu ei korvaa varaston perusprosesseja, vaan toimii niiden päällä. Varaston oikeellisuus varmistetaan edelleen nykyisillä käytännöillä, ja mikropalvelu hyödyntää näitä tietoja päätöksenteon tukena.

Hintalogiikassa riski liittyy enemmän vakauteen ja läpinäkyvyyteen kuin yksittäiseen bugiin. Mikropalvelu ottaa huomioon hinnoittelussa ostohinnan, kampanjat ja muut ehdot ja muodostaa niiden pohjalta summan tai hintasuosituksen. Jos hinnoittelutapaa tai kaavaa muutetaan usein ja ilman kunnollista dokumentointia, voi syntyä tilanne, jossa järjestelmän toimintaa on vaikea ymmärtää tai selittää. Tätä hallitaan etenemällä vaiheittain: ensimmäinen hinnoittelumalli pidetään tietoisesti yksinkertaisena ja myöhempi DSS-tyyppinen kehitys tehdään iteroiden. Hinnat käydään joka tapauksessa manuaalisesti läpi tuotekohtaisesti ostovaiheessa, joten ostosta vastaava ammattilainen voi reagoida nopeasti, jos hinnoittelukaavat tai ehdotukset eivät ole järkeviä.

Teknisesti suurimmat riskit liittyvät siihen, että mikropalvelu on Rengas Turku Oy:lle ensimmäinen tuotantokäytössä oleva erillinen palvelu. Vaikka talossa on kokemusta konttiympäristöistä ja CI/CD-putkista, arjen työ tehdään tällä hetkellä Django-monoliitin ympärillä. Mikropalvelu tarvitsee toimivan julkaisuketjun, valvonnan ja lokituksen, jotta häiriöt eivät jää piiloon (Newman, 2021). Uusi pino vaatii myös opettelua, vaikka perusosaaminen teknologiasta on olemassa. Osaamisriskiä pienennetään sillä, että mikropalvelu toteutetaan myös Pythonilla ja pääasiassa samoilla periaatteilla kuin nykyinen järjestelmä.

Varasuunnitelmana on siirtyä vaihtoehtoon A eli Django-laajennukseen, jos mikropalvelun toteutus ei etene riittävän hyvin. Käytännön rajaksi on ajateltu tilannetta, jossa mikropalvelua ei saada järkevästi alkua pidemmälle teknisessä toteutuksessa tai mikropalvelun integroinnista aiheutuu merkittäviä ongelmia. Tällöin on perusteltua katkaista kehityslinja ja siirtyä varasuunnitelmaan.

Varasuunnitelma ei tarkoita paluuta suoraan alkuun, vaan teknisen toteutustavan vaihtamista. Mikropalveluun suunniteltuja tietomalleja, kanonista tuotemallia, toimittaja-adaptereita ja rajapintaperiaatteita voidaan käyttää samalla tavalla myös Django-laajennuksessa. Suurin osa logiikasta säilyy, vaikka ratkaisu siirretään takaisin olemassa olevaan monoliittiin. Käyttäjän näkökulmasta tavoite on, että käyttöliittymän ja prosessin tavoitetila pysyvät samoina, vaikka taustalla oleva arkkitehtuuri vaihtuisi.

Yhteenvetona mikropalveluvalinta tuo mukaan lisäriskejä integraatioon, käyttöönottoon ja osaamiseen, mutta niitä voidaan hallita tarkistuksilla, vaiheittaisella etenemisellä ja etukäteen sovitulla varasuunnitelmalla.

6. VALITUN RATKAISUN SUUNNITTELU JA TESTAUS

6.1 Tavoitteet, rajaus ja oletukset

Seuraavaksi tutkija hahmottelee ratkaisun, jonka avulla Rengas Turku Oy:n ostoprosessi voidaan digitalisoida luvussa neljä tunnistettujen kipupisteiden ja luvussa viisi määriteltyjen vaatimusten pohjalta. Taustalla on erityisesti tarve vähentää sesonkitaulukoiden manuaalista käsittelyä, tukea ostohinnan laskentaa paremmin ja tehdä ostopäätöksistä läpinäkyvämpiä siten, että ostaja pitää päätösvallan. Ratkaisu rakentuu luvussa 5.3 valitun arkkitehtuurivaihtoehdon varaan, jossa ostoprosessin pohjaksi toteutetaan erillinen mikropalvelu nykyisen Django-pohjaisen ydinjärjestelmän rinnalle.

Ratkaisun keskipiste on ostotyökalu, joka kokoaa toimittajilta tulevan tuotetiedon, auttaa hinnoittelussa ja valinnoissa ja kokoaa varastotiedot yhteen kanoniseen tuotemalliin. Tavoitteena ei ole automatisoida ostopäätöksiä täysin, vaan vähentää mekaanista työtä ja virheriskiä niin, että ostaja voi keskittyä nimenomaan ostamiseen liittyvään työhön. Käytännössä tämä tarkoittaa sitä, että massadata saadaan sisään rajapintojen ja sanomien kautta, hinnoittelua voidaan tarkastella TCO-tyyppisesti ja päätöksistä jää järjestelmään jälki.

Tässä luvussa tutkija rajaa suunnittelun renkaisiin, koska empiirinen tarkastelu ja vaatimusten määrittely on tehty nimenomaan rengasvalikoiman, sesonkitaulukoiden ja niiden hinnoittelun ympärille. Arkkitehtuurin ja kanonisen tuotemallin osalta tavoitteena on kuitenkin se, että ratkaisu ei jää pitkällä tähtäimellä pelkkiin renkaisiin, vaan toimii pohjana myös vanteiden ja tarvikkeiden lisäämiselle. Tämä näkyy erityisesti kanonisen mallin ja integraatiokerroksen suunnittelussa sekä siinä, miten toimittajakohtaiset adapterit liitetään yhteiseen malliin.

Suunnittelu pohjautuu muutamaan perusoletukseen, jotka on syytä käydä läpi. Ensimmäinen oletus on, että yritys jatkaa nykyisen Django-pohjaisen ydinjärjestelmän käyttöä ja että uusi ostotyökalu liitetään siihen rajapintojen ja tietokannan kautta sen sijaan, että koko järjestelmä kirjoitettaisiin uudelleen. Toinen käytännön oletus liittyy toimittajiin: suurimmalla osalla rengastoimittajista on mahdollista toimittaa vähintään PRICAT- ja INVRPT-tyyppinen massadata tai sitä vastaava aineisto, joka voidaan kohdistaa kanoniseen tuotemalliin toimittajakohtaisilla adaptereilla. Kolmantena oletetaan, että ostoprosessi säilyy käynnissä koko kehitystyön ajan, joten ratkaisu on tehtävä niin, että käyttöönotto voidaan tehdä vaiheittain ilman yhtä isoa katkoa.

Lisäksi oletetaan, että ostopäätökset perustuvat myös tulevaisuudessa asiantuntijan harkintaan, eikä sääntömoottori tai suositukset korvaa ostajaa, vaan tukevat hänen työtään. Tämä vaikuttaa sekä käyttöliittymän suunnitteluun että siihen, miten sääntömoottorin ja integraatioiden tuottama tieto esitetään. Näiden tavoitteiden ja rajausten pohjalta seuraavat alaluvut käyvät läpi arkkitehtuuria, tietomallia, integraatioita ja käyttöönoton etenemistä niin pitkälle kuin tässä työssä on tarkoituksenmukaista mennä.

6.2 Kokonaisarkkitehtuuri ja integraatioperiaatteet

Tässä alaluvussa kuvataan, miten valittu ratkaisu rakennetaan suhteessa nykyiseen Nettivanne-järjestelmään ja toimittajaintegraatioihin. Tarkoitus on hahmottaa, missä kohdin mikropalvelu tulee mukaan, mitä nykyinen järjestelmä edelleen hoitaa ja millä periaatteilla toimittajilta tuleva data yhdistetään kanoniseen tuotemalliin.

Kokonaisarkkitehtuurin lähtökohta on, että nykyinen Django-pohjainen järjestelmä jatkaa myynnin, varaston ja arjen operatiivisen työn ytimenä, eikä sitä kirjoiteta uudelleen. Uusi mikropalvelu toteutetaan sen rinnalle omana palvelunaan. Mikropalvelu vastaa toimittajaintegraatioista ja ostotyökalun tarvitsemasta tuotetiedosta, kun taas Nettivanne käyttää tätä tietoa myynnin ja varastoinnin näkökulmasta. Tässä työssä tutkija ja yrityksen ammattilaiset valitsivat, että tuotetiedon “single source of truth” sijoitetaan mikropalvelun hallinnoimaan tietomalliin, jolloin yksi keskitetty tietolähde ohjaa muuta ympäröivää järjestelmää (Richards & Ford, 2020). Samalla nykyinen modulaarinen monoliitti säilyy yrityksen ydinjärjestelmänä, jonka rinnalle uusia palveluja voidaan lisätä vaiheittain, mikä vastaa Newmanın kuvaamaa asteittaista siirtymää monoliitista mikropalveluihin (Newman, 2021).

Toimittajilta tuleva tieto kulkee mikropalvelun kautta. Luvussa 4.4 kuvattujen havaintojen perusteella suurilla rengastoimittajilla on tyypillisesti käytössä GS1 EANCOM -pohjaiset PRICAT- ja INVRPT-tyyppiset massasanomat tai niitä vastaava rakenne hinnastoille ja varastosaldoille sekä erillinen kanava yksittäisille saatavuus- ja tilauskyselyille (GS1, 2016/2021a; GS1, 2016/2021b; Continental, n.d.; EDIWheel, n.d.). Näiden perusteella oletetaan, että toimittajat pystyvät toimittamaan ainakin hinnasto- ja varastotiedot eräajona sekä tarjoamaan jonkin rajapinnan yksittäisille tapahtumille, vaikka tarkka formaatti ja teknologia voivat vaihdella toimittajittain. Ratkaisua ei siksi sidota yhteen toimintatapaan, vaan jokaisen toimittajan integraatiot toteutetaan erikseen, ja niille suunnitellaan adapterit, joiden tehtävänä on muotoilla toimittajakohtainen tuotteisto yhteiseen rakenteeseen.

Integraatioperiaatteet voidaan tiivistää kahteen toisiaan tukevaan virtaan. Ensimmäinen on massadataan perustuva push-virta, jossa toimittaja lähettää hinnasto- ja varastotiedot säännöllisesti sovitussa muodossa. Nämä sanomat tai tiedostot vastaanotetaan mikropalveluun, tarkistetaan ja muutetaan adapterin avulla yhteiseen muotoon. Toinen virta on transaktioihin perustuva pull-virta, jossa mikropalvelu kutsuu toimittajan tarjoamaa rajapintaa yksittäisiin saatavuus- ja hintakyselyihin sekä tilausten välittämiseen, tyypillisesti REST-tyyppisten HTTP-rajapintojen kautta (Fielding, 2000; Continental, n.d.). Massadata pitää tuotemassan perusrakenteen ja varastotilanteen ajan tasalla, ja transaktiokanava hoitaa yksittäiset reaaliaikaiset tapahtumat.

Massadatan käsittelyssä noudatetaan periaatetta, jossa sama erä käsitellään vain kerran. Toimittajan lähettämä aineisto luetaan taustaprosesseissa, tarkistetaan ja muotoillaan yhteiseen rakenteeseen. Käsittelyssä hyödynnetään idempotenttia vastaanottajaa ja korrelaatiotunnisteita, jotta sama erä ei pääse vahingossa päivittämään tietoja useampaan kertaan ja jotta tapahtumaketju voidaan jäljittää tarvittaessa (Hohpe & Woolf, 2004). Yksityiskohtaisempi kuvaus taustaprosesseista, tehtäväjonoista ja suorituskykyyn liittyvistä valinnoista esitetään luvussa 6.4.

Kanoninen tuotemalli sijaitsee mikropalvelun omassa tietokannassa ja toimii tuotetiedon määrityksenä. Tässä alaluvussa riittää todeta, että eri toimittajien adapterit liittävät eri lähteistä tulevan datan tähän yhteiseen rakenteeseen, jonka päälle ostotyökalun logiikka ja Nettivanteen tarvitsema tuotetieto rakentuvat. Kanonisen mallin käyttö ja sen suhde rengas-, vanne- ja tarvikeattribuutteihin käsitellään luvussa 6.3.

Arkkitehtuuri suunnitellaan niin, että se tukee useita toimittajia ja myöhemmin myös muita tuoteryhmiä ilman, että ydinratkaisua täytyy rakentaa uudelleen. Toimittajakohtaiset erot pidetään mahdollisimman pitkälti integraatioissa ja adaptereissa, jolloin yhteistä rakennetta voidaan hallita. Nettivanne pystyy jatkamaan väliaikaisesti toimintaansa nykyisellä toimintatavalla myös ongelmatilanteissa, jos mikropalvelu ei ole tilapäisesti käytettävissä. Näin riski yhdestä isosta käyttökatkosta pysyy hallittuna ja ratkaisu voidaan ottaa käyttöön vaiheittain.

