Tampereen yliopisto

Matias Leinonen

MOBIILIKÄYTTÖLIITTYMÄ MES-JÄRJESTELMÄÄN

TYÖNJOHTAJAN PERUSNÄKYMÄ

Informaatioteknologian ja viestinnän tiedekunta
Pro gradu -tutkielma
Lokakuu 2023

TIIVISTELMÄ

Matias Leinonen: Mobiilikäyttöliittymä MES-järjestelmään – Työnjohtajan perusnäkymä
Pro gradu -tutkielma
Tampereen yliopisto
Tietojenkäsittelyopin maisteriohjelma
Lokakuu 2023

Tuotantoteollisuusyritysten tuotantoprosessien tehostaminen on elintärkeää nykypäivän kilpaillussa markkinaympäristössä. Tuotannonohjausjärjestelmät (MES) parantavat merkittävästi tuotantoprosessien ohjauksen tehokkuutta järjestelmien tarjoamilla laatuominaisuuksilla. Monen laatuominaisuuden tarjonnasta huolimatta MES-järjestelmiä kehitetään jatkuvasti tarjoamaan yhä enemmän parannuksia tuotantoprosessien ohjaukseen.

Yksi MES-järjestelmän kehityskohteista on mobiililaitteiden hyödyntäminen tuotantoympäristössä. MES-järjestelmien käyttöä mobiililaitteiden kautta ei vielä nykypäivänä tueta laajasti. Vähäisen tuen taustalla on MES-järjestelmän tarjoaman suuren tietomäärän esittämisen hankaluudet pienillä näytöillä, tuotantoympäristöjen ainutlaatuiset toiminnallisuustarpeet sekä vakiintuneen mobiilistandardin puuttuminen.

Tutkimuksen tavoitteena oli luoda kohdeyrityksen MES-järjestelmään mobiilikäyttöliittymä käytettävyyden ja paikasta riippumattomuuden näkökulmista. Mobiilikäyttöliittymä tultiin tekemään pienimmän mahdollisen tuotteen (MVP) muodossa, jossa työnjohtajalle tehtäisiin mobiilinäkymä. Tutkimus toteutettiin suunnittelutieteen periaatteita noudattaen. Teoreettisissa luvuissa kartoitettiin ensin tutkimukseen tarvittava teoria, josta saadun tiedon pohjalta vastattiin tutkimukselle asetettuihin tutkimuskysymyksiin ja luotiin pohja konstruktion toteutukselle.

Tutkimuksen tuloksena luotiin mobiilikäyttöliittymä työnjohtajalle, jota voidaan käyttää yrityksen MES-järjestelmässä on- ja off-premises-ympäristöissä. Alkuperäisten vaatimusten lisäksi toteutettiin mobiilikäytettävyysparannuksia järjestelmän navigointinäkymiin, joiden tarkoituksena on helpottaa pääsyä uuteen työnjohtajanäkymään sekä parantaa järjestelmän mobiilikäyttökokemusta. Toteutuksessa käytetty Angular BreakpointObserver -moduuli osoittautui tehokkaaksi työkaluksi toteuttaa responsiivisen web-kehityksen mukaista näytönkoon seurantaa ja sen avulla dynaamisesti eri sisältöjen esittämistä Angularin ominaisuussitomisen kanssa. Mobiilikäyttöliittymässä käytettiin Angular Material -tyylikirjastoa, joka tarjosi sopivat elementit suunnitelmien mukaisen sisällön esittämiseen. Konstruktio täytti sille annetut vaatimukset ja loi kohdeyritykselle mahdollisen pohjan tulevalle mobiilikäyttöliittymän jatkokehitykselle.

Tutkimuksen tulos on yleistettävissä myös muihin web-sovelluksiin. Tulos on lähes kokonaan siirrettävissä web-sovelluksiin, joissa käytetään Angular-ohjelmistokehystä Angular Material -tyylikirjaston kanssa. Tutkimuksessa on myös yleispäteviä piirteitä, kuten käytettävyyden ja mobiilikäyttöliittymän suunnittelun käytännöt, joita voidaan hyödyntää myös muissa mobiilisovelluksissa, kuten alustakohtaisten mobiilisovellusten käyttöliittymäkehityksessä.

Avainsanat: MES, mobiilikäyttöliittymä, käytettävyys, web-sovelluskehitys, Angular

ABSTRACT

Matias Leinonen: Mobile user interface into MES system – Basic view for the foreman
Master of Science Thesis
Tampere University
Master's Degree Programme in Computer Science
October 2023

Improving the efficiency of production processes in manufacturing companies is vital for operating in today's competitive market environment. Manufacturing execution systems (MES) significantly improve the efficiency of the control of production processes with quality features provided by the systems. Despite already offering many quality features, MES-systems are constantly being developed to offer even more improvements for the control of production processes.

One of the development targets of a MES-system is the utilization of mobile devices in the production environment. The use of MES-systems with mobile devices is not yet widely supported. The low level of support is due to the difficulty of presenting large amounts of information provided by the MES-system on small screens, the unique functionality needs in production environments and the lack of an established mobile standard.

The goal of the research was to create a mobile user interface for the target company's MES-system from the viewpoints of usability and location independence. The mobile user interface was to be done as a minimal viable product (MVP) where a mobile view was to be done for a foreman. The research was done by following the principles of design science. In the theoretical chapters, theory needed for the study was first mapped, from which the information obtained was used to answer research questions set for the study and to create a basis for the implementation of the construction.

As a result of the research, a mobile user interface was developed for the foreman, which can be used in the company's MES-system in on- and off-premises environments. In addition to the original requirements, mobile usability improvements were implemented to the system's navigation views, the purpose of which was to ease the ability to access the new foreman view and improve the mobile user experience of the system. Angular BreakpointObserver module used in the implementation proved to be an effective tool to implement responsive web development's screen size monitoring and using it to dynamically present different content along with Angular's property binding. Angular Material style library was used in the mobile user interface, that provided matching elements for presenting content according to design plans. The construction met the requirements set for it and created a possible foundation for the company's future mobile user interface implementation.

The result of the research can also be generalized to other web applications. The result is almost entirely transferable to web applications, where Angular framework is used together with Angular Material style library. The research also has general traits, such as usability and mobile user interface design practices, that can also be used in other mobile applications, such as in user interface development of native mobile applications.

Keywords: MES, mobile user-interface, usability, web application development, Angular

ALKUSANAT

Haluan kiittää Leanware Oy:tä mahdollisuudesta tehdä lopputyöni kiinnostavasta sekä konkreettisesta aiheesta. Kiitos työpaikan ohjaajat Pekka Saarelainen ja Ville Harju nopeista vastauksista, avusta ja kattavan kokemustiedon tarjoamisesta työhön. Kiitos myös yliopiston ohjaajat ja tarkastajat Timo Poranen ja Antti Sand erinomaisesta ohjauksesta, kommenteista ja parannusehdotuksista. Kiitos myös ystävilleni ja työkavereilleni kannustuksesta ja tuesta lopputyöni teon ja opiskeluni aikana. Kiitos vielä perheelleni korvaamattomasta tuesta koko opiskelujeni ajan.

Tampereella, 8.10.2023

Matias Leinonen

SISÄLLYSLUETTELO

LYHENTEET JA MERKINNÄT

AI	Tekoäly (Artificial Intelligence)
AJAX	Asynkroninen JavaScript ja XML (Asynchronous JavaScript And XML)
API	Ohjelmointirajapinta (Application Programming Interface)
CRUD	Luo, lue, päivitä, poista (Create Read Update Delete)
DTO	Tiedonsiirto-objekti (Data Transfer Object)
ERP	Toiminnanohjausjärjestelmä (Enterprise Resource Planning)
GUI	Graafinen käyttöliittymä (Graphical User Interface)
IoT	Esineiden internet (Internet of Things)
JSON	Avoimen standardin tiedostomuoto (JavaScript Object Notation)
KPI	Keskeiset suoritusindikaattorit (Key Performance Indicators)
MES	Tuotannonohjausjärjestelmä (Manufacturing Execution System)
MESA	Tuotantoyritysratkaisujen yhdistys (Manufacturing Enterprise Solutions Association)
ML	Koneoppiminen (Machine Learning)
MOM	Valmistustoimintojen hallinta (Manufacturing Operations Management)
OEE	Tuotantolaitteiston kokonaistehokkuus (Overall Equipment Effectiveness)
PLC	Ohjelmoitava logiikkaohjain (Programmable Logic Controller)
PWA	Progressiivinen web-sovellus (Progressive Web Application)
REST	Ohjelmointirajapinta-arkkitehtuurityyli (Representational State Transfer)
RWD	Responsiivinen web-suunnittelu (Responsive Web Design)
SCADA	Valvomo-ohjelmisto (Supervisory Control And Data Acquisition)
SPA	Yhden sivun web-sovellus (Single Page Application)
SQL	Standardoitu tietokantakyselykieli (Structured Query Language)
UX	Käyttökokemus (User Experience)
VPN	Virtuaalinen erillisverkko (Virtual Private Network)

1. JOHDANTO

Tuotantoteollisuudessa tuotteiden valmistusprosessiin liittyy järjestelmiä, jotka tuottavat suuria määriä tietoja tuotannon tapahtumista. Yksi tällainen järjestelmä on tuotannonohjausjärjestelmä (Manufacturing Execution System, MES). Suuren tietomäärän muotoileminen ymmärrettävään muotoon on merkittävää yrityksen tehokkuuden kannalta. Tietojen analysoinnista voidaan saada arvokasta tietoa tuotantoprosessin toiminnasta ja tehokkuudesta. (Kaczmarczyk et al. 2022) Erityisesti kojelauta- (*dashboard*) ja raportointiratkaisut ovat kasvattaneet suosiotaan tuotantoympäristöissä niiden valvonta- ja päätöksenteon tukiominaisuuksien ansiosta (Allen et al. 2021).

Monille tuotantoyrityksille, jotka eivät pysty keräämään tarpeeksi pääomaa siirtyäkseen täysin digitaaliseen tuotantoon, jäävät epävarmuuden tilaan, koska he eivät pysty tehostamaan toimintaansa täydellä potentiaalillaan. Monille yrityksille apuna tehokkuuden parantamisessa ovat erilliset visualisointi- ja analytiikkaohjelmistot. (Allen et al. 2021)

Yksi kehityskohde MES-järjestelmissä on mobiilikäyttöliittymien ja mobiililaitteiden hyödyntäminen. Mobiililaite mahdollistaa järjestelmän käytön käyttäjän paikasta riippumatta. Mobiililaitteiden käytössä kuitenkin kohdataan haasteita, kuten tiedon esittämisen rajoitukset pienellä näytöllä sekä tiedon turvallisen saatavuuden varmistaminen sijainnista riippumatta.

Tämän tutkimuksen tavoitteena on suunnitella ja toteuttaa mobiilikäyttöliittymä työnjohtajan perusnäkymästä kohdeyrityksen MES-järjestelmään. Mobiilikäyttöliittymä toteutetaan pienimmän toimivan tuotteen muodossa (MVP), jonka tavoitteena on luoda pohja MES-järjestelmän muiden osien tulevaa mobiilitoteutusta varten. Mobiilikäyttöliittymän toteutuksessa keskitytään käytettävyyden ja paikasta riippumattomuuden näkökulmiin.

Tutkimus tehdään kohdeyritykselle, joka tuottaa ja ylläpitää sovellusjärjestelmäratkaisuja tuotannonohjaukseen, varastonhallintaan, sisälogistiikkaan sekä oston ja hankinnan prosesseihin. Tämä tutkimus keskittyy yrityksen tuottamaan ja ylläpidettävään MES-järjestelmään.

Kohdeyritys on tunnistanut tarpeen kehittää MES-järjestelmän mobiilikäytettävyyttä ja paikasta riippumattomuutta. Eräät kohdeyrityksen asiakkaista ovat vuosien saatossa esittäneet halukkuutta käyttää MES-järjestelmää älypuhelimilla. Kuitenkin kohdeyrityk-

sen MES-järjestelmä ei tue järjestelmän käyttöä älypuhelimilla riittävästi. Esimerkiksi mobiilikäytettävyydessä on puutteita ja paikasta riippumattoman käytön turvallisuus ei ole varmistettu. Tutkimuksessa luotavan MVP:n tavoitteena on parantaa nykyisen järjestelmän mobiilikäytettävyyttä työnjohtajan perusnäkymän muodossa, josta käytettyjä tekniikoita voidaan hyödyntää myös järjestelmän muissa osissa. Lisäksi pyritään selvittämään, miten järjestelmää ja luotavaa mobiilikäyttöliittymää voidaan käyttää turvallisesti paikasta riippumatta.

1.1 Tutkimusmenetelmä ja tutkimuskysymykset

Tutkimuksessa tultiin käyttämään suunnittelutieteellistä (*design science*) tutkimusmenetelmää. Suunnittelutiede valittiin tutkimuksen soveltuvaksi tutkimusmenetelmäksi, koska tutkimuksessa tultiin luomaan ratkaisu kohdeyrityksen konkreettiseen ongelmaan. Suunnittelutieteessä ongelman ratkaisu on konstruktio, joka voi olla esimerkiksi konkreettinen tuote tai toimintamalli. Tässä tutkimuksessa konstruktiona toimii toteutettava mobiilikäyttöliittymä ja siihen liittyvät vaatimukset. Konstruktion suunnittelussa ja toteutuksessa hyödynnetään kirjallisuudessa olemassa olevaa teoreettista tietoa samalla kun uutta tietoa syntyy tuotoksena ongelmanratkaisutyöstä. (Hevner et al. 2004)

Tutkimuksen tutkimuskysymykset ovat:

1. Mitä tutkimusta on tehty MES-järjestelmissä käytetyistä mobiilikäyttöliittymistä?
2. Miten voidaan luoda mobiilikäyttöliittymä MES-järjestelmään?
3. Miten voidaan käyttää mobiilikäyttöliittymää turvallisesti paikasta riippumatta?