6.3 Kanoninen tuotemalli ja tiedonhallinta

Tässä alaluvussa kuvataan, mitä kanonisella tuotemallilla tarkoitetaan tässä työssä ja miten se tukee ostoprosessin digitalisointia. Painopiste on työn rajauksen mukaan rengastuotteissa, mutta rakenne suunnitellaan niin, että samaa periaatetta voidaan käyttää myöhemmin myös vanteiden ja tarvikkeiden kanssa.

Kanoninen tuotemalli toimii mikropalvelun tietokannassa tuotetiedon yhteisenä runkona. Tarkoituksena on, että toimittajilta tuleva tieto sovitetaan ensin tähän malliin, ja vasta sen jälkeen sitä käytetään ostotyökalussa ja viedään Nettivanne-järjestelmän tarvitsemiin rakenteisiin. Tämä vastaa tuotetiedon hallinnan (PIM) perusajatusta, jossa tuotetieto kootaan yhteen paikkaan ja jaellaan sieltä eri kanaviin yhtenäisessä muodossa (Abraham, 2014). Käytännössä mikropalvelun kanoninen malli toimii masterina, josta Nettivanne lukee tarvitut tiedot myyntiä varten.

Abraham (2014) kuvaa PIM-järjestelmän “master catalog” -rakenteen siten, että jokainen tuote esiintyy luettelossa kerran ja siihen liitetään joukko erilaisia ominasuuksia. Osa ominaisuuksista koskee kaikkia tuotteita, kuten tuotetunniste, nimi ja kuvaus, kun taas osa on vain tiettyjen tuoteryhmien tarpeisiin (Abraham, 2014). Näiden periaatteiden pohjalta tässä työssä tutkija suunnitteli kanonisen tuotemallin niin, että siinä on yhteinen "pohjatuote" (“Product” -runko) ja sen päälle tuoteryhmäkohtaiset laajennukset kuten renkaat, vanteet ja tarvikkeet.

Ytimen eli pohjatuotteen kenttiin kuuluvat vähintään mikropalvelun oma tuote-ID, toimittajakohtaiset tuotekoodit, yrityksen oma mahdollinen tuotekoodi, tuotteen nimi, merkki, tuoteryhmä, verokanta ja perushinnat. Lisäksi malliin varataan paikka kansainväliselle tuotetunnisteelle (GTIN), jonka avulla tuotteet voidaan yksilöidä ja yhdistää eri lähteistä tuleva tieto samaan tuotteeseen. GS1:n mukaan GTIN on ensisijainen tunniste kauppatavaroille, joita voidaan hinnoitella, tilata tai laskuttaa missä tahansa toimitusketjun kohdassa (GS1, 2016/2021a). Tässä työssä oletetaan, että etenkin suurempien toimittajien kohdalla GTIN tai muu GS1-yhteensopiva tunniste on usein saatavilla, mutta malli ei perustu pelkästään siihen.

Rengastuotteille määritellään kanoniseen malliin oma tuoteryhmäkohtainen osa, joka rakentuu aikaisemmin mainitun pohjatuotteen päälle. Rengasmallissa on vähintään tiedot koosta (leveys, profiili, halkaisija), kantavuusluokasta ja nopeusluokasta sekä rengastyypistä ja kulkuneuvotyypistä (esimerkiksi henkilöauto, pakettiauto, SUV-tyyppi, moottoripyörä, trukki). Lisäksi malliin varataan kentät rengasmerkinnöille, kuten märkäpito, polttoainetaloudellisuus ja vierintämelu, jotka perustuvat EU:n rengasmerkintöihin ja joita toimittajat tyypillisesti toimittavat tuotetietonsa mukana. Malli tukee myös Nettivanteessa käytössä olevaa “sarja” -ajattelua, jossa etu- ja takarenkaat yhdistetään samaan kokonaisuuteen. Käytännössä tämä tarkoittaa, että kanoniseen malliin lisätään mahdollisuus kuvata tuotteiden välisiä suhteita sen sijaan, että sarjoja ylläpidettäisiin vain Nettivanteen omassa logiikassa.

Vanteet ja tarvikkeet eivät ole tämän työn empiirisen tarkastelun kohteena, mutta ne huomioidaan rakenteessa. Ensimmäisessä vaiheessa kanoniseen malliin otetaan käyttöön rengastuotteiden osuus. Myöhemmin vanteille ja tarvikkeille voidaan suunnitella omat laajennuksensa, jotka käyttävät samaa perusrakennetta, mutta sisältävät esimerkiksi pulttijaon, ET-arvon ja halkaisijan vanteille sekä yksinkertaisempia attribuutteja tarvikkeille. Näin malli ei jää ”rengasspesifiksi” ratkaisuksi, vaan tarjoaa rungon, johon muita tuoteryhmiä voidaan lisätä ilman, että ydintä täytyy purkaa.

Tunnisteiden ja tuotteiden yhdistämisen kannalta malli käyttää useampaa rinnakkaista tunnistetta. Mikropalvelun oma tuote-ID on aina pakollinen ja sitä käytetään sisäisissä viittauksissa ja tuotteiden välisissä suhteissa. GTIN toimii vahvimpana ulkoisena tunnisteena silloin, kun se on saatavilla, koska sen avulla sama kauppatavara voidaan yhdistää luotettavasti eri toimittajien ja kanavien välillä (GS1, 2016/2021a). Sen rinnalla säilytetään toimittajan oma tuotekoodi ja tarvittaessa yhdistelmä teknisiä attribuutteja, kuten koko, kantavuusluokka ja nopeusluokka. Tuotteiden yhdistäminen suunniteltiin vaiheittaiseksi: ensin pyritään yhdistämään tuotteet GTINin perusteella, sen jälkeen toimittajan koodin ja muiden tunnisteiden avulla ja viimeisenä keinona attribuuttien yhdistelmiä hyödyntämällä. Jos tulos jää epäselväksi, tuote ohjataan erilliseen tarkastusjonoon, jossa ostaja tai pääkäyttäjä voi tarkistaa yhdistämisen käsin. Epäselvät tilanteet joudutaan käsittelemään manuaalisesti.

Toimittajilta tuleva data validoidaan ennen kuin sitä tallennetaan kanoniseen malliin. Massadatan käsittelyvaiheessa adapteri tarkistaa, että pakolliset kentät, kuten koko, toimittajan koodi ja mahdollinen GTIN, ovat mukana ja loogisessa muodossa. Virheelliset tai puutteelliset rivit ohjataan erilliseen tarkasteluun sen sijaan, että ne vietäisiin suoraan malliin. Wang ja Strong (1996) korostavat, että tiedon kontekstinmukaiseen laatuun kuuluu muun muassa se, että tietoa on “sopiva määrä” suhteessa käyttötarpeeseen. Tässä työssä se tarkoittaa sitä, ettei kanoniseen malliin kerätä kaikkia toimittajien mahdollisesti tarjoamia attribuutteja, vaan mukaan otetaan ne, jotka ovat olennaisia ostolle, hinnoittelulle, varastonhallinnalle ja myöhemmin suunniteltaville DSS-näkymille. Mahdollisia lisäkenttiä voidaan tarvittaessa tallentaa erilliseen “raakadata” -rakenteeseen myöhempää käyttöä varten.

Nettivanteen nykyinen tuotemalli voidaan ajatella kanonisen mallin käytännöllisenä kevyeempänä versiona. Se ei ylläpidä enää omaa täydellistä tuotetietoa, vaan lukee mikropalvelusta ne kentät, joita tarvitaan myynnissä, varastossa ja asiakasrajapinnoissa. Kanoninen malli puolestaan huolehtii siitä, että hinnan eri komponentit ja tuotteen perustiedot pysyvät yhtenäisinä toimittajasta riippumatta. Nettivanne voi edelleen ylläpitää

omia paikallisia kenttiään, mutta sen ei tarvitse kantaa vastuuta koko tuotetiedon historiasta ja yhdistämisestä.

Kokonaisuutena kanoninen tuotemalli toimii sekä integraatioiden että päätöksenteon tuen kannalta perusrakenteena. Kun samat tuotteet on kuvattu samalla tavalla riippumatta toimittajasta, niiden vertaileminen, hinnoittelun arviointi ja valikoiman suunnittelu helpottuvat. Tässä alaluvussa on kuvattu rakenteen periaatteet. Seuraavat alaluvut rakentavat tämän päälle rajapintojen, taustaprosessien ja käyttöliittymän karkean suunnittelun.

6.4 Rajapinnat, taustaprosessit ja suorituskyky

Tässä alaluvussa kuvataan lyhyesti, millaisia rajapintoja ostotyökalun mikropalvelussa käytetään, miten taustaprosessit toteutetaan ja millaisia suorituskykyyn liittyviä tavoitteita ratkaisulle asetetaan. Tarkoitus ei ole määritellä yksityiskohtaista teknistä toteutusta, vaan kuvata periaatteet, joilla mikropalvelu liitetään nykyiseen Nettivanne-järjestelmään ja toimittajiin, sekä miten eräajot ja raskaammat laskennat hoidetaan niin, etteivät ne häiritse ostajan työtä.

Sisäiset rajapinnat ja mikropalvelun käyttö

Mikropalvelu tarjoaa sisäisen rajapinnan, jota käyttävät erityisesti ostotyökalun käyttöliittymä ja Nettivanne-järjestelmä. Rajapinta toteutetaan HTTP-pohjaisena ja noudattaa luvussa 2 esiteltyjä REST-tyyppisen arkkitehtuurin periaatteita. Fieldingin (2000) kuvaamassa REST-tyylissä keskeistä on yhtenäinen rajapinta ja resurssipohjaiset pyynnöt, joissa kaikki tarvittava tieto kulkee viestissä mukana ilman palvelimen omaa istuntotilaa. Tämä sopii mikropalveluun, jonka tehtävä on tarjota selkeitä ja ennakoitavia rajapintakutsuja eri asiakkaille.

Ostotyökalun suunniteltu React-käyttöliittymä käyttää mikropalvelua suoraan. Rajapinnan perusrakenne pidetään tarkoituksella yksinkertaisena. Tuotteita haetaan esimerkiksi polkujen tyyliin `/api/products/` ja yksittäisiä tuotteita polulla `/api/products/{id}/`. Lisäksi mikropalvelu tarjoaa polkuja, joissa yhdistetään toimittajien, varaston ja hinnoittelun tietoja ostajaa varten, kuten vapaana esimerkkinä `/api/offers/`, jossa tarjoustuotteet voisivat olla koottuna ja rikastettuna ostotyökalun tarpeisiin. Näiden tarkempi sisältö pohjautuu kanoniseen tuotemalliin, mutta rajapinnan kannalta olennaista on, että käyttöliittymä saa yhdellä pyynnöllä riittävän aineiston päätöksenteon tueksi eikä joudu tekemään useita peräkkäisiä hakuja jokaisesta tuotteesta.

Nettivanne-järjestelmä käyttää samaa mikropalvelua eri näkökulmasta. Sille tarjotaan rajapintoja, joilla se voi lukea myynnin ja varaston kannalta oleelliset kentät, kuten voimassa olevat hinnat ja tuoterakenteet, ja tarvittaessa päivittää omia paikallisia tietojaan. Yhteys voidaan toteuttaa jatkossa joko HTTP-rajapinnan tai suoran tietokantayhteyden kautta, mutta periaatteena on, että tuotetiedon ydin säilyy mikropalvelussa ja Nettivanne käyttää sitä omien toimintojensa taustalla. Näin luvussa 6.2 kuvattu kanoninen tuotemalli ja modulaarinen monoliitti liittyvät toisiinsa selkeän sisäisen rajapinnan kautta, eikä rinnakkaisia tuotemalleja tarvitse ylläpitää.

Toimittajiin päin käytettävät rajapinnat on käsitelty yksityiskohtaisemmin luvussa 4.4 ja luvun 6.2 integraatioperiaatteissa, joten tässä riittää rajata ne osaksi kokonaisuutta. Kuten aiemmin todettiin, suurilla rengastoimittajilla on tyypillisesti käytössä PRICAT- ja INVRPT-tyyppiset massasanomat tai vastaavat hinnasto- ja varastorakenteet sekä erillinen kanava yksittäisiin saatavuus- ja tilauskyselyihin (GS1, 2016/2021a; GS1, 2016/2021b; Continental, n.d.; EDIWheel, n.d.). Mikropalvelu toimii näiden suhteen keskipisteenä: se vastaanottaa eräajot, käsittelee ne taustaprosesseissa ja tarjoaa lopuksi tuloksen yhtenäisenä rajapintana ostotyökalulle ja Nettivanteelle. Yksittäiset pull-tyyppiset saatavuus- ja hintakyselyt tehdään edelleen toimittajan rajapintoihin, mutta ne kulkevat mikropalvelun kautta, jotta lokitus, virheiden käsittely ja mahdollinen välimuistitus pysyvät yhdenmukaisina.