1.2 Tutkimuksen vaiheet ja rakenne

Peffersin ja muiden (2007) mukaan suunnittelutieteellisen tutkimuksen prosessi jakautuu kuuteen vaiheeseen:

1. Ongelman tunnistaminen
2. Tavoitteiden asettaminen
3. Suunnittelu ja toteutus
4. Ratkaisun todentaminen
5. Arviointi
6. Viestintä

Ensimmäisessä vaiheessa tunnistetaan käytännön ongelma, joka tullaan tutkimuksessa ratkaisemaan. Toisessa vaiheessa määritetään ratkaisun tavoitteet ja vaatimukset, jotka

pyritään saavuttamaan toteutuksessa. Kolmannessa vaiheessa suunnitellaan ja toteutetaan ongelman ratkaisu eli konstruktio. Neljännessä vaiheessa pyritään esittämään, miten luotu konstruktio voi ratkaista ongelman. Viidennessä vaiheessa arvioidaan, miten hyvin konstruktio ratkaisee alkuperäisen ongelman. Kuudennessa vaiheessa viestitetään tutkimuksen prosessin ja tuloksien yleistettävyydestä sekä hyödyllisyydestä. (Peffers et al. 2007)

Tässä tutkimuksessa vaiheet jakautuvat tutkimuksen rakenteen eri lukuihin. Ensimmäisen vaiheen ongelman tunnistaminen esitettiin johdannossa, jossa kuvattiin kohdeyrityksen ongelma MES-järjestelmän nykytilan avulla. Toisen vaiheen tavoitteiden asettaminen esitettiin keskeisten tavoitteiden osalta johdannossa ja luvussa 7 esitetään tarkemmin empiirisen toteutuksen tavoitteet ja vaatimukset.

Kolmannen vaiheen ongelman ratkaisemisen suunnittelu ja toteutus esitetään luvuissa 2, 3, 4, 5, 6 ja 7. Teoreettisissa luvuissa 2, 3, 4 ja 5 kartoitetaan kirjallisuudesta tarvittavia tietoja tutkimuskysymysten ratkaisemiseen. Luku 2 keskittyy tuotannonohjaukseen ja ensimmäisen tutkimuskysymyksen vastaamiseen. Luvut 3 ja 4 vuorostaan käsittelevät käytettävyyttä, web-kehittämistä ja mobiilikäyttöliittymän suunnittelua, joista saaduilla tiedoilla pyritään vastaamaan toiseen tutkimuskysymykseen. Luku 5 käsittelee MES-järjestelmän asennusmuotojen paikasta riippumattomuuden käyttöä, josta saadulla tiedolla vastataan kolmanteen tutkimuskysymykseen. Luku 6 esittää yleiskuvan ja teknisen kuvauksen tutkimuskohteena olevasta MES-järjestelmästä, johon konstruktio tullaan toteuttamaan. Luku 7 keskittyy empiirisen toteutuksen dokumentointiin, jossa esitetään konstruktion varsinainen suunnittelu- ja toteutusprosessi.

Neljännen vaiheen ratkaisun todentaminen esitetään luvun 7 lopussa tehdyssä konstruktion testauksessa. Viidennen vaiheen tuloksen ja tutkimuksen arviointi esitetään luvussa 8 tavoitteiden saavuttamisen todentamisella. Luvussa 8 myös esitetään kuudennen vaiheen tutkimuksen tuloksen viestintä. Tutkimuksen lopuksi esitetään yhteenveto tutkimuksesta luvussa 9.

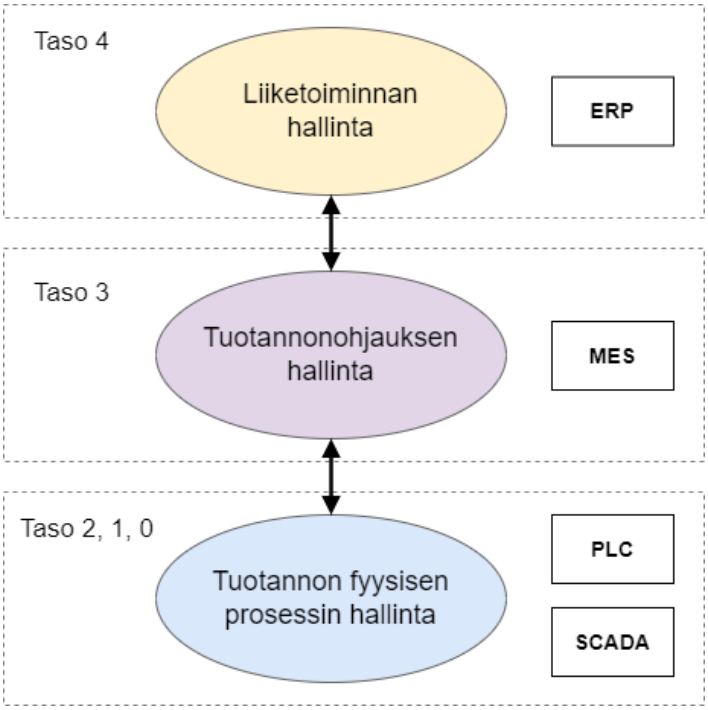
2. TUOTANNONOHJAUS

Tässä luvussa esitetään tuotannonohjauksen määritelmät, joita tarkastellaan vakiintuneiden ja suosiotaan kasvattavien määritelmien avulla. Tavoitteena on luoda tausta tutkimuksen käyttökontekstiin, jotta ymmärretään mihin kohtaan tuotantoprosessissa mobiilikäyttöliittymää suunnitellaan toteutettavaksi.

Kohdassa 2.1 esitetään tuotantoprosessin ohjauksen vakiintunut määritelmä ISA-95-standardin avulla. Kohdassa 2.2 esitetään tuotantoprosessin ohjauksessa suosiotaan kasvattava määritelmä Teollisuus 4.0 (Industry 4.0) sekä näkyvyyden määritelmä. Kohdassa 2.3 esitetään tuotannonohjausjärjestelmän määritelmä ja nykypäivän kehityssuunnat. Kohdassa 2.4 esitetään työnjohtajan tehtävät ja rooli tuotannonohjausjärjestelmän käytössä.

2.1 Tuotantoprosessin ohjaus

Tuotantoprosessilla tarkoitetaan kaikkia vaiheita ja resursseja tuotantotilauksen luomisesta tuotteen valmistukseen asti. Tuotantoprosessin ohjaamisen hallinta ja toiminnot jakautuvat organisaation eri osa-alueille. ISA-95-standardi jakaa osa-alueet viiteen eri tasoon kuvan 1 mukaisesti. (ISA-95 2005; Gröger et al. 2016)



Kuva 1. *ISA-95-standardin tasot, roolit ja esimerkkijärjestelmät mukaillen lähteistä (ISA-95 2005; Jaskó et al. 2020; Harju 2023a; Rix et al. 2016).*

Taso 4 koostuu tuotantoprosessin liiketoiminnan tehtävistä, kuten henkilöstö- ja materiaaliresurssien hallinnasta, tuotantotilausten karkeasuunnittelusta ja varaston tiedonhallinnasta. Näitä tehtäviä voidaan hallinnoida toiminnanohjausjärjestelmistä, kuten ERP-järjestelmistä, jotka toimivat strategisen johtamisen tukitietojärjestelminä. (ISA-95 2005; Rix et al. 2016)

Tason 3 tuotannonohjauksen hallinnan tehtävät koostuvat esimerkiksi tuotantotilausten hienokuormituksesta, tuotannon seurannasta ja ohjaamisesta sekä tuotteiden jäljitettävyydestä. Tason 3 toimintoja ja niitä toteuttavia järjestelmiä kutsutaan myös käsitteellä MOM (Manufacturing Operations Management). Tuotannonohjausjärjestelmät, kuten MES-järjestelmät, mahdollistavat tuotannonohjauksen hallinnan tehtävien toteuttamisen sekä toimivat toiminnallisena tuotteiden valmistuksen ja ohjauksen kerroksena toiminnanohjausjärjestelmien ja fyysisten prosessiohjausjärjestelmien välillä (ISA-95 2005; SAP 2023; Jaskó et al. 2020).

Tasot 0–2 käsittävät fyysisten tuotanto- ja automaatiotason laitteiden prosessien hallinnan. Tuotantolaitteiden sisäisten tai ulkoisten antureiden ja sensoreiden avulla saadaan tietoa fyysisestä tuotantotilanteesta esimerkiksi ohjelmoitaviin logiikkaohjaimiin (PLC) ja valvontajärjestelmiin (SCADA). PLC:n tapaisten ohjaimien avulla MES-järjestelmä voi saada lähes reaaliaikaista tietoa laitteiden tuotantotilasta. (ISA-95 2005; Rix et al. 2016)

Tässä tutkimuksessa keskitytään yllä esitettyyn ISA-95-standardin tasoon 3 ja MES-järjestelmään, johon tutkimuksen mobiilikäyttöliittymä toteutetaan. ISA-95-standardi on toiminut vakiintuneena tuotantoprosessin hierarkiamallina yli kahden vuosikymmenen ajan, mutta nykypäivänä uudet mallit, kuten Teollisuus 4.0, ovat kasvattaneet suosiotaan merkittävästi. Seuraavassa luvussa käsitellään Teollisuus 4.0 -määritelmää ja siihen liittyvää näkyvyyden määritelmää, joka on olennainen osa tätä tutkimusta.

2.2 Teollisuus 4.0 ja näkyvyys

Teollisuus 4.0 tarkoittaa teollistumisen neljättä vallankumousta, jonka keskiössä toimii digitaalisen tehtaan käsite. Digitaalisen tehtaan konseptissa tehtaan tuotantokoneet ja laitteet ovat yhdistettynä antureiden ja langattomien yhteyksien avulla yhtenäiseksi järjestelmäksi. Konseptissa keskeistä on koneiden ja laitteiden välinen kommunikaatio, jonka takia konsepti usein esitetään verkkorakenteisena arkkitehtuurina ISA-95-standardista poiketen. (Jaskó et al. 2020; Allen et al. 2021)

Teollisuudessa tuotteiden valmistukseen keskittyville yrityksille digitaalisen tehtaan käyttöönotto onnistuu parhaiten pitkällä aikavälillä, kun pieniä toiminnallisuuksia toteutetaan inkrementaalisesti olemassa olevaan järjestelmään. Tämä tuottaa jatkuvaa toiminnallista

hyötyä tehokkuuden lisääntymisen ja investointikustannusten jakautumisen ajan myötä. (Jaskó et al. 2020)

Digitaalisen tehtaan käyttöönoton yksi ensimmäisistä askelista on näkyvyyden (*visibility*) lisääminen tuotantoprosessiin. Näkyvyyden lisäämistä pidetään tärkeänä ponnahduskivenä digitaalisen tehtaan käyttöönotossa, sillä se mahdollistaa tuotantoprosessin reaaliaikaisen seurannan ja analysoinnin, mitkä ovat merkittäviä menestystekijöitä digitaalisen tehtaan toteutumisessa. (Allen et al. 2021) Tuotantoympäristössä näkyvyydellä tarkoitetaan näkyvyyttä tuotantoprosessin tilaan. Esimerkiksi tietyn tuotantokoneen tai resurssin tilaa voidaan tarkastella yksityiskohtaisella tasolla, josta voidaan nähdä reaaliaikaisesti työn alla oleva tuotantotilaus, tuote tai erä. Lisäksi voidaan tarkastella tulevan työjonon tuotantotilauksien sekä työvaiheiden ajoituksia.

Graafiset käyttöliittymät (GUI) ovat ensiarvoisen tärkeitä jokaisessa ihmisten käyttämässä järjestelmässä, mikä pätee erityisesti järjestelmiin, jotka ohjaavat monimutkaisia teollisia prosesseja, kuten MES-järjestelmät (Jaskó et al. 2020). Digitaaliset kojelaudat toimivat näkyvyyden tiedon esittämisessä käyttäjälle. Kojelautojen tavoitteena on esittää intuitiivisia ja helppokäyttöisiä käyttöliittymiä tuotantoprosessin toimintojen seurantaan, joiden avulla käyttäjät eri organisaation osa-alueilta pystyvät tehostaa toimintaansa. ERP-järjestelmissä kojelautoja käytetään liiketoimintaprosessien seurantaan ja analysointiin. MES-järjestelmissä kojelaudat sekä näyttöpäätteet esittävät arvokasta tietoa tuotannon nykyisestä tilasta työnjohtajille sekä tuotantovaiheista ja tehtävistä lattiatason operatiivisille tuotantotyöntekijöille. Automaatiotason järjestelmissä kojelaudat voivat esimerkiksi esittää mittaustietoa fyysisiltä tuotantolaitteilta PLC-ohjaimien kautta SCADA-järjestelmään. (Gröger et al. 2016; Rix et al. 2016)

Kuva 2. *Esimerkkikojelauta työnjohtajille ja tuotantopäälliköille (Tokola et al. 2016).*

Tokola ja muut (2016) toteuttivat tutkimuksessaan katsauksen kojelautoihin tuotantoteollisuudessa. Kuvan 2 työnjohtajille ja tuotantopäälliköille toteutettu kojelauta toteutettiin kyselyn tuloksien perusteella, missä pyrittiin selvittämään mitä henkilöt tuotantoyrityksen eri työtehtävissä haluavat ja tarvitsevat nähdä kojelaudan näkymissä. Kuvan kojelaudassa esitetään tietoja tuotantoprosessin tilasta päivä-viikko-tarkasteluajanjaksolla, kuten tuotantolaitteiden käyttöasteesta prosentteina, tuotantolaitteiston kokonaistehokkuudesta (OEE) prosentteina, tuotantotilausten läpimenoajoista päivinä, toimitusvarmuudesta prosenttina, työlinjojen tehokkuudesta prosentteina ja tuotereklamaatioiden määrä numerona ja vertausluku edelliseen mittaukseen prosenttina.