Taustaprosessit ja eräajot

Taustaprosessien tehtävä on hoitaa raskaammat työt, kuten eräajot, tuotepäivitykset ja hinnoittelun uudelleenlaskennat, ilman että ostotyökalun käyttöliittymä tai Nettivanteen normaali käyttö hidastuu. Hohpe ja Woolf (2004) kuvaavat viestijonoihin perustuvien integraatioratkaisujen etuna sen, että lähettäjä ja vastaanottaja voidaan irrottaa toisistaan ja viestit voidaan käsitellä omassa tahdissaan jonon kautta. Tässä ratkaisussa sama perusajatus toteutetaan niin, että toimittajilta tulevat eräaineistot jonotetaan käsittelyyn, ja varsinainen validointi ja tietokantapäivitykset tehdään taustalla ilman, että käyttäjän tarvitsee odottaa koko käsittelyketjua valmiiksi.

Tyypillisiä taustaprosesseja ovat toimittajien hinnasto- ja varastoerien käsittely ja tuonti, massahinnoittelun laskelmat, mikropalvelun ja Nettivanteen välinen laajempi synkronointi sekä myöhemmin mahdollisten suosittelumallien ajot. Esimerkiksi PRICAT-tyyppinen hinnasto voidaan vastaanottaa tiedostona tai viestinä, jonottaa käsittelyyn ja käydä läpi rivi kerrallaan siten, että jokainen tuoterivi validoidaan ja muunnetaan kanoniseen

malliin toimittajakohtaisen adapterin avulla. Jos aineistossa on rivejä, joita ei pystytä suoraan kartoittamaan tai joissa on puutteellisia kenttiä, kyseiset rivit siirretään erilliseen tarkastusjonoon sen sijaan, että koko eräajo keskeytyisi tai muu ajo menisi hukkaan.

Taustaprosesseissa noudatetaan samoja periaatteita, joita Hohpe ja Woolf (2004) kuvaavat viestipohjaisissa integraatioissa idempotentin vastaanottajan ja virhekanavien yhteydessä. Käytännössä jokaiselle erälle muodostetaan tunniste, jonka avulla voidaan varmistaa, että sama erä käsitellään vain kerran ja että käsittelyn tulokset voidaan jäljittää myöhemmin. Jos käsittely epäonnistuu, prosessia voidaan tapauskohtaisesti yrittää automaattisesti uudelleen. Vasta toistuvasti epäonnistuvat rivit ohjataan manuaaliseen tarkasteluun, jotta ostajan aikaa ei kuluteta yksittäisten teknisten virheiden selvittelyyn.

Taustaprosessien ajo perustuu tässä työssä pääasiassa ennalta suunniteltuihin ajankohtiin. Toimittajien massadata ajetaan aina, kun toimittaja sitä tarjoaa, ja ajot voidaan ajoittaa esimerkiksi yöaikoihin tai muuhun hiljaiseen aikaan. Mikropalvelun ja Nettivanteen välinen synkronointi voidaan tehdä useammin, esimerkiksi kerran tunnissa tai aina, kun merkittävä osa tuotetiedoista päivittyy. Tulevaisuudessa, jos suosittelumalleja halutaan ajaa päivittäisen työn aikana, on mahdollista lisätä tapahtumapohjaisia käynnistimiä, mutta tässä työssä niiden tarkempi pohdinta jätetään jatkokehitykseen.

Suorituskyky ja tietojen ajantasaisuus

Suorituskyvyn osalta ratkaisu mitoitetaan PK-yrityksen tarpeisiin. Toimittajilta tulevien erien koko on käytännössä melko maltillinen. Esimerkiksi Continentalin esimerkkiaineistossa oli noin 1 500 riviä, ja tavoitteena on, että vastaavan kokoluokan erä voidaan käsitellä taustaprosesseissa selvästi alle tunnissa. Käytännössä tämä tarkoittaa, että järjestelmän olisi hyvä pystyä käsittelemään tämän kokoluokan erässä noin 20–25 riviä minuutissa niin, että validointi, kanoniseen tuotemalliin yhdistäminen ja tietokantapäivitykset ehtivät mukaan ilman, että taustaprosessien kuormitus muodostuu pullonkaulaksi.

Päivittäisessä käytössä tärkeämpää on rajapintojen vasteaika kuin eräajon läpimenoaika. Richards ja Ford (2020) korostavat ohjelmistoarkkitehtuuria käsittelevässä teoksessaan, että suorituskyky on yksi laadullinen ominaisuus muiden, kuten skaalautuvuuden ja käytettävyyden, joukossa, eikä sitä yleensä pyritä maksimoimaan irrallaan muista tavoitteista. Tässä työssä tavoitteena on, että ostotyökalun käyttämät mikropalvelun kutsut pysyvät normaalitilanteessa tyypillisesti alle muutaman sadan millisekunnin vasteajoissa. Nämä ovat käytännön tavoitteita, eivät kirjallisuudesta johdettuja raja-arvoja, mutta ne vastaavat karkeasti sitä tasoa, jota voidaan pitää riittävänä rengasostoja tekevän käyttäjän kannalta.

Asynkronisten taustaprosessien käyttö helpottaa sitä, että raskaat ajot voidaan hoitaa erillään käyttöliittymän toiminnasta. Tapahtumapohjaista käsittelyä kuvatessaan Richards ja Ford (2020) tuovat esiin, että viestijonoihin ja tapahtumiin perustuva malli voi parantaa koettua nopeutta ja skaalautuvuutta, vaikka taustalla kaikki vaiheet kestäisivät yhtä kauan ja tieto olisi vain lopulta yhdenmukaista. Sama perusajatus näkyy tässä ratkaisussa: ostajan ei tarvitse odottaa kokonaisen eräajon päättymistä, kunhan rajapinnan kautta näkyvät tiedot ovat riittävän tuoreita.

Tietojen ajantasaisuuden osalta ratkaisun kannalta hyväksytään pieni viive mikropalvelun ja Nettivanteen välillä. Suurimmalle osalle tuotteista riittää, että hinnat ja varastosaldot päivittyvät eräajojen yhteydessä. Kriittisissä tilanteissa, kuten yksittäisen tilauksen tekemisessä, käytetään aina toimittajan pull-kanavaa reaaliaikaisen saatavuuden ja hinnan varmistamiseen.

Virhetilanteiden käsittely taustaprosesseissa noudattaa samoja periaatteita kuin integraatioissa. Eräajon yksittäiset epäonnistuneet rivit pyritään käsittelemään automaattisesti uudelleen, ja vain toistuvasti epäonnistuvat tapaukset ohjataan manuaaliseen tarkasteluun. Hohpe ja Woolf (2004) korostavat, että virhekanavien ideana on ohjata epäonnistuneet viestit omaan käsittelyynsä ilman, että koko virtaa tarvitsee keskeyttää. Tässä työssä sama periaate sovelletaan toimittajien riveihin: pääosa datasta kulkee automaattisen prosessin läpi, ja vain poikkeukset vaativat ostajan tai pääkäyttäjän huomiota.

Tässä osuudessa rajapinnoille, taustaprosesseille ja suorituskyvylle on tehty karkea suunnitelma, jonka varaan myöhempi toteutus voidaan rakentaa. Yksityiskohtaiset päätökset, kuten tarkempi työjonokäytäntö tai eräajojen ajorytmi, voidaan määritellä projektin edetessä, kunhan tässä osiossa kuvattuja periaatteita noudatetaan. Tavoitteena on, että sisäinen rajapinta säilyy selkeänä ja REST-tyyppisenä, raskaat työt hoidetaan asynkronisina taustaprosesseina ja suorituskykytavoitteet asetetaan realistisesti PK-yrityksen tarpeiden mukaan.

6.5 Sääntö- ja valintamoottori

Tämän alaluvun aiheena on sääntö- ja valintamoottori, joka tukee ostajan työtä erityisesti hinnoittelussa ja tuotevalikoiman suunnittelussa. Tavoitteena ei ole automatisoida ostopäätöksiä, vaan rakentaa vaiheittain kehittyvä ratkaisu, joka laskee ehdotuksia ja jättää lopullisen valinnan asiantuntijalle. Ajattelu noudattaa päätöksenteon tukijärjestelmiä käsittelevää kirjallisuutta, jossa järjestelmä kokoaa ja jalostaa dataa, mutta ei tee päätöksiä käyttäjän puolesta (Kulkarni et al., 2007; Power, 2002).

Selkeyden vuoksi tässä alaluvussa “vaihe” tarkoittaa sääntö- ja valintamoottorin toiminnallista tasoa (mitä moottori osaa). Ratkaisun käyttöönoton vaiheet (MVP → pilotti/laajennus) kuvataan erikseen luvussa 6.8, ja tässä viitataan niihin vain silloin, kun moottorin taso kytkeytyy suoraan MVP-versioon.

Kirjallisuudessa hinnoittelua tarkastellaan usein omistamisen kokonaiskustannuksen (TCO) näkökulmasta. Ellram (1995) korostaa, että pelkkä yksikköhinta ei riitä, vaan mukaan on otettava alennukset, kampanjat, kuljetus, käsittely, hylkäykset ja muut kustannuserät. Ramakrishnan et al. (2019) osoittavat, että manuaalinen hinnoittelu on altis virheille ja muutama väärin hinnoiteltu tuote voi heikentää sekä kannattavuutta että asiakaskokemusta. Anta Callersten et al. (2024) kuvaavat käytännön mallia, jossa sääntö- tai tekoälymoottori laskee hinnoitteluehdotuksia, mutta asiantuntija hyväksyy tai hylkää ne. Tässä työssä sääntö- ja valintamoottorin suunnittelu käyttää samaa hybridiajattelua, jossa järjestelmä laskee hinnan ja muut ehdotukset, mutta ostaja päättää, mitä ehdotuksille tehdään.

Ensimmäinen toiminnallinen taso (MVP)

Ensimmäisessä toiminnallisessa tasossa (eli MVP-versiossa, ks. Luku 6.8) sääntömoottori keskittyy hinnoitteluun. Lähtökohtana on, että tuotteille lasketaan pelkästään kaavoihin perustuva suositeltu ostohinta ja myyntihinta. Laskenta perustuu listahintaan ja perusparametreihin, mutta hinnoittelun käyttäytymistä ohjataan aluksi valmiilla hinnoittelupreseteillä, jotka määritellään manuaalisesti. Näin hyödynnetään Ellramin (1995) TCO-ajattelua, mutta pysytään PK-yrityksen arkeen sopivassa yksinkertaisuudessa.

Hinnoittelulogiikkaa suunniteltiin tutkijan ja ostopäällikön yhteistyönä siten, että sitä tuetaan valmiilla "preseteillä", jotka vastaavat erilaisia tuotekategorioita. Esimerkkejä ovat esimerkiksi bulkkituote, harvinainen ja kallis koko tai erikoistuote, joissa hinnoittelun painotukset ja tavoitekate voivat poiketa toisistaan. Ostaja voi valita yksittäiselle tuotteelle sopivan presetin, ja sääntömoottori laskee sen perusteella suositellun hinnan. Näin hinnoittelusta saadaan hieman dynaamisempi ilman, että joudutaan heti rakentamaan täysin automaattista, tuote- ja tilannekohtaista optimointia. Ostajan on nopeampi valita presetti kuin laskea tuote kerrallaan hintoja. Myöhemmissä toiminnallisissa tasoissa presetit voidaan korvata hienommalla logiikalla, joka tunnistaa tuotetyypit automaattisesti, mutta tässä työssä ne toimivat helposti käyttöönotettavana lähtökohtana.

Ensimmäisessä toiminnallisessa tasossa sääntömoottori ei anna suosituksia tilattavista määristä. Määräpäätökset tekee ostaja, mutta ostotyökalu tukee päätöksiä esittämällä myyntihistorian, nykyiset varastosaldot ja yksinkertaisia tunnuslukuja, kuten myynti vali-

tulla ajanjaksolla. Käytännössä tämä tarkoittaa kevyttä DSS-tyyppistä tukea, jossa olennaiset luvut ovat yhdessä näkymässä ja helposti luettavassa muodossa, mutta varsinainen päätös jää ihmiselle (Kulkarni et al., 2007; Power, 2002).