Allenin ja muiden (2021) mukaan yritykset, jotka haluavat siirtyä kevyemmin digitaaliseen valmistukseen, panostaminen tuotantoprosessin näkyvyyteen digitaalisen kojelaudan avulla voi mahdollistaa arvokkaan prosessinäkemyksen paljon tehokkaammin kuin täysi digitalisaatio. Kevyellä siirtymisellä tarkoitettaisiin esimerkiksi IoT (Internet of Things) -laitteiden ja antureiden hyödyntämistä havaitsemaan ja keräämään tietoa tuotantoprosessin tapahtumista. Tokolan ja muiden (2016) mukaan kuitenkin panostaminen tuotantoyrityksen toimintaa hyödyntävään kojelautaan voi olla käytännössä kallis investointi, jos yrityksen käytössä ei ole entuudestaan ERP- tai MES-järjestelmää. Tietovarasto, johon on yhdistettynä ERP- ja MES-järjestelmistä saatavat tiedot, helpottaa merkittävästi ajallisesti sekä rahallisesti kojelaudan käyttöönotossa.

2.3 Tuotannonohjausjärjestelmä ja nykyiset kehityssuunnat

Tuotannonohjausjärjestelmä on tietojärjestelmä, jonka avulla voidaan ohjata ja optimoida valmistusprosessien tehokasta toteutumista tuotantotilauksen luomisesta tuotteen toimitukseen asti. MES-järjestelmän yhtenä päätarkoituksena on välittää keräämäänsä tietoa tuotantoprosessin vaiheiden tapahtumista ERP-järjestelmien ja tuotannon automaatiotason järjestelmien välillä. Kerätyn tiedon avulla henkilöt lattiatason operatiivista tuotantotyöntekijöistä liiketoiminnan päätöksentekijöihin pystyvät tehostamaan ja optimoimaan tuotantoa jatkuvasti. (Jaskó et al. 2020; Kaczmarczyk et al. 2022; Leanware 2023; SAP 2023)

Yli 25 vuotta sitten vuonna 1997 tuotantoyritysratkaisujen yhdistys MESA (Manufacturing Enterprise Solutions Association) määritteli MES-järjestelmän keskeiset toiminnot MESA-11-mallissaan. Pitkästä käyttöajasta huolimatta MESA-11-malli on säilyttänyt asemansa MES-järjestelmän ydintoimintojen esittämisessä ja perustana lähes kaikentyyppisille tuotannonohjausta havitteleville organisaatioille. MESA-11-mallin toiminnot ovat kuvattuna seuraavasti (MESA 1997; SAP 2023):

- *Resurssienhallinta*: Resurssien, kuten tuotantokoneiden, materiaalien ja nimikkeiden tilaa ja tietoja voidaan seurata ja hallita.
- *Toimintojen yksityiskohtainen ajoitus:* Tuotettavien yksiköiden tuotantovaiheiden ja tehtävien ajoittamisen hallinta ja optimointi prioriteettien ja resurssikapasiteetin perusteella.
- *Tuotteiden seuranta ja jäljitettävyys:* Tuotettujen yksiköiden tarkkoja jäljitettävyys- ja historiatietoja voidaan tarkastella jälkikäteen yksityiskohtaisesti.
- *Asiakirjojen hallinta:* Asiakirjojen, kuten työohjeiden, piirustusten ja reseptien hallinta ja kohdistus niitä koskeville resursseille.
- *Tiedon keruu:* Tuotantoprosessin joka tapahtumasta, kuten tehtäväkuittauksista, vaihekuittauksista ja häiriöraporteista, voidaan kerätä tietoa. Tiedon keruu mahdollistaa tuotannon läpinäkyvyyden.
- *Tuotantoyksiköiden hallinta ja aikataulutus:* Tuotettavien yksiköiden liikettä voidaan hallita eri resurssien, kuten tuotantovaiheiden ja tuotantokoneiden välillä. Tämä mahdollistaa reaaliaikaisen tuotannon optimoinnin ja tehostamisen.
- *Työnjohto:* Työnjohtajat voivat hallita ja seurata työjonojen tehokasta toteutumista sekä operatiivisten tuotantotyöntekijöiden kapasiteettia, läsnäoloa, työtaitoja ja työtehokkuutta.

- *Tuotantoprosessin hallinta:* Koko tuotantoprosessin hallinta tuotantotilauksen vapautuksesta tuotteen valmistumiseen. Pullonkaulojen ja tuotteiden laatua haittaavien prosessitoimintojen hallitseminen ja korjaus.
- *Suorituskykyanalyysi:* Tuotantoprosessista saatujen tietojen, kuten tuotantotilausten läpimenoaikojen tuloksia verrataan aiempiin tuloksiin ja tavoitteisiin. Analyysistä saatuja tietoja voidaan hyödyntää järjestelmien ja prosessien tehostamisessa.
- *Laadunhallinta:* Laatutiedon kerääminen tuotantoprosessista esimerkiksi poikkeamien ja häiriöiden avulla. Laatutiedon analysoinnin avulla voidaan kartoittaa ja parantaa tuotteiden laatua.
- *Kunnossapidon hallinta:* Tuotantokone- ja laiteongelmien tunnistaminen ennen niiden ilmenemistä huolto- ja vikahistorioiden avulla. Näiden tietojen avulla voidaan paremmin ennustaa ja lyhentää laitteiden seisokkiaikoja.

Jaskó ja muut (2020) toteuttivat laajan tutkimuksen Teollisuus 4.0 -konseptista ja miten se erityisesti tulee koskemaan MES-järjestelmän kehitystä. Tutkimuksessa esitettiin MES-järjestelmän toimintojen nykyiset kehityssuunnat, jotka koostettiin kahdeksasta Teollisuus 4.0:ta ja MES-järjestelmää käsittelevistä tieteellisistä tutkimuksista ja konferenssijulkaisuista vuosilta 2017–2020. Esitetyt kehityssuunnat ovat (Jaskó et al. 2020):

- *Resurssienhallinta:* Resurssien vaihdettavuuden kehittäminen ja tuotantotilausten valmistelu ja toteuttaminen oikeilla resursseilla.
- *Tuotteiden seuranta ja jäljitettävyys:* Yksittäisten tuotteiden tarkempi seuranta, kuten raaka-aineiden seurannan lisääminen.
- *Asiakirjojen hallinta:* Tuotteen valmistustavan tarkempi määrittely materiaaliluetteloiden, tuotantosääntöjen, reseptitietojen ja prosessin asetusarvojen avulla.
- *Tiedon keruu:* Pilvipohjaisten teknologioiden hyödyntäminen suuren määrän tiedon tallentamisessa.
- *Suorituskykyanalyysi:* Analyysin tehostaminen tekoälyn (AI) ja koneoppimisen (ML) avulla.
- *Tuotantoyksiköiden hallinta ja aikataulutus:* Mikroaikataulutuksen kehittäminen eli resurssien optimaalisen käytettävyyden varmistaminen ja aikataulutus.
- *Visualisointi:* Panostaminen selkeisiin ja helposti ymmärrettäviin graafisiin käyttöliittymiin, jotka toimivat tuotantoprosessin tietojen välittämisessä eri organisaation tasoilla työskentelevien henkilöiden yhteistyön tukemiseksi.

Grögerin ja muiden (2016) mukaan MES-järjestelmät ovat yleensä toteutettu käytettäviksi kiinteiden työasemien näyttöpäätteiden kautta, eivätkä nykyiset MES-järjestelmät juurikaan tue käytettävyyttä mobiililaitteilla, kuten älypuhelimilla ja tableteilla. InTech-tieteellisessä lehdessä (2019) julkaistussa ISA-101-komitean artikkelissa mainitaan, että mobiililaitteiden, kuten älypuhelimien ja tablettien ihmisen ja koneen välisten käyttöliittymien (Human Machine Interface, HMI) tutkiminen tuotantoprosessiympäristössä on tällä hetkellä käynnissä. ISA-101-komitea luo standardeja, suositeltuja käytäntöjä ja teknisiä raportteja ihmisen ja koneen välisistä käyttöliittymistä tuotannonohjaussovelluksissa. Vakiintuneen standardin puuttuminen mobiililaitteiden käytöstä tuotantoprosessiympäristössä saattaa olla yksi mahdollinen syy siihen miksi mobiililaitteiden käyttöä ei vielä tueta MES-järjestelmissä niin laajasti. Nykyisten MES-järjestelmän kehityssuuntien ja digitaalisen tehtaan vaatimuksien puolesta mobiililaitteiden tiedonvälityksellä on keskeinen rooli tuotantoprosessin joustavuuden ja tehokkuuden edistämisessä (Gröger & Stach 2014; Kaczmarczyk et al. 2022).

Gröger ja muut (2016) toteuttivat tutkimuksessaan mobiililaitteelle suunnitellun tieto- ja analytiikkapohjaisen tuotantoprosessin hallintapaneelin. Hallintapaneeli suunniteltiin operatiivisten tuotantotyöntekijöiden ja työnjohtajien käyttöön. Hallintapaneelin tavoitteena oli parantaa tuotantoprosessin toimintojen suorituskykyä ja ketteryyttä hyödyntämällä MES-järjestelmän tarjoamaa valtavaa tietomäärää. Tutkimuksen käytännön tapauksen tuloksena hallintapaneelin todettiin lisäävän tuotantoprosessin läpinäkyvyyttä ja toimintojen ketteryyttä sekä mahdollistavan tuotantoprosessin optimoinnin ja tiedon lisäämisen lattiatason operatiivisille tuotantotyöntekijöille ja työnjohtajille.

2.4 Työnjohtajan rooli MES-järjestelmän käytössä

Tuotannonohjauksessa työnjohtajalla tarkoitetaan henkilöä, jonka vastuuna on tietyn tuotantoprosessin seuranta ja toteuttaminen. Tietyn tuotantoprosessin toteutumisella voidaan tarkoittaa esimerkiksi yksittäisen tuotantokoneen, työpisteen, tuotantolinjan tai työvuoron työnjonon valmistumista. Työnjohtaja myös seuraa, valvoo sekä hallitsee tuotantoprosessia toteuttavia operatiivisia tuotantotyöntekijöitä, esimerkiksi MESA-11-mallin mukaan työntekijöiden kapasiteettia, läsnäoloa, työtaitoja ja työtehokkuutta. (Gröger et al. 2016)

Työnjohtajan rooliin MES-järjestelmän käytössä sisältyy myös tuotemuutosten ajoituksen hallinta. MES-järjestelmän avulla työnjohtaja voi tehokkaasti ajoittaa eri työpisteiden työjonoja siten, että ne sopivat tuotteen osien valmistusprosessiin. Esimerkiksi jos tuotetaan suurta tuotetta, jonka valmistamiseen tarvitaan useita komponentteja monelta eri

vaiheelta, joidenkin vaiheiden aloittaminen voi riippua edellisten vaiheiden valmistumisesta. MES-järjestelmä auttaa toisistaan riippuvien vaiheiden ajoittamisessa siten, että eri työpisteet voidaan ajoittaa tekemään samanaikaisesti tai tietyssä järjestyksessä samalle tuotteelle meneviä tehtäviä ja siten mahdollistaen seuraavien vaiheiden ajoittamisen sujuvasti. (Leanware 2023)

Tehtäviensä suorittamiseen työnjohtajilla on hyvä olla tapa seurata ja hallita tietyn tuotantoprosessin toteutumista. Tätä tarkoitusta varten työnjohtajalla on tärkeää olla käytössään työkalut, joiden avulla voi tarkkailla tuotannon tapahtumia ja toimintoja. Tuotantoprosessin hallintanäkymä tarjoaa kattavan kuvan tuotannon tilanteesta, jonka avulla työnjohtaja voi tehokkaasti seurata ja hallita prosessin etenemistä. (Gröger et al. 2016) Mantravadin ja muiden (2020) mukaan digitaaliset kojelaudat ja näyttöpäätenäkymät, jotka toteuttavat tuotantoprosessin hallintanäkymiä, ovat erityisen hyödyllisiä työnjohtajille ja muille johtohenkilöille.

Tuotantoprosessin tehokas toteutuminen on tärkeä osa teollisen yrityksen liiketoiminnan kannattavuutta, jota voidaan mitata suorituskykyanalyysien avulla. Suorituskyvyllä viitataan tietoon, kuinka hyvin, tehokkaasti tai muulla ennakkoon määrätyllä tavoiteltavalla tavalla tapahtuma on suoriutunut. MES-järjestelmän keräämän tuotantotiedon avulla voidaan analysoida tuotantoprosessin suorituskykyä. Yksi tärkeimmistä suorituskyvyn arvioinnin menetelmistä ovat keskeiset suoritusindikaattorit (KPI). Näitä ovat esimerkiksi tuotantotilausten läpimenoajat, tuotannosta ilmenneet poikkeamat ja häiriöt sekä tuotannon läpinäkyvyys. (Gröger et al. 2016)

Tuotantotilauksen läpimenoajalla tarkoitetaan kulunutta aikaa tuotantotilauksen aloittamisen ja lopetuksen välillä. Läpimenoaika on tärkeä indikaattori, jonka avulla sekä työnjohtajat että liiketoiminnan päätöksentekijät pystyvät tekemään tuotantoprosessia tehostavia muutoksia (Gröger et al. 2016). Työnjohtaja pystyy läpimenoajan avulla tarkastelemaan yksityiskohtaisesti tuotantoprosessin eri vaiheiden suorituskykyä, esimerkiksi operatiivisten tuotantotyöntekijöiden päivävuoron tehokkuutta verrattuna yövuoroon. Liiketoiminnan kannalta tuotantoprosessin läpimenoaika on tärkeä indikaattori, koska se kertoo tuotantoprosessin aikaosuuden tuotantotilauksen kokonaisajasta. Tuotantotilauksen kokonaisajalla tarkoitetaan tilauksen saapumisajan ja tilatun tuotteen lähettämisajan välistä aikaa. (Leanware 2023) Kokonaisaika on tärkeä mittari myös tuotantoyrityksen asiakkaiden näkökulmasta, koska nykypäivän markkinaympäristössä viiveaikojen pienentäminen on erittäin tärkeää yrityksen tehokkuuden ja kannattavuuden näkökulmista (Mantravadi et al. 2020).