Suunnitellut laajennukset ja valintamoottorin rooli

Seuraavissa toiminnallisissa tasoissa sääntö- ja valintamoottori laajenee hinnoittelusta kohti valikoiman ja määrien suunnittelua. Tavoitteena on, että ostotyökalu pystyy myöhemmin ehdottamaan esimerkiksi tilattavaa kappalemäärää sen perusteella, miten tuotteet ovat aiemmin liikkuneet ja millaisia varaston raja-arvoja on asetettu. Samalla voidaan lisätä näkymiä, joissa järjestelmä nostaa esiin tuotteita, joilla on poikkeuksellisen heikko kate tai joita järjestelmä pitää ostajan kannalta kiinnostavina esimerkiksi hyvän katteen takia.

Gadde et al. (2024) huomauttavat, että pelkkiin sääntöihin nojaavat hinnoittelumallit voivat jäädä jälkeen nopeasti muuttuvissa markkinaolosuhteissa ja että vahvistusoppimiseen perustuvat menetelmät voisivat periaatteessa oppia hinnoittelupolitiikan. He korostavat kuitenkin, että täysautomaattiset ratkaisut ovat käytännössä raskaita datatarpeiden, mallivalvonnan ja integraatioiden vuoksi (Gadde et al., 2024). Tässä työssä täysautomaattinen hinnoittelu tai vahvistusoppiminen rajataan PK-yrityksen kannalta lähinnä teoreettiseksi tulevaisuuden mahdollisuudeksi, eikä niitä suunnitella tarkemmin. Sääntö- ja valintamoottori toteutetaan hybridimallina, jossa järjestelmä laskee ja ehdottaa ja ihminen hyväksyy.

Sääntöjen rakenne, parametrisointi ja ylläpito

Sääntö- ja valintamoottori nojaa kanoniseen tuotemalliin, jossa on koottuna tuotteiden tekniset ominaisuudet, hinnat, tunnisteet ja varastotiedot. Kanoninen malli tarjoaa yhdenmukaiset kentät, joista sääntömoottori saa tarvitsemansa tiedot esimerkiksi TCO-laskentaa, katteen arviointia ja varaston tarkastelua varten. Sääntöjen näkökulmasta olennaisia ovat erityisesti hinnan eri komponentit, myynti- ja varastotiedot, tuotteen luokittelu sekä mahdolliset "preset" -kategoriat.

Sääntöjen ylläpito ja kehitys perustuu käytännössä ostajan ja kehittäjän yhteistyöhön. Tässä vaiheessa kyse on yhden ostajan työstä, mutta mallin on tarkoitus toimia myös useamman ostajan tilanteessa. Ostaja määrittelee, millaisia kynnysarvoja, painotuksia ja presettejä tarvitaan ja antaa palautetta siitä, miltä suositukset näyttävät normaalissa työssä. Kehittäjä muuntaa nämä käytännön periaatteet konfiguroitaviksi säännöiksi ja kaavoiksi, joita voidaan säätää ilman, että koko järjestelmää tarvitsee muuttaa.

Power (2002) kuvaa nopeaa prototypointia kehitysmallina, jossa DSS-ratkaisua rakennetaan iteratiivisesti ja vaatimuksia tarkennetaan prototyyppien ja käytännön kokeilujen

kautta. Tässä työssä sääntö- ja valintamoottorin kehitys noudattaa samaa periaatetta: pieniä muutoksia viedään tuotantoon, niiden vaikutuksia seurataan ja sääntöjä hienosäädetään ostajan ja kehittäjän yhteistyönä.

Ohjauksen ja hallinnan näkökulmasta sääntöjen läpinäkyvyys ja dokumentointi ovat osa tiedonhallinnan ohjausta (data governance). Tässä ratkaisussa sääntö- ja valintamoottorin kannalta keskeiset periaatteet dokumentoidaan ja säännöistä pidetään kirjaa vähintään sillä tasolla, että myöhemmin voidaan nähdä, millä periaatteilla tiettyihin suosituksiin on päädytty. Jos esimerkiksi hinnoittelupresettejä muutetaan, muutoksista jää dokumentoitu jälki, vaikka varsinaista versiohallintaa ei vielä toteutettaisi järjestelmässä.

Ihminen silmukassa ja käyttöliittymän rooli

DSS-kirjallisuudessa korostuu ajatus siitä, että järjestelmän tehtävä on tukea käyttäjää eikä korvata häntä. Kulkarni et al. (2007) ja Power (2002) esittävät malleja, joissa data- ja mallipohjaiset toiminnot yhdistyvät siten, että käyttäjä näkee olennaisen tiedon ja suositukset, mutta voi säätää ja ohittaa niitä. Anta Callersten et al. (2024) kuvaavat käytännön hinnoittelumallia, jossa algoritmin ehdotukset hyväksytään tai hylätään asiantuntijan toimesta. Tässä työssä sääntö- ja valintamoottori suunnitellaan alusta lähtien tällaiseksi ihminen silmukassa -ratkaisuksi.

Käytännössä tämä tarkoittaa, että ostaja ei näe pelkkää lopputulosta, vaan myös keskeiset perustelut. Näkymässä, jossa tehdään päivittäisiä täydennysostoja, ostotyökalu voi esittää tuotekohtaisesti suositellun hinnan, myyntihistorian, varastosaldot ja karkean arvion katteesta. Ostaja voi tämän perusteella valita hinnoittelupresetin, säätää suositeltua hintaa ja päättää, kuinka paljon tuotetta tilataan ja mihin varastoon se ohjataan. Jos ollaan tekemässä sesonkiostoja, näkymä voi painottaa valikoiman suunnittelua ja nostaa esiin erityisesti ne tuotteet, joita järjestelmä pitää sesongin kannalta kiinnostavina.

Käyttöliittymässä tehdään selväksi, mikä tieto ja mikä suositus tulee sääntö- ja valintamoottorilta ja mitä muutoksia käyttäjä on itse tehnyt. Tämä tukee luottamusta järjestelmään ja helpottaa myöhempää jäljitettävyyttä, jos suosituksia halutaan arvioida jälkikäteen.

Yhteenveto

Yhteenvetona sääntö- ja valintamoottori toimii tässä ratkaisussa ostajan apurina, ei korvaajana. Se laskee TCO-tyyppiseen ajatteluun pohjautuvia hintaehdotuksia, hyödyntää kanonisen tuotemallin yhtenäisiä tietoja ja tarjoaa valikoiman suunnitteluun liittyviä näkymiä, joissa suositukset ja perustelut näkyvät selkeästi. Ratkaisu rakennetaan vaiheittain siten, että ensimmäisessä toiminnallisessa tasossa keskitytään hinnoittelun tukemiseen ja myöhemmissä toiminnallisissa tasoissa voidaan lisätä hienompia ominaisuuksia,

kuten määräsuosituksia ja valikoiman laajempaa analyysiä. Näin sääntö- ja valintamoottori kytkeytyy luontevasti sekä PIM- ja TCO-kirjallisuuteen että DSS-ajatteluun, jossa ostotyökalu jalostaa tietoa, mutta lopullinen päätös jää asiantuntijalle (Ellram, 1995; Abraham, 2014; Kulkarni et al., 2007; Power, 2002; Anta Callersten et al., 2024).

6.6 Käyttöliittymä ja käyttäjäpolut

Tässä osiossa kuvataan lyhyesti, millainen ostotyökalun käyttöliittymä on ja miten sitä käytetään. Painotus on ostajan työssä, ei käyttöliittymän yksityiskohtaisessa suunnittelussa. Käyttöliittymä toimii sääntö- ja valintamoottorin sekä kanonisen tuotemallin esittäjänä, eli se kokoaa tiedon yhteen näkymään ja tukee päätöksentekoa niin, että ihminen tekee lopullisen ratkaisun, kuten päätöksenteon tukijärjestelmiä (DSS) käsittelevässä kirjallisuudessa korostetaan (Kulkarni et al., 2007; Power, 2002).

Ostajan päätyönäkymä

Ostajan päätyönäkymä on taulukkomuotoinen lista tuotteista. Riveittäin esitetään tuotetunnus ja kuvaus, keskeiset raakatiedot (nykyinen ostohinta, myyntihinta, varastosaldo ja valitun ajanjakson myynti) sekä sääntömoottorin laskema suositus, kuten ehdotettu myyntihinta. Näkymässä erotellaan selkeästi, mikä tieto on raakaa taustajärjestelmästä, mikä on sääntö- ja valintamoottorin laskema arvo ja mitä ostaja on muuttanut käsin.

Työskentely tapahtuu ensisijaisesti massapäivityksenä. Ostaja suodattaa ja järjestää tuotteita esimerkiksi sen mukaan, missä hinnat ovat muuttuneet eniten tai missä kate poikkeaa tavoitetasosta. Yksittäisen rivin voi avata tarkempaan näkymään, mutta pääosa työstä tehdään taulukossa. Hinnoittelua varten käyttöliittymä voi tukea kevyttä what-if-tyyppistä tarkastelua, jossa ostaja säätää hintaa ja näkee samalla, miten muutos vaikuttaisi katteeseen kyseisellä tuotteella. Näin DSS-ajattelu toteutuu käytännössä siten, että analyysi ja suositus ovat näkyvissä samassa näkymässä, mutta lopullinen päätös jää asiantuntijalle (Power, 2002).

Muutokset, virheet ja suositukset

Päätyönäkymää täydentävät muut näkymät, jotka helpottavat datan käsittelyä. Muuttuneiden tuotteiden näkymä kokoaa tuotteet, joissa toimittajadata tai sääntömoottorin laskema suositus on muuttunut nykyisestä tilanteesta. Näin ostaja voi priorisoida tarkastelun niihin tuotteisiin ja päivittää tuote- ja hintatiedot nopeasti.

Aikaisemmin työssä kuvatulle virhejonolle tulee olla näkymä, joka kokoaa tuotteet ja erät, joita ei ole voitu käsitellä automaattisesti, esimerkiksi puutteellisten tunnisteiden tai ristiriitaisten tietojen vuoksi. Tarkoitus on, että vain nämä poikkeustapaukset vaativat erillistä

huomiota ja pääosa datasta kulkee automaattisesti taustaprosessien läpi. Myöhemmin voidaan lisätä erillinen suosittelunäkymä, jossa järjestelmä nostaa esiin esimerkiksi ostatunnisteella varustettuja tuotteita. Lisäksi ostotyökalu voisi nostaa esille tuotteita, joilla kate on poikkeuksellisen heikko. Nämä suositukset ovat tässä työssä konseptitasolla, eivät varsinaisina toteutusvaatimuksina.

Kaikissa näissä näkymissä perusperiaate on sama: ostaja näkee, mitä toimittaja on toimittanut, mitä järjestelmä ehdottaa ja mitä muutoksia hän on itse tehnyt. Tämä tukee läpinäkyvää ihminen silmukassa -mallia, johon myös DSS-kirjallisuus viittaa (Kulkarni et al., 2007; Power, 2002; Anta Callersten et al., 2024).

Integraatio Nettivanteeseen ja käyttökonteksti

Ostotyökalu toteutetaan React-pohjaisena käyttöliittymänä, joka upotetaan osaksi nykyistä Nettivanne-järjestelmää. Käytännössä se näkyy uutena osana Nettivanteen päävalikossa ja hyödyntää samaa kirjautumista ja käyttöoikeusmallia. Työkalu on suunniteltu käytettäväksi pöytäkoneella ja kannettavalla, eikä mobiilikäyttöä optimoida ensimmäisessä vaiheessa. Käyttöliittymä pidetään visuaalisesti ja rakenteellisesti linjassa nykyisen järjestelmän kanssa, jotta käyttöönotto sujuu mahdollisimman kevyesti ja ostajan työskentelytapa säilyy tutun oloisena.

6.7 Tietoturva, pääsynhallinta ja auditointi

Tässä työssä tietoturvaa, pääsynhallintaa ja auditointia käsitellään periaatetasolla osana tiedonhallinnan ohjausta, erityisesti niin sanottua “data access” -aluetta, jossa määritellään, kuka saa käyttää mitäkin tietoa ja millä ehdoilla (Khatri & Brown, 2010). Koska ostotyökalu ohjaa suoraan yrityksen rahankäyttöä, perusajatus on, että oikeudet rajataan tiukasti ja että kriittisistä päätöksistä jää selkeä jälki.

Käyttöoikeudet ja roolit

Ostotyökalu tukeutuu Nettivanteen olemassa olevaan käyttäjähallintaan ja kirjautumiseen. Ostotyökalu ei siis rakenna erillistä tunnistautumista, vaan käyttää samaa käyttäjäkantaa, istunnonhallintaa ja TLS-suojattua (Transport Layer Security, TLS) yhteyttä kuin muu järjestelmä. Nettivanteeseen voidaan määritellä erillinen ostajarooli, jota käytetään ostotyökalussa vähimmän oikeuden periaatteella: vain ostajaroolin omaavat käyttäjät voivat vahvistaa ostopäätöksiä tai muuttaa hintoja, eikä muille anneta turhia oikeuksia. Herkempiä kenttiä voidaan myös myöhemmin suojata tarkemmalla pääsynhallinnalla, jos on tarvetta.