Poikkeamilla tarkoitetaan tuotteen valmistuksessa ilmenneitä vikoja, puutteita tai ongelmia. Häiriöillä vastaavasti tarkoitetaan esimerkiksi tuotantokoneiden toiminnassa esiintyviä vikoja ja häiriötilanteita. Poikkeama- ja häiriötietojen tuoman jäljitettävyystiedon avulla operatiiviset työntekijät ja työnjohtajat voivat kartoittaa tuotteiden ja koneiden laadullisuutta ja luoda dokumentaatiota vikojen ennaltaehkäisemiseksi tulevaisuudessa. (Gröger et al. 2016; LeanwareMES 2023; Leanware 2023)

Tuotannon läpinäkyvyydellä tarkoitetaan tuotannon tilannekuvan helposti saatavuutta, minkä mahdollistaa MES-järjestelmän keräämä tuotantotieto. Tuotannon tilannekuva kuvaa tuotannon kokonaistilannetta sisältäen esimerkiksi tietoja tietyn asiakkaan tilauksen tilasta tai päivän suunnittelun työjonon jäljellä olevasta työmäärästä. Tuotantotieto koostuu tuotannon tapahtumista kerätyistä tiedoista, kuten valmistuskuittauksista, kunnossapidon poikkeamista ja häiriöistä sekä materiaalivirroista. MES-järjestelmässä tuotantotiedon visualisointi kuvien ja kaavioiden avulla luo läpinäkyvyyden tuotantoon, joka auttaa tuotantoprosessin osapuolia ymmärtämään paremmin tuotannon kokonaistilannetta sekä tehostamaan tuotannon toimintaa tulevaisuudessa. (Leanware 2023; Harju 2023b)

Läpinäkyvyyden avulla voidaan myös kehittää päivittäisjohtamista (*daily management*). Päivittäisjohtamisella tarkoitetaan toiminnan jatkuvaa parantamista ja ongelmien ratkaisemista yhteistyössä työntekijöiden ja esimiesten kanssa. Päivittäisjohtamisessa korostetaan tiivistä yhteistyötä ja avointa kommunikointia työntekijöiden ja esimiesten välillä toiminnan tehostamiseksi sekä parantamiseksi. (Zarbo et al. 2015) Esimerkiksi Zarbon ja muiden (2015) tutkimuksessa päivittäisjohtamisen avulla saatiin selville, että kirurgisen patologian laboratorion prosessin lopputyövaiheissa lisääntynyt työmäärä johtui edellisten työvaiheiden käytännöistä. MES-järjestelmän tarjoaman tuotannon läpinäkyvyyden yhdessä päivittäisjohtamisen kanssa voi mahdollistaa vastaavia tuloksia tuotannonohjauksen tehokkuuden parantamisessa.

3. KÄYTETTÄVYYS JA MODERNIT KÄYTTÖLIITTYMÄT

Nykypäivänä graafisia käyttöliittymiä käytetään yhä enemmän ja ne muodostavat merkittävän osan ihmisen päivittäistä työympäristöä. Usein myös käyttöliittymien loppukäyttäjät arvioivat sovelluksen laatua yksinomaan heille näkyvän käyttöliittymän perusteella. Siksi käyttöliittymien suunnittelijoiden kannattaa panostaa käytettävyyden vahvistamiseen, koska hyvä käytettävyyden aste luo kuvan laadukkaasta tuotteesta ja toimii merkittävänä etuna muita kilpailevia sovelluksia vastaan. Hyvä käytettävyys myös vähentää opastuksen tarvetta ja auttaa käyttäjää oppimaan uuden tuotteen käytön helpommin käyttöönottovaiheessa. (Niemelä 2020)

Tässä luvussa perehdytään käytettävyyden ja käyttöliittymien määritelmiin. Kohdassa 3.1 esitetään käytettävyyden määritelmä. Kohdassa 3.2 käsitellään responsiivisen websuunnittelun kehitystapa, jota tullaan hyödyntämään tutkimuksen konstruktiossa. Kohdassa 3.3 ja sen alakohdissa käsitellään mobiilikäyttöliittymän rakenteen ja sisällön suunnittelua, joita myös hyödynnetään tutkimuksen empiirisessä osiossa.

3.1 Käytettävyys

Käytettävyys voidaan määritellä monella eri tapaa ja se kohdistuu eri käyttötapauksiin. Tietoteknisestä näkökulmasta käytettävyys kohdistuu sovellusten ja järjestelmien käyttöön ihmisen ja koneen välisten käyttöliittymien kautta. Käytettävyys voi myös kohdistua minkä tahansa tuotteen, kuten sähköhellan käyttöön. (Niemelä 2020)

ISO 9241-11 (2018) -standardin mukaan käytettävyys määritetään ”missä määrin tietyt käyttäjät voivat käyttää järjestelmää, tuotetta tai palvelua tiettyjen tavoitteiden saavuttamiseen tuottavalla, tehokkaalla ja tyytyväisellä tavalla tietyssä käyttöyhteydessä”. Standardin kuvauksessa esiintyy huomionarvoisia piirteitä. Standardi korostaa, että käytettävyyttä ei voida määritellä yksiselitteisesti kaikille käyttäjille, vaan sen määrittely riippuu tiettyjen käyttäjien, tavoitteiden ja käyttöyhteyksien huomioimisesta. Tietyillä käyttäjillä tarkoitetaan yksittäisiä kohderyhmiä tai valittuja käyttäjäkuntia, jotka tulevat käyttämään tuotetta. Käyttäjät on tunnettava ominaisuuksien, kuten iän ja ammatin perusteella, jotta tuotteen käytettävyys voidaan optimoida käyttäjien tarpeita vastaaviksi. Käyttäjien tarpeet määritellään tiettyjen tavoitteiden avulla, joita käyttäjät pyrkivät saavuttamaan tuot-

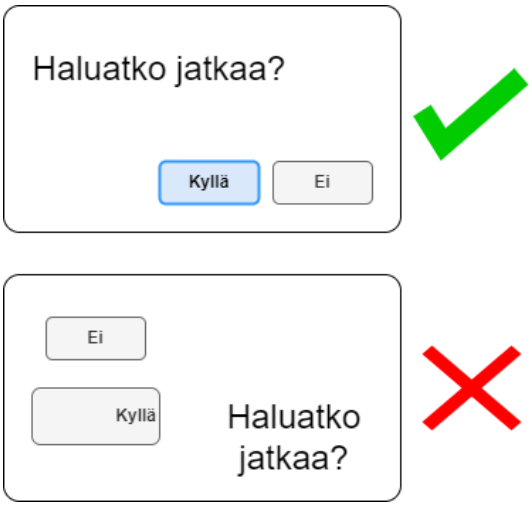
teen käytöstä. Tietyllä käyttöyhteydellä tarkoitetaan tuotteen käyttöympäristöä ja -olosuhdetta missä tuotetta käytetään. Esimerkiksi tehtaan lattiatasolla käytetty näyttöpääte voi tarvita erilaisen käyttöliittymän kuin toimistoympäristössä.

Standardin kuvauksessa esitetään myös laatumittarit, joita käytettävyydessä tavoitellaan (ISO 9241-11 2018; Niemelä 2020):

- Tuottavuus (*effectiveness*): Miten hyvin käyttäjä saavuttaa tavoitteensa?
- Tehokkuus (*efficiency*): Miten nopeasti käyttäjä suorittaa tehtävän?
- Tyytyväisyys (*satisfaction*): Miten miellyttävää on käyttäminen?

Nielsen (2012) määrittelee käytettävyyden tuotteen laatuominaisuudeksi, joka arvioi käyttöliittymän helppokäyttöisyyttä. Nielsenin mukaan käytettävyys on keskeinen tekijä tuotteen hyödyllisyydessä ja hyväksyttävyydessä. Käytettävyyden kontekstissa hyödyllisyydellä viitataan tuotteen toimivuuteen eli tekeekö se sen mitä käyttäjät tarvitsevat. Nielsen on myös laajentanut ISO 9241-11 -standardin tavoitteellisia laatumittareita opittavuuden, muistettavuuden ja virheettömyyden käsitteillä (Nielsen 2012):

- Opittavuus (*learnability*): Kuinka helppoa on tehtävien suorittaminen ensimmäisellä käyttökerralla?
- Muistettavuus (*memorability*): Kuinka helposti käyttäjä muistaa tuotteen käyttämisen tauon jälkeen?
- Virheettömyys (*errors*): Kuinka monta virhettä käyttäjät tekevät ja kuinka helposti he voivat toipua virheistä?



Kuva 3. *Yksinkertainen esimerkki käytettävyyden parantamisesta käyttöliittymäsuunnittelussa mukaillen lähteestä (Wikimedia Commons 2007).*

Kuvassa 3 esitetään esimerkki käytettävyyden parantamisesta käyttöliittymäsuunnittelussa. Laatikot esittävät käyttöliittymän valintaikkunoita (*dialog*), jossa pyydetään käyttäjän syötettä. Alempi valintaikkuna on merkitty punaisella rastilla, sillä se ei täytä hyvän käytettävyyden astetta. Kyllä- ja Ei-painonapit ovat epätasapainossa, koska ne ovat erikokoisia sekä niiden ohjetekstit ovat sijoiteltu eri kohtiin nappien sisällä. Lisäksi valintaikkunan kysymysteksti on epäloogisessa paikassa, koska se ei ole vasemmassa yläkulmassa mistä usein aloitetaan käyttöliittymän käyttö. Nämä käytettävyyspuutteet johtavat muistettavuuden vähenemiseen, sillä käyttöliittymän loogisuutta ei seurata. Käytettävyyspuutteet voivat johtaa myös virheiden tapahtumiseen, mikä vähentää käyttöliittymän virheettömyyttä.

Yhteenvetona käytettävyys tarkoittaa tuotteen oppimisen, muistamisen ja käyttämisen tehokkuutta. Tuotteen toimintalogiikka täytyy olla helposti ymmärrettävissä ja antaa tukea virheistä toipumiseen. Tuote saavuttaa erinomaisen käytettävyyden asteen, kun käyttäjä voi keskittyä tehtävänsä suorittamiseen ilman, että työvälineeseen tarvitsee kiinnittää erityistä huomiota. (Niemelä 2020)

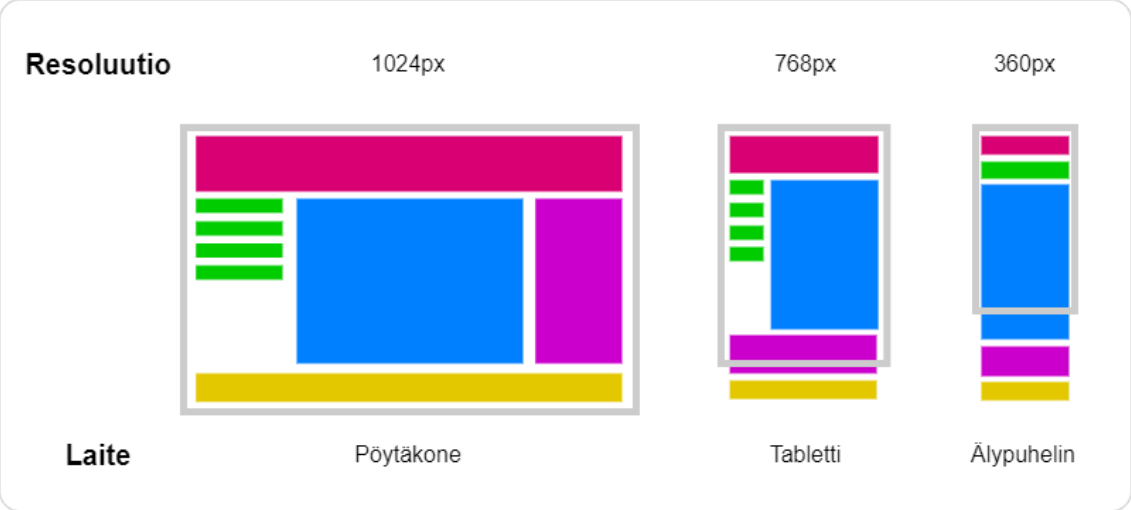
3.2 Responsiivinen web-suunnittelu

Responsiivinen web-suunnittelu (RWD) on verkkosivuston kehitystapa, jonka tavoitteena on sivuston ulkoasun automaattinen mukauttaminen eri laitteiden näyttökoiden ja resoluutioiden mukaan. RWD mahdollistaa optimaalisen käyttökokemuksen (UX) ja käytettävyyden riippumatta siitä käytetäänkö sivustoa esimerkiksi pieneltä älypuhelimen näytöltä tai isolta pöytäkoneen näytöltä. (Schade 2014)

Yksi RWD:n suurimmista hyödyistä on yhden sivuston luominen tukemaan käyttäjiä erikokoisilla näytöillä. Tällöin jokaiselle laitteelle ei tarvitse luoda erillistä koodikantaa ja käyttöliittymää. Tämä nopeuttaa kehitystä ja helpottaa ylläpitoa, kun keskitytään vain yhden sivuston kehittämiseen monen sijaan. RWD on myös tulevaisuudenkestävä suuntaus, sillä sen mukainen koodi tukee laajasti myös uusia näyttökokoja. Jos uusi näyttökoko ei ole tuettu, sen tukeminen on pienempi muutos RWD:ssä kuin kokonaan uuden erillisen käyttöliittymän luominen. Tämä mahdollistaa sivustojen joustavan käytön tulevaisuudessa, vaikka uusia näyttökokoja tulisi käyttöön. (Schade 2014)

RWD:tä hyödyntämällä sivustolla näkymän kasvaessa tai pienentyessä sivuston käyttöliittymäelementit liikkuvat ja järjestyvät näyttökoolle sopivaksi (Schade 2014). Käyttöliittymäelementeillä tarkoitetaan komponentteja, joiden avulla käyttäjä voi käyttää käyttöliit-

tymää. Käyttöliittymäelementtejä ovat esimerkiksi painonapit ja tekstikentät, joita voidaan ryhmitellä säiliöiden (*container*) avulla. (Usability.gov 2023) Kuva 4 havainnollistaa miten käyttöliittymäelementit järjestyvät eri laitteiden näyttökoiden resoluutioilla.