Tekniset suojaukset

Teknisesti ostotyökalu noudattaa samoja perustason käytäntöjä kuin muu selainpohjainen järjestelmä. Näihin kuuluvat vahva tunnistautuminen, suojattu istunnonhallinta ja salattu tiedonsiirto. NIST:n digitaalisia identiteettejä koskevat ohjeet korostavat, että verkon yli tapahtuva tunnistautuminen tulee suojata hyväksytyillä salausmenetelmillä ja istunnonhallinnassa on huolehdittava esimerkiksi istunnon salaisuudesta ja aikarajoista (National Institute of Standards and Technology [NIST], 2017). Toimittajarajapinnat ja eräajot toteutetaan SFTP:n ja rajapintojen avulla niin, että yhteydet ovat salattuja ja todennus perustuu avain- tai token-pohjaisiin mekanismeihin, jotka voidaan tarvittaessa uusia ilman, että varsinainen käyttäjäroolirakenne muuttuu.

Lokitus ja auditointijälki

Auditoinnin kannalta valitut tapahtumat kirjataan niin, että yksittäinen ostopäätös voidaan jälkikäteen yhdistää tiettyyn tuotteeseen, versioon ja käyttäjään. Käytännössä ostajan tekemät vahvistukset ja manuaaliset muutokset tallentuvat kanoniseen tuotemalliin sekä erilliseen päätöslokiin, jossa säilytetään esimerkiksi aikaleima, käyttäjätunnus ja tieto siitä, mitä arvoja järjestelmä alun perin ehdotti. Lisäksi eräajot ja rajapintakutsut tuottavat lokitietoa onnistuneista ja epäonnistuneista ajoista. OWASP Application Security Verification Standard korostaa, että sovelluksessa tulisi olla riittävät ja suojatut lokit turvallisuuteen ja liiketoimintaan liittyvien tapahtumien seurantaa ja jälkiselvityksiä varten (OWASP, 2023). Tässä työssä auditointimalli pidetään kuitenkin yksinkertaisena. Lähtökohtana on, että kriittinen muutos ei koskaan jää “henkilöstä ja hetkestä irralliseksi”, vaan sille löytyy päätöspolku lokista.

Kaiken kaikkiaan tietoturva ja pääsynhallinta tukeutuvat Nettivanteen ominaisuuksiin ja auditointi perustuu yleisesti hyväksyttyihin periaatteisiin. Tässä luvussa nämä rajataan periaatetasolle, eli yksityiskohtaiset asetukset ja menetelmät määritellään ja testataan käyttöönoton yhteydessä.

6.8 Käyttöönotto, toimittajan lisääminen ja tuoteryhmien laajentaminen

Tässä osiossa kuvataan, miten ostotyökalu otetaan käyttöön vaiheittain, miten uusia toimittajia liitetään mukaan ja miten ratkaisu voidaan myöhemmin laajentaa renkaista muihin tuoteryhmiin. Luvussa 2 todettiin, että PK-yritysten digitalisointi etenee käytännössä inkrementaalisin askelin sen sijaan, että kaikkea muutettaisiin kerralla (Aliyev, 2025; Vial, 2019). Tässä luvussa sama ajattelutapa otetaan mukaan käytännön suunnitelmaan, eli

rakennetaan ensin toimiva perusversio yhden toimittajan ja rengastuotteiden varaan ja laajennetaan siitä vaiheittain eteenpäin.

Selkeyden vuoksi tässä alaluvussa “vaihe” tarkoittaa ratkaisun käyttöönoton etenemistä (MVP → laajennus), kun taas sääntö- ja valintamoottorin toiminnalliset tasot on kuvattu erikseen luvussa 6.5.

Ensimmäinen versio: yksi toimittaja ja renkaat

Ensimmäisessä vaiheessa ostotyökalu otetaan käyttöön yhden toimittajan ja rengastuotteiden kanssa. Tavoitteena on, että ostaja saa heti käyttöön alustavan taulukkomaisen näkymän, jossa toimittajalta tuleva tuotetieto on koottu yhteen ja tuotteita voidaan selata, muokata ja tilata suoraan työkaluista käsin. Hinnoittelu perustuu tässä vaiheessa ostajan manuaalisiin päätöksiin: hinnat voi syöttää ja muuttaa työkalussa, mutta varsinainen hinnoittelun tuki ja sääntömoottori eivät vielä ole käytössä. Käyttöönoton ehto on, ettei ostajan työ hidastu, vaan tilaukset kulkevat edelleen Nettivanteen kautta ja pysyvät synkassa nykyisten prosessien kanssa.

Tämä ensimmäinen versio toimii käytännössä kevyenä “taulukko-ohjelmana”, joka on kiinni kanonisessa tuotemallissa ja integraatioissa, mutta jonka logiikka pidetään riittävän yksinkertaisena, jotta järjestelmä saadaan käyttöön ja sitä voidaan testata yhdessä ostopäällikön kanssa. Näin varmistetaan, että perusprosessit toimivat ja että kanonisen tuotemallin rakenne tukee normaalia ostotyötä renkaiden osalta, ennen kuin mukaan tuodaan monimutkaisempia ominaisuuksia.

Toinen versio (laajennus): hinnoittelun tuki ja useampi toimittaja

Kun ensimmäinen versio on käytössä ja prosessi yhden toimittajan kanssa on todettu toimivaksi, seuraavassa vaiheessa lisätään hinnoittelun tuki ja mahdollisuus käyttää ostotyökalua useamman toimittajan kanssa. Käytännössä tämä tarkoittaa sitä, että luvussa 6.5 kuvattu sääntö- ja valintamoottori otetaan rajoitetusti käyttöön renkaiden hinnoittelussa, esimerkiksi ennalta määriteltyjen "presettien" muodossa. Tavoite ei ole automatisoida päätöksiä, vaan tuoda ostajan näkymään suositeltu hinta ja antaa mahdollisuus tarkastella nopeasti, miten eri valinnat vaikuttavat katteeseen.

Samassa vaiheessa mukaan voidaan liittää toinen ja kolmas toimittaja. Käytännön etenemislogiikka on, että kullekin uudelle toimittajalle rakennetaan adapteri, joka muuntaa toimittajan datan kanonisen tuotemallin muotoon. Integraation aluksi käydään läpi toimittajan dokumentaatio ja tekniset mahdollisuudet, valitaan sopiva siirtotapa (erätiedosto tai rajapinta) ja määritellään, miten keskeiset kentät, kuten tuotetunnukset, koot ja hinnat, mapataan yhteiseen malliin. Tämän jälkeen voidaan määritellä toimittajakohtaiset validoinnit, testata eräajot rajatulla aineistolla ja vasta sitten kasvattaa volyymia. Ostajalle

tämä näyttäytyy mahdollisimman yhtenäisenä näkymänä, jossa toimittajasta riippumatta tuotteet ovat samassa rakenteessa.

Uuden toimittajan lisääminen

Uuden toimittajan lisääminen noudattaa periaatteessa samaa kaavaa riippumatta siitä, missä vaiheessa käyttöönottoa ollaan. Ensin selvitetään, millaista tuotetietoa toimittaja tarjoaa ja missä muodossa. Sen jälkeen määritellään, miten tiedot sovitetaan kanoniseen tuotemalliin ja mitä oletuksia tai muunnoksia tarvitaan. Tämän pohjalta toteutetaan adapteri ja validaattorit, jotka varmistavat esimerkiksi sen, että pakolliset tunnisteet löytyvät ja keskeiset mitat ja hinnat ovat järkevällä välillä. Pilottivaiheessa ajetaan rajattu määrä dataa järjestelmän läpi, verrataan tuloksia nykyiseen prosessiin ja korjataan havaittuja virheitä ennen kuin siirrytään normaaliin isompaan erään tietoa.

Powerin (2002) kuvaama päätöksenteon tukijärjestelmien kehitys noudattaa usein juuri tällaista prototyyppipohjaista ja iteratiivista mallia, jossa ensin rakennetaan rajattu mutta toimiva järjestelmä ja sitä laajennetaan käyttäjäpalautteen ja havaintojen perusteella. Tässä työssä sama ajattelu näkyy siinä, että ostotyökalu otetaan käyttöön ensin yhdessä kapeassa tapauksessa, minkä jälkeen integraatioita ja hinnoittelun tukea laajennetaan askel kerrallaan.

Laajennus renkaista muihin tuoteryhmiin

Renkaiden lisäksi yrityksen ostoissa on muita tuoteryhmiä, kuten vanteet ja huoltoon liittyvät tarvikkeet. Tässä työssä varsinainen suunnittelu ja testaus rajataan renkaisiin, mutta luvussa 6.1 todettiin jo, että arkkitehtuuria ja kanonista tuotemallia rakennetaan siten, että sama perusrakenne voidaan myöhemmin laajentaa myös vanne- ja tarvikeostoihin ilman merkittävää uudelleensuunnittelua. Käytännössä laajennus muihin tuoteryhmiin tapahtuu sen jälkeen, kun renkaiden kanssa toteutetut integraatiot ja hinnoittelu on saatu vakiintuneeseen käyttöön.

Ensimmäinen luonteva laajennusaskel on vanteet, koska ne muistuttavat monella tapaa renkaita ja jakavat useita samoja attribuutteja, kuten koko- ja mallimääritykset. Kun vanteet on saatu mukaan, samaa mallia voidaan soveltaa huoltotuotteisiin ja tarvikkeisiin, joissa datan rakenne voi olla vaihtelevampi. Näiden tuoteryhmien osalta tarkempi suunnittelu ja pohdinta jäävät yrityksen jatkopäätösten varaan.

Koulutus ja iteratiivinen kehittäminen

Käyttöönottoa ei suunnitella erilliseksi “isoksi koulutushankkeeksi”, vaan ostaja on mukana suunnittelussa ja testauksessa alusta lähtien. Tämä tarkoittaa käytännössä sitä,

että ostaja oppii työkalun peruskäytön samassa tahdissa, kun toiminnallisuuksia rakennetaan ja otetaan käyttöön. Ensimmäisessä vaiheessa riittää, että dokumentoidaan peruskäyttö ja muutama tyypillinen työtilanne, jotta osaaminen voidaan jakaa myöhemmin myös muille ostajille ja myyjille.

Aliyev (2025) korostaa, että PK-yritysten prosessinkehitystä tukevat nimenomaan pienet, konkreettiset edistysaskeleet, joista saadaan nopeasti palautetta ja oppia. Vialin (2019) esittämä näkemys digitaalisesta muutoksesta jatkuvana prosessina tukee samaa ajatusta siitä, että teknistä ratkaisua ei rakenneta kerralla valmiiksi, vaan sitä kehitetään yrityksen kyvykkyyksien ja kokemusten karttuessa. Tässä työssä esitetty käyttöönotto- ja laajennuspolku toimii käytännön suunnitelmana tällaiselle vaiheittaiselle etenemiselle.

6.9 Dokumentointi ja tiedonhallinnan ohjaus

Tämän osion tarkoitus on kuvata, miten ostotyökalun dokumentointi ja siihen liittyvä tiedonhallinnan ohjaus järjestetään käytännössä. Tavoitteena on, että käytännöt ja tapahtumat eivät jää yksittäisten henkilöiden hiljaiseksi tiedoksi, vaan ovat löydettävissä ja saatavilla, kun ostotyökalu ja toimittajakanta kehittyvät.

Dokumentointi kootaan neljään kokonaisuuteen, mutta esitystapa pidetään kevyenä. Ensimmäiseksi laaditaan käyttäjäohje, joka näyttää tyypilliset työtilanteet, esimerkiksi miten uutta toimittajadataa tarkastellaan, miten tuotteita suodatetaan ja järjestetään, miten hintoja muokataan ja miten tilaukset lähetetään. Toiseksi dokumentoidaan sääntö- ja valintamoottorin periaatteet ja sen eri versiot. Tarkoitus on, että ostaja näkee, millä logiikalla järjestelmä tuottaa ehdotuksia ja millainen on tämän ominaisuuden historia. Kolmanneksi kuvataan toimittajakohtaiset käytännöt ostajan näkökulmasta. Tähän kuuluu esimerkiksi se, miten kyseisen toimittajan kanssa yleensä toimitaan, millaisia kampanjoita ja bonuksia on tavallista saada ja mitä erityispiirteitä toimitusehdoissa on. Neljänneksi kirjataan lyhyesti, miten auditointia hyödynnetään työssä, eli miten ostaja ja johto voivat tarvittaessa tarkistaa, mitä on ostettu, millä periaatteilla ja milloin.

Hiljaisen tiedon näkyväksi tekemiseksi tämän työn kannalta olennaiset toimittajakohtaiset käytännöt kootaan yhteen samaan paikkaan. Käytännössä tämä tarkoittaa ostopäällikön kokoamia toimittajakortteja, joissa on lyhyt kuvaus toimittajasta, tavallisista sopimuskäytännöistä, hinnoittelun erityispiirteistä ja tyypillisistä neuvottelutilanteista. Uuden ostajan on tarkoitus pystyä nopeasti lukemaan näistä korteista, miten kunkin toimittajan kanssa on tapana toimia, sen sijaan että käytännöt siirtyisivät vain suullisesti henkilöltä toiselle.

Tekninen taustadokumentointi toteutetaan “doc-as-code” -ajattelulla. Integraatioihin, kanoniseen tuotemalliin ja rajapintoihin liittyvä dokumentaatio säilytetään samassa versionhallinnassa kuin koodi ja sitä päivitetään normaalin kehitysprosessin osana. Näin varmistetaan, että tietomallit, rajapintakuvaukset ja eräajojen toimintaperiaatteet pysyvät ajan tasalla ilman erillistä dokumentointiprojektia. Samalla ostajalle näkyvä dokumentaatio voidaan linkittää suoraan taustalla olevaan tekniseen kuvaukseen silloin, jos tälle on tarvetta.

Tutkija ja ostopäällikkö sopivat dokumentoinnin vastuista niin, että sääntöjen ja presettien dokumentointi on ostopäällikön vastuulla, koska hän määrittelee niiden sisällön ja käyttää niitä työssään. Myös yleisten ohjeiden kokonaisvastuu on ostopäälliköllä, joka huolehtii siitä, että käyttöohjeet, toimittajakortit ja periaatteet pysyvät riittävän ajantasaisina. Näihin voidaan käyttää kehittäjän tukea, jos sitä tarvitaan. Kehittäjät vastaavat puolestaan teknisen dokumentaation ylläpidosta osana normaalia kehitystyötä.

Näin dokumentointi tukee myös luvussa 3.2 käsiteltyä tiedonhallinnan ohjausta (data governance), jossa päätösoikeudet, vastuut ja käytännön toimintatavat tehdään näkyviksi eikä niitä jätetä yksittäisten henkilöiden muistin varaan (Khatri & Brown, 2010).

7. TULOKSET JA ARVIOINTI

Tämän luvun tarkoituksena on arvioida työn tuotoksia ja vastata tutkimuskysymyksiin. Ensin kuvataan, miten suunniteltu ratkaisu vastaa päätutkimuskysymykseen ja alatutkimuskysymyksiin. Tämän jälkeen tarkastellaan monikriteerisen päätöksenteon menetelmän soveltuvuutta, ratkaisun liiketoiminnallista potentiaalia sekä työn rajoitteita ja menetelmän toimivuutta tässä ympäristössä.

7.1 Vastaus tutkimuskysymyksiin

Työn päätutkimuskysymys oli:

”Miten Rengas Turku Oy:n ostoprosessia voidaan digitalisoida siten, että manuaalityö, virheriski ja läpimenoaika vähenevät, tuotetiedon laatu paranee ja hinnoittelua voidaan tukea päätöksenteon tukijärjestelmällä PK-yrityksen reunaehdot huomioiden?”

Tähän vastataan kokonaisuudella, joka koostuu erillisestä ostopalvelusta, kanonisesta tuotemallista, yhdistetystä integraatiosta, joka koostuu push- ja pull-virroista sekä vaiheistetusta käyttöönottosuunnitelmasta. Ratkaisu pysyy yrityksen nykyisen Django-pohjaisen modulaarisen monoliitin rinnalla ja hyödyntää olemassa olevia rakenteita. Mikropalvelu rajataan ostoprosessin digitalisointiin (ostotyökalu ja integraatiot), kun taas varsinainen ydinjärjestelmä säilyy monoliittina. Tämän käyttöönotto tukee Aliyevin (2025) ja Omowole et al. (2024) kuvaamaa vaiheittaista digitalisointia. Ensin automatisoidaan eniten viivettä ja virheitä aiheuttavat kohdat, eikä koko järjestelmää uudisteta kerralla.

Alatutkimuskysymysten osalta tulokset voidaan tiivistää seuraavasti:

TK1: Mitkä ovat nykyisen ostoprosessin keskeiset kipupisteet ja digitalisoinnin tarpeet?

Empiirisen kartoituksen perusteella kriittisiksi ongelmakohdiksi nousivat sesonkitilausten raskas manuaalinen käsittely, toimittajakohtaisten tuotetaulukoiden ja kampanjoiden pirstaleisuus sekä se, että hinnoittelu nojautuu liikaa yksittäisten asiantuntijoiden hiljaiseen tietoon. Nämä havainnot vastaavat kirjallisuudessa kuvattuja PK-yritysten digitalisoinnin tyypillisiä haasteita, joissa manuaaliset prosessit, tiedon hajanaisuus ja henkilöriippuvuus hidastavat kehitystä (Aliyev, 2025; Omowole et al., 2024).

TK2: Millä kriteereillä arkkitehtuurivaihtoehtoja tulisi arvioida ja valita?

Työssä johdettiin monikriteerisen päätöksenteon malli, jossa vaihtoehtoja verrattiin kustannusten, teknisen vaativuuden ja riskin lisäksi ylläpidettävyyden, skaalautuvuuden, tiedon hallinnan ja työkalupotentiaalin näkökulmista. Kriteerit perustuvat sekä kirjallisuuteen modulaarisesta monoliitista ja mikropalveluista (esimerkiksi Richards & Ford, 2020; Newman, 2021; Pahl & Jamshidi, 2016) että yrityksen sisäisiin keskusteluihin ja reunaehtoihin. Lopputuloksena määritettiin painotus, joka korostaa pitkän aikavälin ylläpidettävyyttä, työkalupotentiaalia ja tiedon hallintaa, mutta ei unohda PK-yrityksen resurssirajoitteita ja integraatioriskejä.

TK3: Millainen kanoninen tuotemalli tukee ostoprosessin digitalisointia ja hinnoittelua?

Työssä muodostettiin kanoninen tuotemalli, joka yhdistää renkaiden, vanteiden ja muiden tuotteiden keskeiset attribuutit yhteen pohjamalliin. Malli kokoaa toimittajakohtaisen datan, sisäisen tuotehallinnan ja varastotiedot yhteen paikkaan, jolloin samaa rakennetta voidaan käyttää sekä tilausehdotusten, hinnoittelun että myyntikanavien taustalla. Tämä vastaa kirjallisuudessa kuvattua tavoitetta, jossa PIM ei jää pelkäksi tekniseksi varastoksi vaan kiinnittyy osaksi laajempaa prosessia ja päätöksenteon tukijärjestelmää.

TK4: Millainen integraatioarkkitehtuuri tukee PK-yrityksen ostoprosessin digitalisointia ja päätöksenteon tukea?

Tutkimuksessa tunnistettiin yhdistetty push- ja pull-virta, jossa toimittajien aineistot tuodaan sisään PRICAT- ja INVRPT-viesteillä ja konelukuisina tiedostoina. Mikropalvelu tarjoaa yrityksen sisäisille ja ulkoisille järjestelmille rajapinnan jalostettuun ja yhtenäistettyyn tuotetietoon. Käytännössä ratkaisu tukeutuu sekä perinteiseen EDI-aineistoon että API-pohjaisiin integraatioihin. Toteutus on käytännössä linjassa Omowole et al. (2024) esittämän kuvauksen kanssa, jossa pilvipalvelut ja automaatio toimivat integraatioiden ja analytiikan runkona, mutta tässä tapauksessa arkkitehtuuri pysyy tietoisesti kevyempänä ja yrityksen nykyiseen ympäristöön sidottuna.

TK5: Miten ratkaisun käyttöönotto tulisi vaiheistaa PK-yrityksessä?

Työssä suunniteltiin etenemismalli, jossa ensimmäisessä vaiheessa keskitytään sesonkitilausten tehostamiseen ja kanonisen tuotemallin käyttöönottoon yhdellä toimittajalla. Seuraavissa vaiheissa laajennetaan sekä toimittajakantaa että tilausprosesseja ja kytketään DSS-tyyppinen ominaisuus tiiviimmin osaksi normaalia tilaustyötä. Tämä toimintakaava noudattaa Aliyevin (2025) ja Vialin (2019) esittämää ajatusta siitä, että PK-yrityksessä kannattaa edetä pienin, näkyvin askelin ja sitoa tekninen eteneminen suoraan liiketoiminnan kipukohtiin.

Yhteenvetona voidaan todeta, että tutkimuskysymyksiin vastattiin suunnittelemalla kokonaisratkaisu, joka yhdistää arkkitehtuurivalinnan, kanonisen tuotemallin, integraatiot ja etenemissuunnitelman. Ratkaisu on tarkoituksella suunnitelmatasoinen eikä vielä tuotantokäytössä, mutta se on ankkuroitu sekä kirjallisuuteen että yrityksen käytännön tarpeisiin.

7.2 Menetelmän soveltuvuus ja luotettavuus

Ratkaisuvaihtoehtojen vertailu tehtiin painotetun summamallin (SAW) mukaisena monikriteerisenä päätöksentekona. Vaihtoehdot pisteytettiin yhteisesti sovituilla kriteereillä, ja kriteerit muunnettiin samaan suuntaan siten, että suurempi pistemäärä tarkoitti parempaa lopputulosta. Lopullinen arvo muodostui kriteeripisteiden ja painojen tulona lasketusta summasta. Menetelmä on linjassa Tzengin ja Huangin (2011) kuvaaman kehyksen kanssa.

Menetelmän vahvuutena on se, että kriteerit johdettiin empiirisistä havainnoista ja yrityksen teknisistä reunaehdoista, ei pelkästään kirjallisuuden pohjalta. Pisteytys ja painot dokumentoitiin taulukkolaskentaan ja niiden vaikutusta tulokseen tarkasteltiin herkkyysanalyysillä. Analyysin perusteella mikropalvelu säilyi parhaana vaihtoehtona myös silloin, kun painotusta muutettiin selvästi sitä vastaan. Tämä tukee johtopäätöstä, että valinta ei ole herkkä kohtuullisille painomuutoksille.

Toisaalta menetelmä nojaa vahvasti asiantuntija-arvioon. Tekijä on työskennellyt yrityksessä ja tuntee hyvin järjestelmän, ostoprosessin ja käytännön kipukohdat. Tämä on etu, koska hiljaista tietoa saatiin mukaan pisteytykseen ja kriteerien määrittämiseen. Samalla se lisää riskiä sokeille pisteille, koska arvio ei ole täysin ulkopuolinen. Tässä työssä tutkija pienensi riskiä tekemällä pisteytyksen yhdessä yrityksen edustajien kanssa, mutta täysin objektiivista tulosta menetelmällä ei voi saada.

Luotettavuuden kannalta merkittävä rajoite on se, että tekninen testaus ja integraatiotarkastelu kohdistuivat käytännössä yhteen toimittajaprofiiliin (Continental EASY XML) ja rajattuihin EDI-aineistoihin. Näin ollen integraatioiden siirrettävyys muihin toimittajiin ja järjestelmän joustavuus usean toimittajan tilanteessa jäävät vielä testaamatta. Tätä epävarmuutta ei ratkaista laskennalla, vaan se vaatii pilottia ja käytännön testaamista.

Menetelmä tuottaa siis hyvin perustellun ja läpinäkyvän valinnan kahden vaihtoehdon välillä. Se ei kuitenkaan yksin takaa, että ratkaisu toimii odotetusti kaikissa tulevissa integraatiokohteissa tai yrityksen muuttuessa. Tätä taustaa vasten on perusteltua, että mikropalveluun päätyvä valinta on kuvattu nimenomaan tiekarttana ja suunnitelmana, ei vielä valmiina käyttöönottona.

7.3 Ratkaisun vaikutukset ja liiketoiminnallinen potentiaali

Työn keskeinen tavoite oli vähentää manuaalista työtä, virheriskiä ja läpimenoaikaa sekä tukea hinnoittelua tavalla, joka sopii PK-yrityksen resursseihin. Tässä vaiheessa vaikutuksia voidaan arvioida vain suunnitelmien tasolla, mutta jo nämä viittaavat merkittävään parannukseen aikaisemmasta.

Sesonkitilausten osalta nykyinen toimintamalli sitoo suuren osan ostajan työajasta tarkistuksiin, hinnanmääritykseen ja tiedonhakuun eri lähteistä. Jos arvioidaan, että sesonkitilausten käsittely vie tällä hetkellä yli puolet ostajan työajasta kahden kuukauden sesonkijaksolla, ja digitaalinen ratkaisu pystyy leikkaamaan tästä työmäärästä esimerkiksi 30–40 prosenttia, kyse on käytännössä 20–25 työpäivän säästöstä sesonkia kohden. Vuodessa tämä tarkoittaa helposti 40–50 työpäivän vapautumista muuhun työhön. Sama työpanos voidaan kohdistaa paremmin poikkeustilanteiden käsittelyyn, toimittajaneuvotteluihin ja hinnoittelun kehittämiseen sen sijaan, että aikaa kuluu ylimääräiseen rutiinityöhön.