Kuva 4. *Käyttöliittymäelementtien mukautuminen eri laitteiden näyttökoiden resoluutioilla mukaillen lähteistä (W3Schools 2023; Procházka 2012; Schade 2014).*

Kuvasta 4 nähdään, miten käyttöliittymäelementit mukautuvat eri laitteiden näyttökokojen resoluutioille. Kuvan laatikot esittävät elementtejä tai säiliöitä, joita voi esiintyä verkkosivustolla. Punainen laatikko esittää ylätunnistetta (*header*), vihreät laatikot navigointiluetteloa (*navigation list*), sininen laatikko leipätekstiä eli sisältöä (*content*), violetti laatikko tunnisteluetteloa (*tag list*) ja keltainen laatikko alatunnistetta (*footer*).

Pöytäkoneen näkymä esitetään kolmisarakkeisena mallina, johon mahtuu vierekkäin kolme eri elementtiä. Tabletin näkymässä navigointiluettelo kapenee ja sisältö valtaa tunnisteluettelon tilan, joka siirtyy alatunnisteen päälle. Tabletin tilasta myös huomataan, että tunnisteluettelo ja alatunniste ovat osittain tai kokonaan tabletin näytön ulkopuolella. Tämä kuvastaa sitä, että kaikki elementit eivät ole heti näkyvissä käyttäjälle, kuten pöytäkoneen näkymässä, vaan käyttäjän täytyy vierittää tabletin näkymää alaspäin, jotta näkisi loput sivuston sisällöstä. Älypuhelimen näkymässä navigointiluettelo on siirtynyt ylätunnisteen alapuolelle, sisältö on vallannut loput näytöstä ja tunnisteluettelo sekä alatunniste ovat kokonaan näytön ulkopuolella.

Schade (2014) painottaa sisällön priorisoimisesta RWD:ssä. Sisällön priorisoimisella tarkoitetaan, että kaikkia elementtejä ei voida pakottaa mahtumaan pienille näytöille, vaan on hyödyllistä kartoittaa mitä sisältöä tulisi priorisoida näytettäväksi käyttäjälle sivustoa avattaessa, ja mitä voidaan jättää näytön ulkopuolelle.

RWD:ssä sama toimintalogiikka toimitetaan suoritettavaksi kaikille laitteille. Tabletit ja älypuhelimet eivät omaa samantasoista suorituskykyä kuin pöytäkoneet, jolloin saman

toimintalogiikan suorituksesta voi syntyä suorituskykyongelmia, kuten hitaita latausaikoja, jotka vaikuttavat negatiivisesti käytettävyyteen. Schade (2014) korostaa, jotta voidaan arvioida RWD:n todellista käyttökokemusta, täytyy toteutusta testata vaihtelevissa olosuhteissa, kuten paikoissa, joissa verkkoyhteydet ovat epätasaisia.

3.3 Mobiilikäyttöliittymä

Ihmisen ja koneen välisten käyttöliittymien suunnittelu on nykypäivänä yhä tärkeämpi suuntaus, kun käyttöliittymien käyttö on jatkaa yhä kasvuaan ihmisten jokapäiväisessä elämässä. Käyttöliittymien on pystyttävä vastaamaan käyttäjän odotuksiin ja tavoitteisiin, muuten käyttäjä menettää luottamuksen käyttöliittymään ja pahimmillaan lopettaa sen käytön (Nielsen 2012). Tuotantoympäristössä ja prosessiautomaatiossa hyödylliset käyttöliittymät ovat tärkeitä tuotannon luotettavuudelle, turvallisuuden parantamiselle ja projektikustannusten vähentämiselle (O'Brien 2023).

Tässä kohdassa käsitellään mobiilikäyttöliittymien suunnittelua ja sisältöä. Alakohdassa 3.3.1 esitetään mobiilikäyttöliittymän suunnittelun parhaita käytäntöjä ja alakohdassa 3.3.2 käsitellään tärkeimpiä tekijöitä, jotka on otettava huomioon mobiilikäyttöliittymän näkyvän sisällön osalta.

3.3.1 Suunnittelu

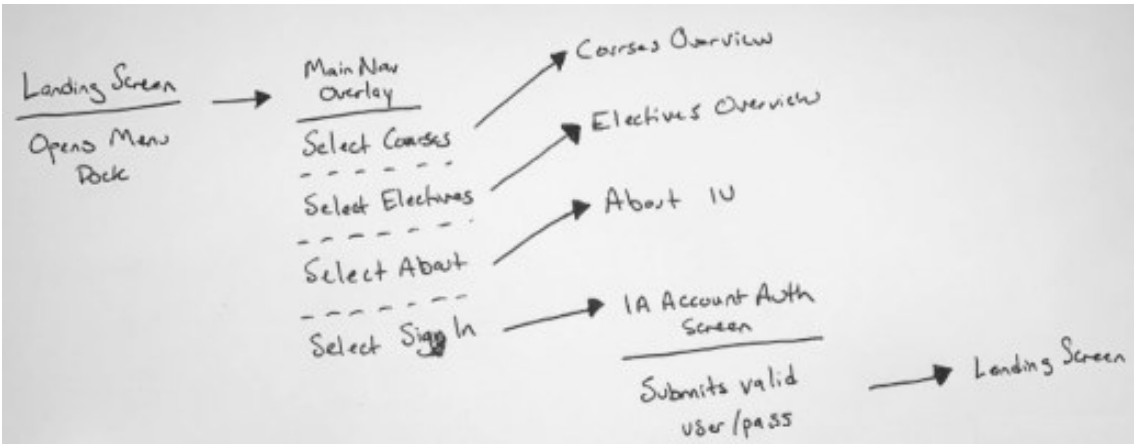
Mobiilikäyttöliittymien suunnittelun parhaat käytännöt ovat nykypäivänä kehittyneitä. Kuitenkin mobiilikäyttöliittymien suunnittelu ja kehitys tuotanto- ja prosessiympäristöissä ovat edelleen suurelta osin tutkimattomia ja määrittämättömiä. Tuotanto- ja prosessiympäristöt sisältävät ainutlaatuisia toiminnallisuustarpeita ja ratkaistavia ongelmia, erityisesti mobiilikäyttöliittymien osalta. Nämä hidastavat mobiilikäyttöliittymien kehitystä näissä ympäristöissä. (Sensenbach 2022) Toinen mahdollinen syy määrittämättömyyteen voi olla vakiintuneen standardin puuttuminen, kuten ISA-101-komitean parhaillaan työstämän teknisen raportin puuttuminen, joka mainittiin kohdassa 2.3.

Sensenbach (2022) esittelee Inductive Automation -teollisuusautomaatio-ohjelmistoratkaisujen organisaation yhteisökonferenssin seminaarissa parhaita käytäntöjä ja ohjeita mobiilikäyttöliittymien suunnittelussa. Sensenbach jakaa suunnitteluprosessin kolmeen osaan: kulkukaavioiden (*user flow diagram*), rautalankamallien (*wireframe*) ja prototyyppien tekemiseen.

Kulkukaavio on kuva, joka esittää käyttäjän kulun sovelluksen eri näyttöjen lävitse. Prosessin mahdollistaman nopean iteroinnin avulla voidaan tunnistaa navigoinnin umpikujat

sekä muut valintahaasteet. Kulkukaavioita voidaan myös luoda yksittäisille näytöille kuvaamaan kaikkia käyttäjän mahdollisia valintoja.

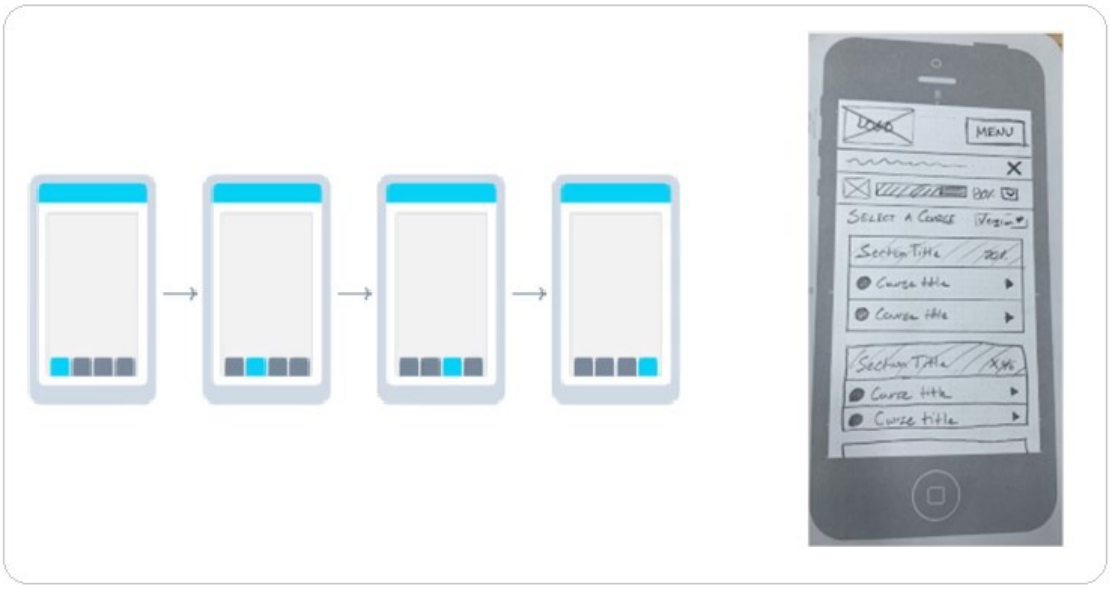
Kuvassa 5 esitetään esimerkki kulkukaaviosta. Kulkukaavio esitetään vaakatasossa vasemmalta oikealle, jossa jokainen nuoli esittää siirtymää eri näyttöjen välillä. Kulkukaavion näytöt ovat hyödyllistä luoda mallilla, missä alkutilassa esitetään minkä näytön käyttäjä näkee ja sen alapuolella viivalla erotettuna mitä käyttäjä voi valita. Nuolilla ilmaistaan siirtymät seuraaville mahdollisille näytölle. Esimerkiksi kuvassa aloitetaan käyttäjän kulku aloitussivulta *(landing screen*), josta käyttäjä valitsee telakkavalikon (*dock menu*). Telakkavalikon avaamisesta avautuu päänavigointinäkymä (*main nav overlay*), josta käyttäjä voi edetä monille eri näytöille. (Sensenbach 2022)



Kuva 5. *Kulkukaavioesimerkki (Sensenbach 2022).*

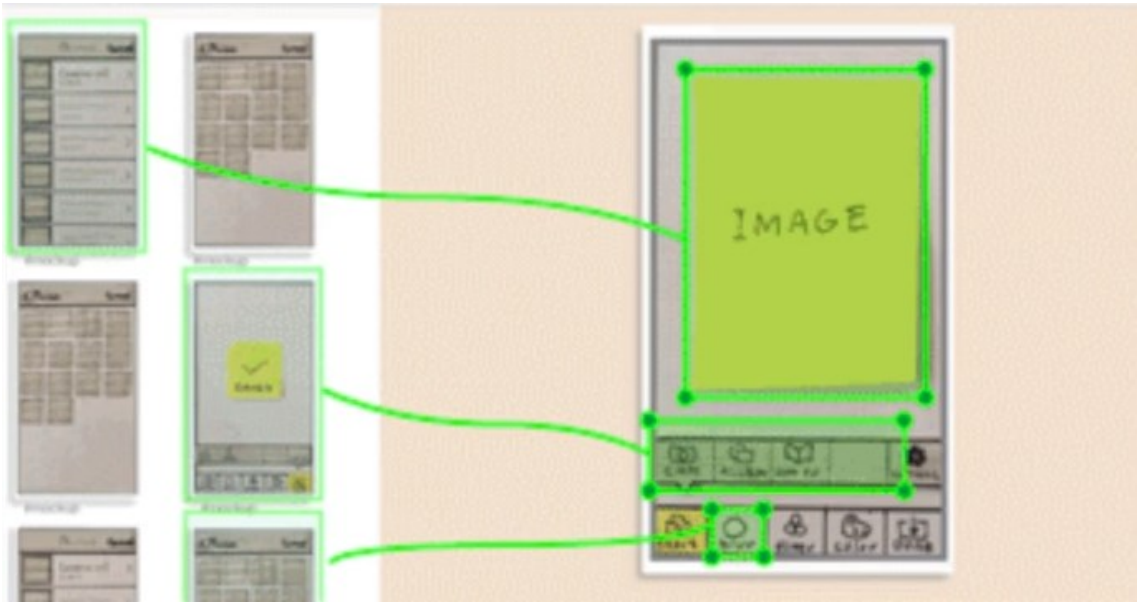
Rautalankamallien avulla voidaan luoda karkea luonnos sovelluksen käytettävyydestä. Kulkukaavioiden tapaan, rautalankamallien luonnissa nopean iterointia hyödynnetään ideoiden ja korjausehdotusten esille tuomiseen. Nopea iterointi on myös kustannustehokas tapa mahdollistaa yhteistyö suuremman tiimin kanssa.