Päivittäistilausten puolella tilanne on melko sama, mutta tasaisempi. Jos keskimääräinen päivittäistilausprosessi vie nykyisin noin kaksi tuntia ja digitaalinen tuki esimerkiksi onnistuu puolittamaan tämän, säästö on luokkaa yksi tunti päivässä. Vuoden mittakaavassa tämä tarkoittaa yli kolmensadan tunnin säästöä työajassa. Yhdistämällä sesonki- ja päivittäistilausten tehostumisen voidaan karkeasti arvioida, että ratkaisulla on realistinen mahdollisuus säästää ostopäälliköltä yli kuusisataa tuntia työtä vuodessa. Nämä luvut ovat arvioita, eivät mitattua dataa, mutta ne antavat karkean mittaluokan, jossa ratkaisun liiketoiminnallinen merkitys alkaa konkretisoitua.

Pelkkä ajansäästö ei kuitenkaan ole ainoa vaikutus. Kanoninen tuotemalli ja integraatiot vähentävät virheiden mahdollisuutta esimerkiksi silloin, kun toimittajan kampanjat tai varastosaatavuus muuttuvat. Yhtenäisempi tuotetieto tukee myös myynnin puolta, kun sama tieto näkyy hinnoittelussa, verkkokaupassa ja sisäisessä ostonäkymässä. Tämä on linjassa sen kanssa, että digitalisoinnin odotetaan lisäävän läpinäkyvyyttä ja vähentävän riippuvuutta yksittäisten asiantuntijoiden hiljaisesta tiedosta (Omowole et al., 2024; Vial, 2019).

DSS-ajattelun kannalta ratkaisu vie yritystä kohti hybridimallia, jossa sääntöihin ja dataan perustuva moottori tuottaa tilausehdotuksia ja ostaja toimii ihminen silmukassa -periaatteella niiden hyväksyjänä. Tämä vastaa kirjallisuudessa kuvattua tapaa yhdistää automaation tehokkuus ja asiantuntijan harkinta ja konkretisoi, mitä päätöksenteon tuki tarkoittaa tietyn PK-yrityksen arjessa.

Samalla on hyvä todeta, että työn tulokset ovat vielä suunnitelmia. Sekä ajansäästölaskelmat että vaikutukset tuotetiedon laatuun ja hinnoitteluun pitää varmistaa pilottien ja seurantamittareiden avulla. Luvussa 8 kuvattavat jatkotoimet ja mittarit ovat keskeisiä sille, muuttuuko potentiaali todelliseksi hyödyksi.

7.4 Rajoitteet ja epävarmuustekijät

Tulokset perustuvat yhteen yritykseen ja sen nykyiseen tekniseen ympäristöön. Tämä tarkoittaa, että tilastollista yleistettävyyttä ei ole, mutta tuloksien siirrettävyys on mahdollinen vastaavissa PK-yrityksissä, joilla on samantyyppinen modulaarinen monoliitti ja tarve rakentaa sen ympärille rajattuja palveluja.

Richards ja Ford (2020) sekä Newman (2021) korostavat, että modulaarinen monoliitti on usein PK-yritykselle luontevin lähtökohta ja mikropalveluihin kannattaa siirtyä vasta, kun lisäarvo on selvästi perusteltu. Tässä työssä päädyttiin erilliseen mikropalveluun siksi, että ostoprosessin digitalisointi ja integraatiot muodostavat selkeästi oman kokonaisuutensa, jonka riippuvuudet ja kuormitus poikkeavat muusta järjestelmästä. Samalla päästiin rakentamaan uutta koodipohjaa nykyisen monoliitin rinnalle. Valinta on kirjallisuuden avulla perusteltavissa, mutta edellyttää, että CI/CD-putket, valvonta ja operointi saadaan PK-ympäristössä riittävälle tasolle (Pahl & Jamshidi, 2016; Taibi et al., 2017). Tutkijan arvion mukaan yrityksen digitaalinen kyvykkyys riittää tälle tasolle.

Toinen rajoite liittyy dataan ja tiedonhallinnan ohjaukseen (data governance). Työssä määriteltiin kanoninen tuotemalli ja integraatiot, mutta datan omistajuutta, rooleja ja pidemmän aikavälin hallintamalleja käsiteltiin vain rajatusti. Käytännössä ratkaisu onnistuu vain, jos joku vastaa siitä, että kanoninen malli pysyy ajan tasalla, PRICAT- ja INVRPT-aineistot tulkitaan oikein ja poikkeamat käsitellään systemaattisesti. Tämä puoli jää vielä selkeämmin jatkokehityksen tehtäväksi.

Lisäksi käyttäjäkokemuksen arviointi on tässä vaiheessa teoreettista. Ostajan näkökulmasta oleellisia asioita, kuten käyttöliittymän selkeys, ehdotusten luotettavuuden tulkinta ja arjen poikkeustilanteiden käsittely, ei ole testattu konkreettisesti. Ratkaisu näyttäytyy paperilla selkeältä, mutta sen onnistuminen riippuu lopulta siitä, onnistuuko käyttöönottoprojekti, koulutus ja muutoksen johtaminen. Työssä kuvattiin iteroivaa ja yhteistyössä tehtävää kehitystä, jonka varassa käyttäjäkokemusta voidaan parantaa vaiheittain.

7.5 Johtopäätökset menetelmän soveltuvuudesta PK-ympäristössä

Konstruktiivinen tapaustutkimus toimi PK-ympäristössä varsin hyvin. Se mahdollisti sen, että ratkaisua voitiin kehittää arjen työn ja havaintojen rinnalla ilman erillistä raskasta hanketta. Lähestymistapa sopii tilanteeseen, jossa kehitystyö on osa yrityksen normaalia toimintaa ja järjestelmä kehittyy vähitellen.

Samalla työ toi esiin aiemman lähestymistavan riskit. Käytössä ollut malli nojasi vahvasti yksittäisten asiantuntijoiden osaamiseen ja hiljaiseen tietoon, mikä lisää henkilöriippuvuutta. Tämä korostaa dokumentaation, versionhallinnan ja tiedonjakokäytäntöjen merkitystä. Kokonaisuutena voidaan todeta, että menetelmä ja valittu ratkaisu sopivat PK-yrityksen kehitysprojektiin, kunhan ne sidotaan selkeään tiekarttaan ja dataa, integraatioita sekä järjestelmälogiikkaa koskeva tieto tehdään näkyväksi. Näin vältetään se, että vastuu ja tieto jää yksittäisten henkilöiden varaan, ja parannetaan mahdollisuuksia viedä suunnitelmat käytäntöön myöhemmin luvussa 8 kuvatulla tavalla.

8. YHTEENVETO

Tämän työn tavoitteena oli suunnitella toteutuskelpoinen arkkitehtuuri, jolla Rengas Turku Oy voi digitalisoida ostoprosessia ja tuotetiedon hallintaa PK-yrityksen rajoitteet ottaen huomioon. Lähtökohtana oli hyvin käytännöllinen ongelma: toimittajatiedon pirstaleisuus, epätasainen eräaineistojen käsittely ja hinnanmuodostuksen vahva henkilöriippuvuus. Näiden ympärille rakennettiin tutkimuskysymys, jossa yhdistyivät manuaalityön vähentäminen, virheriskin pienentäminen, läpimenoajan lyhentäminen, tuotetiedon laadun parantaminen ja hinnoittelun tukeminen päätöksenteon tukijärjestelmän avulla.

Lähestymistapana oli konstruktiivinen tapaustutkimus, jossa kirjallisuuden ja käytännön havainnot kuljetettiin rinnakkain. Teoriaosassa koottiin keskeiset periaatteet tuotetiedon hallinnasta (PIM), päätöksenteon tukijärjestelmistä (DSS), PK-ympäristöön sopivasta ohjelmistoarkkitehtuurista sekä B2B-integraatiomalleista. Työn empiirisessä osassa ostoprosessi käytiin läpi pala kerrallaan ja havainnoista johdettiin ratkaisun vaatimukset. Ratkaisuvaihtoehdot pisteytettiin monikriteerisellä SAW-menetelmällä, ja teknistä toteutettavuutta testattiin rajatussa integraatiokokeilussa yhden toimittajan aineistolla. Näin lopputulos ei jäänyt pelkäksi konseptiksi.

Tuloksena syntyi kokonaisuus, joka yhdistää erillisen ostopalvelun, kanonisen tuotemallin, yhdistetyn push- ja pull-virran ja vaiheistetun etenemissuunnitelman. Kanoninen tuotemalli toimii käytännössä PIM-ratkaisun ytimenä: sama tuotetieto palvelee oston, hinnoittelun ja myynnin tarpeita sen sijaan, että tieto kiertää eri taulukoissa ja väliaikaisissa virityksissä. Integraatio on rakennettu kahdelle jalalle: eräsiirrot hoitavat massapäivitykset ja rajapinnat ajantasaiset kyselyt. Arkkitehtuurivalinnassa päädyttiin mikropalveluun, joka toimii rinnan olemassa olevan Django-pohjaisen modulaarisen monoliitin kanssa, mutta irrottaa oston ja integraatiot omaksi kokonaisuudekseen. Tämä on pieni poikkeama kirjallisuuden “monoliitti ensin, mikropalvelut myöhemmin” -periaatteesta, mutta tässä kontekstissa ratkaisu on perusteltu sekä teknisen rajauksen että kuormituksen kannalta.

Tässä työssä osoitettiin, miten ostoprosessin digitalisointia voidaan viedä eteenpäin askelittain, ei kertarysäyksellä. Ratkaisu tukee Aliyevin (2025) ja Omowole et al. (2024) kuvaamaa vaiheittaista digipolkua: ensin pureudutaan selkeimpiin kipupisteisiin (sesonkitilaukset, manuaalinen hinnoittelu, hajallaan oleva tuotetieto), ja vasta tämän jälkeen laajennetaan ratkaisua toimittajaverkoston ja DSS-toiminnallisuuksien suuntaan. Päätöksenteko toteutuu hybridimallina, jossa sääntöihin ja dataan perustuva moottori tuottaa

ehdotuksia ja ammattilainen toimii niiden viimeisenä sanana. Käytännössä tämä tarkoittaa, että järjestelmä hoitaa peruslaskennan ja datan yhdistelyn, mutta vastuu ja harkinta jäävät edelleen ostajalle.

Käytännön merkitys yritykselle näkyy ennen kaikkea työkuorman ja riskien hallinnassa. Työ osoittaa, että jo rajattu digitalisointi yhdellä kriittisellä osa-alueella voi vapauttaa merkittävän määrän ostopäällikön työajasta, vähentää manuaalisten virheiden mahdollisuuksia ja parantaa tuotetiedon läpinäkyvyyttä myynnille. Samalla tehdään näkyväksi se, mitä vielä puuttuu: jotkin tiedonhallinan käytännöt (data governance), useamman toimittajan pilotit, käyttäjäkokemuksen hiominen ja mitattavat tunnusluvut ovat seuraavan vaiheen tehtäviä. Kokonaisuutena diplomityö tarjoaa Rengas Turku Oy:lle konkreettisen, perustellun suunnitelman ostoprosessin ja tuotetiedon digitalisoinnille sekä esimerkin siitä, miten PK-yritys voi rakentaa omaan järjestelmäympäristöönsä sopivan PIM- ja DSS-tyyppisen ratkaisun vaiheittain, mutta määrätietoisesti.

LÄHTEET

Abraham, J. (2014). *Product information management: Theory and practice* (2014th ed.). Springer. https://doi.org/10.1007/978-3-319-04885-7

Aliyev, V. (2025). *Digital transformation strategies and challenges in small and medium enterprises (SMEs): A systematic review and future directions*. https://doi.org/10.13140/RG.2.2.27245.29924

Anta Callersten, J., Bak, S., Xu, R., Kalthof, R., & Bradley, S. (2024, April 16). *Overcoming retail complexity with AI-powered pricing*. Boston Consulting Group. https://www.bcg.com/publications/2024/overcoming-retail-complexity-with-ai-powered-pricing

Battistello, L., Haug, A., Suzic, N., & Hvam, L. (2021). Implementation of product information management systems: Identifying the challenges of the scoping phase. *Computers in Industry, 133*, 103533. https://doi.org/10.1016/j.compind.2021.103533

Continental. (n.d.). *Integration documentation (EASY XML, EDIWheel)* [Internal integration documentation, confidential; not publicly available].

DeSanctis, G., & Gallupe, R. B. (1987). A foundation for the study of group decision support systems. *Management Science, 33*(5), 589–609. https://doi.org/10.1287/mnsc.33.5.589

Díaz Andrade, A. (2009). Interpretive research aiming at theory building: Adopting and adapting the case study design. *The Qualitative Report, 14*(1), 42–60.