Kuvassa 6 esitetään esimerkki rautalankamallista. Kuvan vasemmanpuoleisessa välilehtinäkymässä (*tabbed view*) kuvataan kulkukaavion tapaista rakennetta, jossa jokainen älypuhelin esittää omaa näkymäänsä. Älypuhelimien näyttöjen harmaalle alueelle on tavoitteena luoda kuvan oikeanpuoleinen käyttöliittymänäkymä kuvaamaan jokaista näkymää. Käyttöliittymän luonnissa on hyödyllistä keskittyä käyttöliittymäelementtien sijoitteluun ja sisällön priorisoimiseen. Aikaa ei kannata käyttää elementtien yksityiskohtaisten ikonien valitsemiseen tai nimien keksimiseen, koska rautalankamalleissa pyritään hyödyntämään sen nopean iteroinnin prosessia ideoiden luomiseen. (Sensenbach 2022)



Kuva 6. *Rautalankamalliesimerkki (Sensenbach 2022).*

Prototyypillä tarkoittaa suunnittelun tuloksena saatua ensimmäistä versiota. Prototyyppi voidaan esittää interaktiivisena, jossa käyttäjä voi konkreettisesti käyttää ja testata suunniteltua tuotetta. Kuvan 7 mukaisesti paperiset rautalankamallit voidaan skannata käytettäväksi tietokoneelle, joihin voidaan lisätä interaktiivisia ominaisuuksia, kuten napin painamisesta navigoidaan toiselle näytölle. Tämäntyyppistä prototypointia kutsutaan paperiprototypoinniksi *(paper prototyping*), joka on nopea ja hyödyllinen tapa simuloida mobiilikäyttöliittymän käyttökokemusta. (Sensenbach 2022)

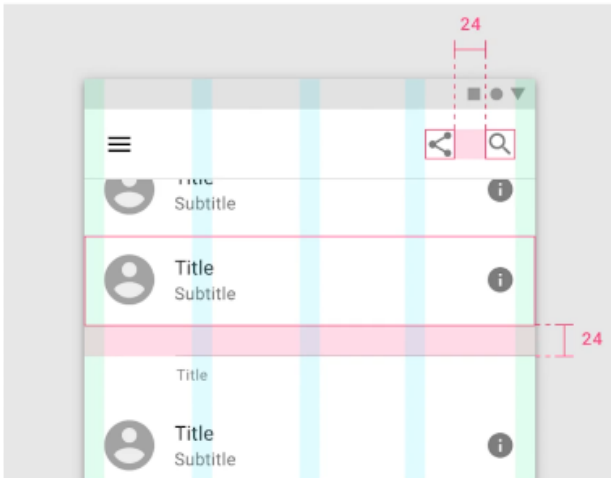


Kuva 7. *Paperiprototypointiesimerkki (Sensenbach 2022).*

3.3.2 Sisältö

Mobiilikäyttöliittymäsuunnittelussa sisällön priorisointi on yksi keskeisimmistä tekijöistä onnistuneen käyttöliittymän luomisessa. Budiun (2015) mukaan sisällön suunnittelu älypuhelimille on lähes kaksi kertaa vaikeampaa verrattuna suuremmille pöytä- ja kannettavien tietokoneiden näytöille. Sensenbach (2022) sanoo, että käyttäjillä on kyky pitää työmuistissaan vain 5–7 tiedonpätkää kerrallaan. Näiden syiden takia on tärkeää esittää sisältö muistettavalla tavalla, jotta käyttäjä voi helposti tunnistaa käyttöliittymän toiminnot sen sijaan, että joutuisi muistamaan ne. Sisällön priorisoimisen ja muistettavuuden avulla voidaan vähentää käyttöliittymän käyttöön vaadittua tajunnallista kuormitusta (*cognitive load*). (Sensenbach 2022; Budiu 2015) Se mitä käyttöliittymällä priorisoidaan, saadaan selville käyttäjien tarpeista ja käyttäjäkontekstista (Budiu 2015).

Käyttöliittymän sisältöstrategialla tarkoitetaan mobiilikäyttökokemuksen vahvistamista keskittymällä vastaamaan kysymyksiin: mitä tietoa käyttäjä etsii ja mitä tehtäviä tarvitaan tiedon saavuttamiseen. Johdonmukainen käyttöliittymä ja visuaalinen selkeys ovat tärkeitä tiedon etsimisen tehostamisessa. Yhdenmukaisten värien, fonttien ja ikonien käyttö läpi sovelluksen auttaa käyttöliittymän tunnistettavuudessa. Myös oikean maailman vertaiskuvien, kuten suurennuslasikuvakkeen hakukentässä, käyttö on tehokas tapa ilmaista käyttöliittymän toiminnallisuutta. Lisäksi tärkein tieto tulisi näyttää ensin käyttäjälle ja loput tiedoista vasta pyynnöstä. Tämän avulla voidaan vähentää käyttöliittymän monimutkaisuutta ja antaa käyttäjälle riittävästi tietoa päätöksentekoon. Käyttöliittymän toiminnallisuuden tulee olla johdonmukaista tehtävien suorittamisen tukemiseksi. Käyttöliittymäelementtien täytyy säilyttää sama toiminnallisuus sovelluksen eri osissa, sillä toiminnallisuuden vaihtelu heikentää käyttäjän luottamusta sovelluksen toimivuuteen. (Sensenbach 2022)

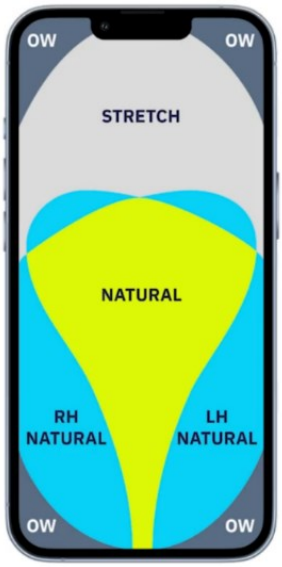


Kuva 8. *Esimerkki johdonmukaisesta käyttöliittymästä, jossa hyödynnetään samanlaisten elementtien ryhmittelyä sekä samanpituisia välityksiä elementtien välillä (Sensenbach 2022).*

Hallinnan tunteen luominen käyttäjille käyttöliittymän käytöstä on tärkeää. Ensimmäinen askel hallinnan tunteen luomiseen on helppokäyttöisen ja intuitiivisen navigaation luominen. Paikkatiedon esittäminen käyttäjälle, esimerkiksi ”sinä olet tässä” -ilmaisimen avulla, on tehokas työkalu tähän tarkoitukseen. Paikkatiedon avulla käyttäjä saa vastaukset kysymyksiin: missä ovat, missä olivat ja minne voivat mennä käyttöliittymällä ollessaan. Tämä auttaa myös tilanteissa, joissa käyttäjä on eksynyt ja tarvitsee selkeän tavan poistua. (Sensenbach 2022)

Toinen askel hallinnan tunteen luomiseen on selkeän palautteen antaminen jokaisesta käyttöliittymällä laukaistusta tapahtumasta. Käyttöliittymän antamalla palautteella tarkoitetaan tapoja, joilla voidaan viestiä käyttäjälle mitä sovelluksessa tapahtuu. Viesti voi olla esimerkiksi varoitusteksti salasanan vaihtamisessa, kun annettu salasana ei täytä sille annettuja vaatimuksia. Usein laukaistuihin tapahtumiin, kuten tilamuutosnapin (*state change button*) painamisesta, palaute voi olla vaatimaton, mutta harvoin laukaistuihin tapahtumiin, kuten salasanan vaihtamiseen, palaute kannattaa olla huomattava, johon käyttäjä kiinnittää huomionsa. (Sensenbach 2022)

Ergonomia on tärkeä huomioida käyttöliittymäsuunnittelussa älypuhelimilla, joita käytetään sormen kosketuksen avulla. Käyttöliittymäelementtien täytyy olla tarpeeksi suuria, jotta kosketuksen avulla voidaan tarkasti valita haluttu kohde. Kohteiden välillä on myös hyvä olla runsaasti tilaa, jotta vältyttäisiin vahingollisilta kosketuksilta. (Sensenbach 2022) Vahingollisille kosketuksille on silti hyvä olla olemassa peruutustoiminto (*undo*), jonka avulla käyttäjä voi nopeasti korjata tai peruuttaa tekemänsä valinnan. Tällaisia ovat esimerkiksi takaisin-navigointilinkit (*back button*) tai valintaikkunoiden sulkemisnapit (*close button*). (Budiu 2015; Rosala 2020)



Kuva 9. *Älypuhelimen näytön kosketusalueet (Sensenbach 2022).*

Toinen tärkeä näkökohta kosketuksen ergonomian huomioimisessa on käyttöliittymäelementtien sijoittelu. Kuvassa 9 esitetään älypuhelimen näytön eri kosketusalueet. Näytön keskellä on luonnollinen (*natural*) alue, joka on helposti ja luontevasti saavutettavissa kaikille käyttäjille. Tälle alueelle kannattaa sijoittaa usein käytetyt elementit ja toiminnallisuudet, kuten varmistusikkunat, joissa pyydetään käyttäjän syötettä. Siniset alueet ovat vastakkaisille puolille sijoittuvia luonnollisia alueita, oikeankätisillä vasemmalla ja vasemmankätisillä oikealla. Venytysalue (*stretch*) on alue, johon käyttäjien täytyy venyttää normaalista käyttöasennosta. Tälle alueelle on hyvä sijoittaa vähemmän käytettyjä toiminnallisuuksia. Näytönkulma-alueet (OW) ovat alueita, jotka olisi hyvä pitää suhteellisen tyhjinä toiminnallisuudesta, koska venyttäessä tai muissa vahinkotilanteissa voi kosketus vahingossa osua näille alueille tahattomasti. Kuitenkin kulma-alueita voidaan hyödyntää tilanteissa, kuten peruuttamattomissa toiminnoissa, joissa käyttäjä täytyy tietoisesti venyttämään ja ymmärtämään mitä on tekemässä. (Sensenbach 2022)

Älypuhelinten käytön keskeytyminen ulkoisten tapahtumien vuoksi on yleistä, mikä vaatii käyttäjän huomion siirtymistä muualle ja laitteen käytön keskeytymistä. Budiun (2015) mukaan tyypillinen älypuhelimen käyttöistunto kestää 72 sekuntia, verrattuna pöytätietokoneen 150 sekunnin käyttöistuntoon. Lyhyet mobiilikäyttöistunnot vaativat suunnittelua keskeytyksien varalle. Käyttäjille on hyvä tarjota mahdollisuus tallentaa keskeneräinen työnsä, jotta käyttäjät voivat palata takaisin siihen mahdollisimman saumattomasti. Tilanteissa, joissa samaa sovellusta voidaan käyttää eri laitteilla, kuten älypuhelimella ja pöytätietokoneella, on tärkeää luoda symmetria laitteiden välille. Budiun (2015) mukaan mobiilikäyttäjät eivät aina tee lopullisia päätöksiä älypuhelimella, vaan saattavat haluta palata tietyn tehtävän suorittamiseen laitteella, jossa näyttö on suurempi. Tietojen ja toiminnallisuuksien on hyvä säilyä samana ja olla saatavilla riippumatta näyttökoosta. (Budiu 2015; Sensenbach 2022)

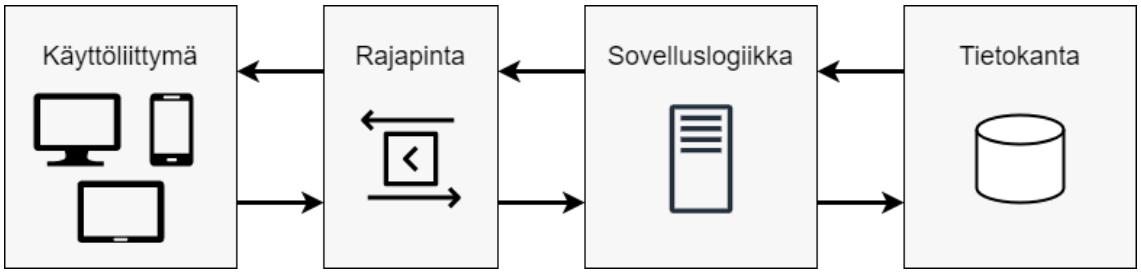
4. WEB-SOVELLUKSEN KEHITTÄMINEN

Tämän tutkimuksen mobiilikäyttöliittymän toteutus keskittyy web-sovelluksen jatkokehitykseen, jolloin on tärkeää ymmärtää web-sovelluskehityksen teoreettiset taustat ja tutkimuksen kontekstissa tarvittavat tiedot. Kohdassa 4.1 esitetään web-sovelluskehityksen arkkitehtuurikehys (*framework*). Kohdassa 4.2 perehdytään Angular web-sovelluskehityskirjaston ominaisuuksiin, joita tullaan hyödyntämään mobiilikäyttöliittymän toteutuksessa. Kohdassa 4.3 keskitytään progressiivisen web-sovelluksen (PWA) malliin ja toiminnallisuuteen.

Web-sovelluksella tarkoitetaan ohjelmistoa, jota käytetään verkkoselaimen käyttöliittymän kautta. Nykypäivänä web-sovelluskehitys on kehittynyt merkittävästi staattisten HTML-sivustojen ajasta kohti nopeasti reagoivia ja mobiiliystävällisiä web-sovelluksia. Kehityksen takana ovat erilaiset web-kehykset ja web-sovelluskehityskirjastot, jotka tarjoavat tehokkaan ja käyttäjäystävällisen tavan kehittää web-sovelluksia sekä pöytätietokoneille että mobiililaitteille. Nämä kehittyneet teknologiat ovat mahdollistaneet monimutkaisempien toiminnallisuuksien ja parempien käyttökokemusten toteuttamisen web-sovelluksissa joka laitteella. (Shahzad 2017)

4.1 Arkkitehtuurikehys

Web-sovelluskehityksessä arkkitehtuurikehyksen avulla voidaan esittää sovelluksen rakenne ja toiminnallisuuden vastuualueet. Usein web-arkkitehtuurikehys esitetään 3-tasoisena arkkitehtuurina (*3-Tier Architecture*), jossa tasot jaetaan käyttöliittymätasoon, sovelluslogiikkatasoon ja tietokantatasoon. Nykyään arkkitehtuuriin usein lisätään myös tiedon välityksen rajapintataso. Arkkitehtuuri helpottaa sovelluksen kehittämistä ja ylläpitoa, kun jokainen taso voi olla toteutettuna erillisinä komponentteina tai palveluina. (William 2022)



Kuva 10. *Web-arkkitehtuurikehys mukaillen lähteistä (Bass et al. 2012; William 2022).*

Käyttöliittymätaso (*presentation layer, frontend*) on arkkitehtuurikehyksen osa, jonka avulla käyttäjät voivat olla vuorovaikutuksessa web-sovelluksen kanssa selaimen kautta. Vuorovaikutuksella tarkoitetaan esimerkiksi CRUD-pyyntöjen (*Create Read Update Delete*) lähettämisen sovelluslogiikkatasolle, joista käyttäjä saa vastauksena tarvitsemansa tiedot käyttöliittymälle. Perinteisesti CRUD-pyynnöstä saatu vastaus on vaatinut koko HTML-sivun uudelleenpäivityksen, mikä on johtanut hitaisiin latausaikoihin.

Nykypäivänä on kuitenkin tullut suosioon yhden sivun web-sovellukset (*Single Page Application, SPA*). SPA lataa ensimmäisellä web-sovelluksen käynnistyskerralla mahdollisimman paljon sisältöä ja päivittää dynaamisesti sovelluksen sisältöä käyttäjän vuorovaikutuksen mukaan. Sisällön dynaaminen päivittäminen on mahdollista AJAX-tekniikan (*Asynchronous JavaScript and XML*) avulla. AJAX ei CRUD-pyynnön tapaan palauta uutta kokonaista versiota sivusta, vaan palauttaa vain käyttäjän tarvitseman tai muuttuneen tiedon päivitettäväksi sovellukseen. SPA:t ovat parantaneet web-sovellusten reagoivuutta ja käyttökokemusta merkittävästi. (Shahzad 2017; William 2022)

Sovelluslogiikkataso (*application layer, backend*) on arkkitehtuurikehyksen avainkomponentti, jonka tärkeimpänä tehtävänä on vastaanottaa käyttäjien pyynnöt käyttöliittymätasolta, suorittaa tarvittava logiikka ja palauttaa tarvittavat tiedot takaisin käyttäjälle (William 2023). Logiikan toimintoihin voi sisältyä esimerkiksi tietokantakyselyjen suorittamista, jossa haetaan tarvittavaa tietoa tietokantatasolta. Sovelluslogiikkataso toteutetaan usein erillisenä osana, koska sen toiminnot vaativat tehokkaan ympäristön verrattuna käyttöliittymätason selaimen rajattuihin resursseihin. (Bass et al. 2012)

Käyttöliittymä- ja sovelluslogiikkataso tarvitsevat keinon keskenään kommunikointiin. Rajapintataso (*Application Programming Interface, API*) toimii tiedonvälityksen tasona käyttöliittymä- ja sovelluslogiikkatasojen välissä mahdollistaen tasojen integroinnin keskenään (William 2023). API voi määrittää ja tarjota joukon kutsuja ja toimintoja, joiden avulla käyttöliittymätaso voi lähettää pyyntöjä ja vastaanottaa tietoa sovelluslogiikkatasolta. API myös varmistaa, että tietomuoto ja kommunikointitapa ovat yhtenäisiä, joka helpottaa ja vahvistaa tasojen yhteensopivuutta.

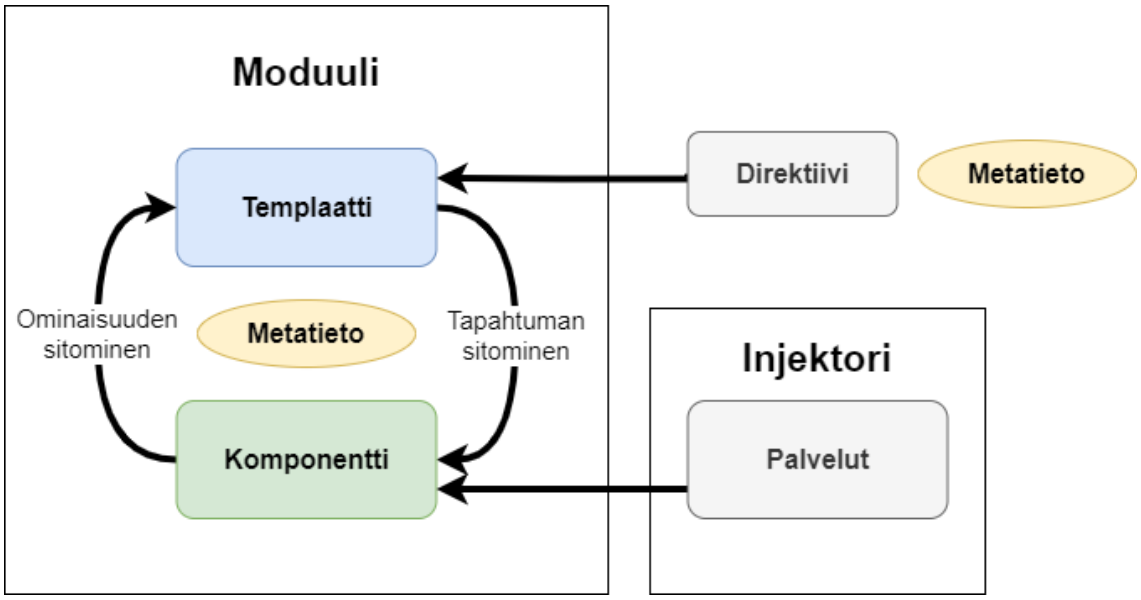
Yksi käytetyimmistä rajapintatason muodoista on REST (*Representational State Transfer*), joka tarjoaa erinomaisen skaalautuvuuden, luotettavuuden ja suorituskyvyn. REST-kommunikoinnissa käytetään HTTP-protokollan kautta kulkevia CRUD-pyyntöjä, joiden tietomuotona hyödynnetään kevyttä JSON-tiedostomuotoa (*JavaScript Object Notation*). JSON mahdollistaa tiedon selkeän ja kompaktin kuvaamisen, mikä tehostaa tiedon siirtoa tasojen välillä. (Richardson et al. 2013; William 2022)

Tietokantataso (*data layer*) on arkkitehtuurikehyksen osa, joka säilyttää web-sovelluksessa käytettyjä tietoja. Käyttöliittymätasolta lähtevä pyyntö kulkee rajapintatason kautta sovelluslogiikkatasolle, josta se edelleen välitetään tietokantatasolle. Usein tietokantatasolla käytetään SQL (*Structured Query Language*) -tietokantoja, joista tietoa haetaan, muutetaan ja lisätään SQL-kyselyiden avulla. SQL-tietokannat sopivat tiedon käsittelyyn, jonka muoto on ennaltaan tiedossa. Nykypäivänä on olemassa myös NoSQL-tietokantoja, jotka soveltuvat tiedon käsittelemiseen, joiden muoto ei ole ennalta tiedossa. NoSQL-tietokannoissa käytetään usein avainarvo (*key-value*)-tekniikkaa, joka mahdollistaa tiedon tallennuksen erilaisissa muodoissa. (William 2022)

4.2 Angular-ohjelmistokehys

Angular on Googlen kehittämä käyttöliittymätason ohjelmistokehys SPA-sovellusten kehittämiseen HTML- ja TypeScript-tekniikoiden avulla. Angular julkaistiin alun perin vuonna 2010 nimellä AngularJS. AngularJS:n ensimmäisenä tarkoituksena oli auttaa web-kehittäjiä luomaan pysyviä verkkolomakkeita tehokkaammin, ja ajan myötä siitä tuli sopiva vaihtoehto yhä monimutkaisempiin kehitysprojekteihin. Kuitenkin AngularJS:n suunnittelumalli rajoitti sen käyttömahdollisuuksia muihin web-ohjelmistokehyksiin verrattuna, mikä aiheutti sen suosion hiipumisen. Googlen kehitystiimi uudelleenkirjoitti ja julkaisi toisen version Angular 2:n, säilyttääkseen sen kilpailukykyisenä modernien web-ohjelmistokehysten rinnalla. Nykyään Angular-termiä käytetään kuvaamaan Angularin versioita kahdesta eteenpäin, aina uusimpaan versioon 16 asti. (Angular 2022a; Xing et al. 2019)

Angular-arkkitehtuuri koostuu karkealla tasolla moduuleista (*module*), komponenteista (*component*) ja palveluista (*service*). Nämä ovat TypeScript-luokkia, jotka merkitään erityyppisiksi dekoraattorin (*decorator*) avulla. Dekoraattorin sisäisessä metatieto-objektissa (*metadata*) määritetään kyseisen luokan tarjoavat ominaisuudet ja mahdolliset riippuvuudet. (Angular 2022a) Seuraavissa alakohdissa käsitellään kuvan 11 Angularin arkkitehtuuria.



Kuva 11. *Angular-arkkitehtuuri mukaillen lähteestä (Trivedi 2016).*

4.2.1 Moduuli

Moduulit ovat Angularin tapa ryhmitellä sovelluksen komponentteja, palveluita ja muita toiminnallisuuksia loogisiin ryhmiin. Moduulit määritetään `@NgModule`-termillä. NgModule ottaa sisäänsä metatieto-objektin, jonka sisältämät ominaisuudet määrittävät moduulin riippuvuudet ja siihen liittyvät toteutukset. (Trivedi 2016) Ohjelmassa 1 esitetään esimerkki AppModule-juurimoduulista (*root module*).

```
1     import { NgModule } from "@angular/core";
2     import { BrowserModule } from "@angular/platform-browser";
3     import { AppComponent } from "./app.component";
4
5     @NgModule({
6       imports: [BrowserModule],
7       providers: [Logger],
8       declarations: [AppComponent],
9       exports: [AppComponent],
10      bootstrap: [AppComponent],
11    })
12    export class AppModule {}
```

Ohjelma 1. *AppModule-juurimoduuli mukaillen lähteistä (Angular 2022b; Trivedi 2016).*

Jokaisella Angular-sovelluksella on määritettynä juurimoduuli, joka toimii sovelluksen käynnistäjänä. Ohjelmasta 1 nähdään, miten AppModule-juurimoduuli määritetään `@NgModule`-dekoroinnilla ja sen sisällä olevassa metatieto-objektissa moduulin ominaisuudet. `Imports`-ominaisuuden avulla voidaan tuoda muita moduuleita ja niiden sisältämiä luokkia kyseisen moduulin toiminnallisuutta varten. `Providers`-ominaisuuteen

voidaan listata palvelut, joita kyseinen moduuli käyttää. `Declarations`-ominaisuus määrittää komponentit, direktiivit (*directive*) ja putket (*pipes*), jotka kuuluvat kyseiselle moduulille. `Exports`-ominaisuudessa määritetään objektit, kuten komponentit, joita halutaan käytettäväksi muiden moduuleiden komponenttien templaateissa. `Bootstrap`-ominaisuus määrittää juurikomponentin, joka isännöi kaikkia muita kyseisen moduulin komponentteja. (Angular 2022b; Trivedi 2016)

4.2.2 Komponentti

Komponentteja käytetään Angular-sovelluksessa eri osien määrittämiseen ja ohjaamiseen. Komponentti määrittelee luokan, joka sisältää ominaisuudet ja toimintalogiikan. Nämä liitetään HTML-templaattiin (*template*), joka määrittää näytettävän näkymän. Jokaisessa Angular-sovelluksessa on vähintään yksi juurikomponentti (*root component*), joka toimii sovelluksen pääsovellusnäkymänä (*view*). Komponentit määritetään `@Component`-dekoraattorin avulla. Sen sisältämät metatiedot kertovat komponentin tarvitsemat rakennuspalikat, joita se tarvitsee komponentin luomiseen. (Angular 2022c) Ohjelmassa 2 esitetään esimerkki AppComponent-juurikomponentista.

```
1     import { Component } from "@angular/core";
2     @Component({
3       selector: "app-root",
4       template: ` <h1 class="title">{{ title }}</h1>
5         <button (click)="onClick()">Click</button>
6         <button [disabled]="isDisabled">Disabled button</button>
7         <p [ngStyle]="{ 'font-weight': 'bold' }">{{ text }}</p>`,
8       styles: [".title { color: red; }"],
9     })
10    export class AppComponent {
11      title = "Click button for text";
12      text = "";
13      isDisabled = true;
14
15      onClick() {
16        this.text = "Hello World!";
17      }
18    }
```

Ohjelma 2. *AppComponent-juurikomponentti mukaillen lähteistä (Angular 2022c; Trivedi 2016).*

Ohjelmassa 2 HTML-templaatti määritetään komponentin dekoraattorin sisällä. Templaatti voidaan myös määrittää linkkinä erikseen luotuun HTML-tiedostoon `templateUrl`-termillä. Komponentin dekoraattorin `styles`-tyylimäärittely voidaan myös luoda erillisenä CSS-tiedostona. Dekoraattorin `selector`-ominaisuus määrittää komponentin

CSS-valitsimen, jonka avulla kyseistä komponenttia voidaan käyttää näkymissä HTML-elementtinä, kuten esimerkin komponentti: `<app-root></app-root>`.

Komponentin ja templaatin välinen vuorovaikutus tapahtuu tietositomisten (*data binding*) avulla. Tietositominen mahdollistaa yhteyden luomisen komponentin ja templaatin välille, jotta näkymää voidaan päivittää synkronisesti muutosten tapahtuessa ja pitää komponentti ja templaatti tietoisina näkymän tilasta. (Angular 2022c) Ohjelmassa 2 käytetään kolmea eri sitomisen muotoa: interpolointi (*interpolation*), ominaisuussitominen (*property binding*) ja tapahtumasitominen (*event binding*). Näiden lisäksi on olemassa kaksisuuntainen sitominen (*two-way data binding*), joka mahdollistaa synkronisen päivittämisen osien välillä, jolloin muutokset molemmissa päivittyvät automaattisesti myös toiseen.