EDIWheel. (n.d.). EDIWHEEL [Website]. Retrieved April 2, 2025, from https://www.ediwheel.net

Ellram, L. M. (1995). Total cost of ownership: An analysis approach for purchasing. *International Journal of Physical Distribution & Logistics Management, 25*(8), 4–23. https://doi.org/10.1108/09600039510099928

Eugster, P. T., Felber, P. A., Guerraoui, R., & Kermarrec, A.-M. (2003). The many faces of publish/subscribe. *ACM Computing Surveys, 35*(2), 114–131. https://doi.org/10.1145/857076.857078

Fielding, R. T. (2000). *Architectural styles and the design of network-based software architectures* (Doctoral dissertation, University of California, Irvine). University of California eScholarship.

Gadde, N., Mohapatra, A., Dey, S., Das, I., Bhatia, V., & Reddy, G. (2024). Optimizing dynamic pricing through reinforcement learning: Techniques, case studies, and implementation challenges. *International Journal of Science and Research (IJSR), 13*(11), 159–165. https://doi.org/10.21275/SR241028211931

GS1. (2016/2021a). *EANCOM® 2002 S4 Part II — PRICAT Price/Sales Catalogue message* (Edition 2016, updated 2021). GS1 AISBL.

GS1. (2016/2021b). *EANCOM® 2002 S4 Part II — INVRPT Inventory report message* (Edition 2016, updated 2021). GS1 AISBL.

GS1. (2022). *EANCOM® 2002 Syntax 4, Part 1* (Release 1.3). GS1 AISBL.

GS1. (2023). *GTIN management standard* (Version 1.1). GS1 AISBL.

Hohpe, G., & Woolf, B. (2004). *Enterprise integration patterns: Designing, building, and deploying messaging solutions* (1st ed.). Addison-Wesley.

Holovaty, A., Kaplan-Moss, J., & Legassick, S. (2009). *The definitive guide to Django: Web development done right* (2nd ed.). Apress.

Irufaan, A., Vasudavan, H., & Singh, P. K. H. (2021). Microservice dynamic resource provision for small and medium-sized enterprises. *Journal of Applied Technology and Innovation, 5*(1), 15–20.

Khatri, V., & Brown, C. V. (2010). Designing data governance. *Communications of the ACM, 53*(1), 148–152. https://doi.org/10.1145/1629175.1629210

Kulkarni, U., Power, D. J., Sharda, R., & International Conference on Decision Support Systems. (2007). *Decision support for global enterprises* (1st ed.). Springer US. https://doi.org/10.1007/978-0-387-48137-1

Lukka, K. (2003). The constructive research approach. In L. Ojala & O.-P. Hilmola (Eds.), *Case study research in logistics* (pp. 83–101). Turku School of Economics.

National Institute of Standards and Technology. (2017). *Digital identity guidelines: Authentication and lifecycle management (SP 800-63B)* (includes updates as of 03-02-2020). https://doi.org/10.6028/NIST.SP.800-63b

Newman, S. (2021). *Building microservices: Designing fine-grained systems* (2nd ed.). O'Reilly.

Omowole, B. M., Olufemi-Phillips, A. Q., Ofodile, O. C., Eyo-Udo, N. L., & Ewim, S. E. (2024). *Barriers and drivers of digital transformation in SMEs: A conceptual analysis. International Journal of Scholarly Research in Science and Technology*, 5(2), 19–36. https://doi.org/10.56781/ijsrst.2024.5.2.0037

OWASP Foundation. (2023). *OWASP Application Security Verification Standard (ASVS) 4.0.3*. Retrieved October 7, 2025, from https://owasp.org/www-project-application-security-verification-standard/

Pahl, C., & Jamshidi, P. (2016). Microservices: A systematic mapping study. In *CLOSER 2016: Proceedings of the 6th International Conference on Cloud Computing and Services Science* (Vol. 1, pp. 137–146). https://doi.org/10.5220/0005785501370146

Power, D. J. (2002). *Decision support systems: Concepts and resources for managers*. Greenwood.

Ramakrishnan, J., Shaabani, E., Li, C., & Sustik, M. A. (2019). Anomaly detection for an e-commerce pricing system. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (pp. 1917–1926). https://doi.org/10.1145/3292500.3330748

Richards, M., & Ford, N. (2020). *Fundamentals of software architecture: An engineering approach* (1st ed.). O'Reilly.

Sommerville, I. (2016). *Software engineering* (10th ed.). Pearson.

Subramania, H. S., & Khare, V. R. (2011). Pattern classification driven enhancements for human-in-the-loop decision support systems. *Decision Support Systems, 50*(2), 460–468. https://doi.org/10.1016/j.dss.2010.11.003

Taibi, D., Lenarduzzi, V., & Pahl, C. (2017). Processes, motivations, and issues for migrating to microservices architectures: An empirical investigation. *IEEE Cloud Computing, 4*(5), Article 8125558. https://doi.org/10.1109/MCC.2017.4250931

Tzeng, G.-H., & Huang, J.-J. (2011). *Multiple attribute decision making: Methods and applications* (1st ed.). Chapman and Hall/CRC. https://doi.org/10.1201/b11032

Vial, G. (2019). Understanding digital transformation: A review and a research agenda. *The Journal of Strategic Information Systems, 28*(2), 118–144. https://doi.org/10.1016/j.jsis.2019.01.003

Wang, R. Y., & Strong, D. M. (1996). Beyond accuracy: What data quality means to data consumers. *Journal of Management Information Systems, 12*(4), 5–33. https://doi.org/10.1080/07421222.1996.11518099

LIITE 1. YRITYSPALAVERIT

Liite 1 alkaa seuraavalta sivulta.

Liite 1 - Yrityspalaverit

Tavoite: valita etenemispolku yrityksen ostotoiminnan digitalisointiin. Painopiste kahdessa keskustelussa on arkkitehtuurivalinnassa sekä siinä miten kriteerit ja painot perusteltiin.

Osallistujat: LH pääkehittäjä, MV ohjelmistovastaava, TK diplomityön tekijä
Toteutus: Etäpalaverit Google Meetissä

Kokous 1: toteutuskriteerien tulkinta ja painotuksen pohjustus

Aika: 2025-09-12 klo 13.00–14.20

Konteksti
Yrityksen tarpeet ja ongelmat ovat tiedossa. TK toi keskusteluun ennalta pohditut ratkaisuvaihtoehdot. Tavoitteena oli ymmärtää mikä yritykselle on tärkeää ja mihin suuntaan kriteerien painot viedään. Pisteytystä ei vielä tehty. Arkkitehtuuria ei vielä valittu. Varsinainen kriteeritaulukko ja painot koottiin TK:n toimesta palaverin jälkeen.

Vaihtoehdot
A: Django laajennuksena nykyiseen monoliittiin. Tuttu pino ja nopea alku
B: erillinen mikropalvelu integraatioineen ja omalla elinkaarella
C: kolmannen osapuolen työkalu esimerkiksi PIM tai ostotyökalu
D: jatketaan prosesseilla ilman uutta järjestelmää eli Excel ja nykykäytännöt

Keskustelun ydin

- Ylläpidettävyys ja ajankäyttö korostuivat. PK yrityksessä työntekijöitä on vähän ja LH kantaa vastaan teknisestä puolesta. Mitä vähemmän ylläpitoa sitä parempi. B nousi hyvänä suuntana, koska irtikytkettyä palvelua on helpompi huoltaa omana komponenttina. Samalla on huolehdittava että Nettivanne ja mikropalvelu pysyvät yhteensopivina. LH pitää A ratkaisua helppona, koska se on tuttu ja turvallinen.
- TK toi esiin mikropalvelun ylläpitoedut. Oma CI ja CD, versiointi ja rajattu vaikutusalue helpottavat pitkällä aikavälillä. Todettiin että nämä realisoituvat alkutyön jälkeen.
- Legacy huoli on todellinen. Nettivanne on paisunut monoliitiksi ja mukana on legacyä. B eriyttäisi ostotyökalun puhtaalle koodipohjalle ja pidentäisi elinkaarta.
- UX ketteryys on tärkeä. Ostosta vastaavat tarvitsevat selkeän ja nopean käyttöliittymän. B tarjoaa tuoreemmat työkalut, mutta LH korostaa, että A:llakin voidaan rakentaa asiallinen UI ja hyödyntää työkaluja.
- Skaalautuvuus painotettiin matalalle. Performanssiongelmia ei ole juuri ollut. Varauskalenteri on poikkeus, mutta ostotyökalu ei itsessään ole raskas.

- Kustannuksissa paino on pitkän aikavälin kokonaiskustannuksessa. MV totesi että kiinteää projektibudjettia ei määritetä, mutta kustannukset huomioidaan järkevästi.
- Tiedonhallinta ja PIM. Nettivanteessa tieto on koottu käytännöllisesti yhteen ilman varsinaista PIM määritelmää. TK:n näkemys oli että B mahdollistaa PIM ratkaisun. Se kokoaa tuotetiedon yhdenmukaiseksi kokonaisuudeksi ja helpottaa toimittajakohtaisen datan yhtenäistämistä. Yritys on ollut nykytilaan tyytyväinen, mutta PIM nähtiin tasonnostona pitkällä aikavälillä.
- C ja D eivät ole realistisia jatkoon. C ei istu kulttuuriin jossa suositaan omassa kontrollissa olevia ratkaisuja. Räätälöinnin tarve ja toimittajalukko huolettavat. D ei poista perusongelmia. Data jää hajalleen ja manuaalityö jatkuu.

Riskit tunnistettuina

Mikropalvelu

- integraatiopinta kasvaa
- operointikuorma
- valvonta ja hälytykset
- hajautetun järjestelmän konsistenssi
- schemojen versiointi
- testauksen monimutkaistuminen
- tiimin osaamisvaatimus
- DevOps ja infra
- alkuvaiheen työmäärä

Monoliitti

- legacy riippuvuus syvenee
- kytkentä kasvaa
- julkaisut ovat riskialttiimpia
- skaalautuvuus on rajallinen
- tekninen velka kasvaa
- regressioriski

Päätökset

- C ja D rajataan pois jatkoarvioinnista
- TK laatii kriteerit ja painot taulukoksi palaverin ja aiemman tiedon pohjalta

Vaikutus diplomityöhön

- Vertailu rajataan A ja B välille
- Kriteerit ja painot kootaan kasaan pääasiassa tämän palaverin tietojen pohjalta

---

Kokous 2: pisteiden esittely, arkkitehtuurivalinta ja teknologiat

Aika: 2025-09-30 klo 12.00–13.00

Konteksti
TK esitteli kriteerit, painot ja pisteet taulukkomuodossa. Tavoitteena oli hyväksyä kriteeristö ja painot sekä valita arkkitehtuurin päälinja. TK ehdotti samalla teknologiavalintoja ja varmisti että linja on LH:n ja MV:n mielestä toimiva.

Pisteet ja tulkinta
A Django laajennus 3,82 - B Mikropalvelu 4,22
Ero syntyi etenkin ylläpidettävyydestä ja elinkaaresta sekä UX kehityksen ketteryydestä. Skaalautuvuus vaikutti vähemmän, mutta oli B:n puolella. Herkkyystarkastelu vahvisti että B pysyy voittajana järkevillä painomuutoksilla.

Hyväksyntä
LH ja MV hyväksyivät kriteerit ja painot. Eriäviä mielipiteitä ei ollut.

Päätös
Valitaan B mikropalvelu arkkitehtuurin päälinjaksi. Perusteena strateginen suunta, uusi koodipohja ja pitkä elinkaari sekä paremmat mahdollisuudet rakentaa ostajalle hyviä työkaluja ja käyttöliittymä. Yrityksen näkökulmasta huoleksi jää toteutuksen vaikeustaso ja riskit. Nämä huomioidaan toteutuksessa.
A vaihtoehtoa voidaan käyttää varasuunnitelmana.

Teknologialinja
TK ehdotti kokonaisuuden. LH ehdotti backendiksi DRF ja TK hyväksyi sen.
Frontend: React ja TypeScript
Backend: DRF pohjainen mikropalvelu. TK:n alkuperäinen ehdotus oli FastAPI ja kevyt Python backend
Taustaprosessit: Celery ja Redis
Integraatioperiaate: selkeä sisäinen REST sopimus ja toimittajakohtaiset adapterit mikropalvelun reunalla
DevOps: erillinen CI ja CD

Yhteenveto
Pisteet tukivat B vaihtoehtoa. Ero oli niukka, mutta perusteltu. Päätös tehtiin pisteiden ja keskustelun perusteella. Teknologiat linjattiin. Mikropalvelu toteutetaan erillisenä ja sopimusohjattuna komponenttina. Varavaihtoehto A on käytettävissä, jos realiteetit sitä vaativat.