Interpoloinnin avulla komponentin määrittelemää tietoa voidaan esittää näkymässä. Ohjelmassa 2 käytetään `h1`- ja `p`-elementtejä, joiden sisältö tuodaan komponentilta käyttäen `{{value}}` -syntaksia. Ominaisuussitomisen avulla voidaan asettaa arvoja HTML-elementin attribuuteille. Esimerkiksi ohjelmassa 2, kuudennen rivin `button`-elementti on asetettu toimimattomaksi komponentilta saadun `isDisabled`-totuusarvon perusteella. Tapahtumasidonta kuuntelee käyttäjän tekemiä toimintoja käyttöliittymässä. Esimerkiksi ohjelmassa 2, viidennen rivin `button`-elementin painalluksesta kutsutaan komponentin `onClick`-metodia, joka asettaa uuden tekstin `p`-elementtiin.

Direktiivit (*directive*) ovat luokkia, joiden avulla voidaan lisätä ylimääräistä toiminnallisuutta elementeille. Angularissa on kahdenlaisia direktiivejä: rakenteellisia (*structural*) ja attrbuuttityyppisiä (*attribute*). Rakenteelliset direktiivit muokkaavat näkymän ulkoasua lisäämällä, poistamalla ja korvaamalla elementtejä näkymässä. Attribuuttityyppiset direktiivit puolestaan muokkaavat olemassa olevien elementtien toiminnallisuutta tai ulkonäköä. (Angular 2022c) Esimerkiksi ohjelmassa 2 seitsemännen rivin `p`-elementissä käytetään Angularin omaa `ngStyle`-direktiiviä määrittämään elementin tekstityyli.

4.2.3 Palvelu

Angular-palvelulla tarkoitetaan laajaa käsitettä, jota sovellus tarvitsee, kuten arvoa, toimintoa tai ominaisuutta. Palvelu on tyypillisesti luokka, jolla on määritelty tarkoitus, mutta joka ei toteuta näkymän varsinaista logiikkaa. Tällaisia palveluita voivat olla esimerkiksi tiedon hakeminen sovelluslogiikkatasolta tai käsiteltävän tiedon kirjoittaminen konsoliin. (Angular 2022d)

Palveluiden hyödyllisyys korostuu tilanteissa, joissa samaa toiminnallisuutta tarvitaan useissa eri sovelluksen komponenteissa. Palvelu voidaan tehdä tuontikelpoiseksi käyt-

tämällä `@Injectable`-dekoraattoria, joka määrittää palvelulle riippuvuusinjektion (*dependency injection*). Riippuvuusinjektio on osa Angular-kehyksen tarjoamaa mekanismia, joka mahdollistaa komponenttien käyttää muita resursseja. (Angular 2022d) Ohjelmassa 3 esitetään esimerkki Logger-palvelusta.

```
1     import { Injectable } from "@angular/core";
2     @Injectable({
3         providedIn: 'root',
4     })
5
6     export class Logger {
7         log(msg: any)   { console.log(msg); }
8         error(msg: any) { console.error(msg); }
9         warn(msg: any)  { console.warn(msg); }
10    }
```

Ohjelma 3. *Logger-palvelu mukaillen lähteestä (Angular 2022c).*

Ohjelman 3 dekoraattorin metatieto-objektissa määritetään palveluntarjoaja (*provider*) `providedIn`-attribuutin avulla. Tämä kertoo riippuvuusinjektiolle, miten palvelun riippuvuus hankitaan. Palvelu otetaan käyttöön komponentilla ensin määrittämällä palvelu isäntämoduulin `providers`-metatietoon. Tämän jälkeen palvelu määritetään vielä samanlailla komponentin omaan `providers`*-metatietoon.* (Angular 2022d)

4.3 Progressiivinen web-sovellus

Progressiivinen web-sovellus (PWA) on moderneja web-rajapintoja hyödyntäen kehitetty yhden koodikannan sovellus, joka tarjoaa käyttäjälle alustakohtaisen sovelluksen (*native app*) tapaisen käyttökokemuksen kaikilla laitteilla (Richard & LePage 2020). PWA yhdistää alustakohtaisten sovellusten ja perinteisten web-sovellusten parhaat ominaisuudet yhteen. Alustakohtaisen sovelluksen ominaisuuksista PWA toteuttaa esimerkiksi sovelluksen kyvyn toimia ilman verkkoyhteyttä (*offline*), koko näytön käyttöliittymän hyödyntämisen ja helppokäyttöisyyden asennetun kuvakkeen kautta. Perinteisen web-sovelluksen ominaisuuksista PWA toteuttaa yhden koodikannan hyödyntämisen, jotta sovellus on saatavilla kaikilla laitteilla ja voidaan käyttää myös normaalisti selaimen kautta verkon ylitse. (MDN 2023)

PWA:t ovat luonteeltaan web-sovelluksia, sillä ne käyttävät samoja perustoiminnallisuuksia kuten HTML-, CSS- ja JavaScript-koodeja. PWA:ssa kuitenkin hyödynnetään myös lisäominaisuuksia, kuten Service Worker -ohjelmointirajapintaa ja Web App Manifest -JSON-metatietotiedostoa. (MDN 2023)

Service Worker mahdollistaa web-sovellusten tarjoaman lähes samantasoisen käyttökokemuksen luotettavuudesta ja tehokkuudesta kuin alustakohtaiset sovellukset. Service Worker on yksinkertaisimmillaan ajettava toimintalogiikka (*script*), joka suoritetaan verkkoselaimessa ja vastaa sovelluksen välimuistin hallinnasta ja verkkoliikenteen käsittelemisestä. Esimerkiksi jos käyttäjä sulkee web-sovelluksen Service Worker säilytetään selaimen paikalliseen välimuistiin. Seuraavalla käynnistyskerralla Service Worker kuuntelee ja käsittelee kaikki resurssipyynnöt, jotka tarvitaan sovelluksen lataamiseen. Tämä mahdollistaa sovelluksen osittaisen käytön ilman verkkoyhteyttä, kun tarvittavat tiedot ja resurssit tallennetaan ja haetaan välimuistista. (Angular 2022e)

Web App Manifest -JSON-metatietotiedoston avulla voidaan määritellä PWA:n ominaisuudet. Tiedostossa voidaan määritellä sovelluksen nimi, kuvake, väriteema, aloitusnäytön asetukset ja muut tiedot, jotka auttavat sovellusta toimimaan ja näyttäytymään samalla tavalla kuin perinteinen asennettu sovellus. Määritetyn Web App Manifest -tiedoston avulla käyttäjät voivat asentaa PWA-sovelluksen laitteelleen ja käyttää sitä kuin itsenäistä sovellusta. PWA-sovelluksen asentaminen edellyttää toimituslähteeltään suojatun HTTPS-yhteyden tai vastaavan suojan. On myös mahdollista asentaa PWA-sovellus palvelimelle, jolla sitä ajetaan, localhost-isäntäosoitteen kautta, sillä myös sitä pidetään suojattuna yhteytenä. (MDN 2023)

5. MOBIILIKÄYTTÖLIITTYMÄN KÄYTTÖ PAIKASTA RIIPPUMATTA

Ohjelmistojärjestelmien asennus- ja käyttömuodot ovat vaihdelleet viime vuosikymmeninä paikallisesti asennetuista (on-premises) pilvipalveluihin asennettuihin (off-premises) järjestelmiin. Erityisesti pilvipohjaiset ohjelmistot ovat kasvattaneet suosiotaan kiihtyvästi viimeisen vuosikymmenen aikana. Kranzin ja muiden (2016) toteuttamassa tutkimuksessa havaittiin vielä, että suurin osa yrityksistä käytti paikallisia järjestelmiä. Kuitenkin Nakkeeranin ja muiden (2020) toteuttamassa tutkimuksessa havaittiin, että noin 447 yritystä 639 yrityksestä (70 %) suunnitteli siirtymistä paikallisista ohjelmistoista pilvipohjaisiin ratkaisuihin lähivuosien aikana.

Tämän tutkimuksen MES-järjestelmä on asennettavissa kohdeyrityksen asiakkaille on-premises- tai off-premises-ympäristöön. Tämän luvun tavoitteena on selvittää, miten tutkimuksessa MES-järjestelmään kehitettyä mobiilikäyttöliittymää voidaan käyttää näissä ympäristöissä turvallisesti paikasta riippumattomana. Kohdassa 5.1 käsitellään on-premises-ympäristöä ja kohdassa 5.2 off-premises-ympäristöä.

5.1 On-premises

On-premises-ohjelmistolla tarkoitetaan paikallisesti yrityksen toimitiloihin asennettua ohjelmistoa. On-premises-ohjelmisto vaatii toimiakseen yrityksen oman laitteistoinfrastruktuurin, kuten palvelimet ja tietokannat. On-premises-ohjelmiston etuna on, että yritys voi täysin hallita käyttöympäristönsä tietoturvaa ja tiedonhallintaa. Tämä on erityisen tärkeää yrityksille, jotka tarvitsevat täydellisen hallinnan ohjelmiston käsittelevästä tiedosta ja sen tietoturvasta. (Cleo 2023) Esimerkiksi MES-järjestelmissä voidaan käsitellä asiakas-, resepti- ja henkilötietoja, joiden arkaluontoisuus edellyttää niiden säilyttämistä yrityksen omassa infrastruktuurissaan.

Toimitiloissa on-premises-ohjelmistoa voidaan käyttää yhdistämällä laite yrityksen sisäiseen verkkoon. Usein kun sisäiseen verkkoon yhdistetään, saadaan oikeudet käyttää sinne asennettuja ohjelmistoja ja toimintoja. Esimerkiksi tutkimuksen yrityksen on-premises web-sovelluksen tapauksessa sovellus asennetaan asiakkaan omaan sisäiseen verkkoon ja web-sovellukseen päästään käsiksi selaimella sille asetetun verkko-osoitteen kautta. Jos mobiililaite on yhdistetty yrityksen sisäiseen verkkoon, se voi käyttää on-premises-ohjelmistoa ja sen toiminnallisuuksia samalla tavalla kuin yrityksen sisäiset työasemat tai pöytätietokoneet.

On-premises-ohjelmiston oletettu käyttöpaikka on usein yrityksen toimitilat, johon ohjelmisto on asennettu. Kuitenkin yrityksen laitteistoinfrastruktuuriin voidaan antaa oikeudet päästä siihen käsiksi myös yrityksen toimitilojen ulkopuolelta. Yksi tapa tämän mahdollistamiseksi on käyttää RDP-protokollaa (Remote Desktop Protocol) hyödyntävää etäyhteyssovellusta, kuten Microsoftin Remote Desktop -sovellusta, joka mahdollistaa yhteyden muodostamisen yrityksen verkkoon etäältä (Microsoft 2023a). RDP-protokolla luo yhteyden yrityksen verkon ja etäkäyttäjän tietokoneen välille, mikä mahdollistaa pääsyn yrityksen verkkoon turvallisesti. RDP-protokolla on myös kehitetty käytettäväksi mobiililaitteilla, kuten älypuhelimilla. Microsoftin Remote Desktop -mobiilisovelluksen avulla käyttäjät iOS- ja Android-käyttöjärjestelmää käyttävillä mobiililaitteilla voivat yhdistää etänä yrityksen verkkoon ja pääsevät käyttämään Windows-käyttöjärjestelmään asennettua on-premises-ohjelmistoa.

Remote Desktop -yhteyden muodostamisessa mobiililaitteella esiintyy kuitenkin haasteita. Yksi haasteista on käytettävyys, kun etäyhteyden kohteena on usein pöytätietokone, jonka suuri näytön sisältö pyritään sovittamaan mobiililaitteen pienelle näytölle. Tästä voi aiheutua käytettävyysongelmia, kuten käytön sujuvuuden ja tiedon luettavuuden heikkeneminen. Kuitenkin Microsoft jatkaa Remote Desktop -mobiilisovelluksen päivittämistä ja parantamista korjaamalla vikoja sekä lisäämällä uusia käytettävyyttä tehostavia ominaisuuksia (Microsoft 2023b).

Toinen tapa ohjelmiston käyttämiseen yrityksen toimitilojen ulkopuolelta on yhdistää yrityksen sisäiseen verkkoon käyttämällä virtuaalista erillisverkkoa (VPN). VPN-yhteyden avulla yksittäinen laite voidaan turvallisesti yhdistää ja liittää tiettyyn verkkoon, jonka avulla käyttäjä pääsee käsiksi resursseihin, jotka ovat vain saatavilla kyseiseen verkkoon yhdistettynä. Kun laite on yhdistetty verkkoon, on-premises-ohjelmistoa voidaan käyttää selaimen kautta määritetystä verkko-osoitteesta.

5.2 Off-premises

Off-premises-ohjelmistolla tarkoitetaan kolmannen osapuolen (*third party*) palveluntarjoajan palvelimilla isännöimää ohjelmistoa, joka on käytettävissä verkon ylitse. Off-premises-ohjelmistoa varten ei tarvita omaa paikallista laitteistoinfrastruktuuria on-premises-ohjelmistojen tapaan, vaan kaikki tarvittava ohjelmiston suorittamiseen toteutetaan palveluntarjoajan toimesta. (Cleo 2023; Nakkeeran et al. 2020)

Off-premises-ohjelmistoratkaisussa ei kuitenkaan ole on-premises-ohjelmiston tapaista täydellistä hallintaa ohjelmistosta ja sen ympäristöstä. Pessl & Rabel (2022) mukaan

MES-järjestelmien kohdalla tämä on usein tekijä, jonka takia MES-järjestelmää ei käyttöönoteta off-premises-versiona. Syynä voi olla myös huoli arkaluonteisten yritystietojen säilyttämiseen liittyvistä epäselvistä tietosuojasäännöksistä sekä palvelutarjoajaan kohdistuvat mahdolliset laitteistohaavoittuvuudet ja avoimuuden puute palveluntarjoajan kustannuksissa.

Off-premises-ohjelmisto on usein käytettävissä verkon kautta selaimen avulla, mikä ei kuitenkaan tarkoita, että kaikilla olisi pääsy ohjelmistoon. Tutkimuksen kohdeyrityksen tapauksessa MES-järjestelmän off-premises-ohjelmistoversio tarjotaan asiakkaille Microsoft Azure -palveluntarjoajan kautta.

Azure-pilvipalvelussa voidaan määrittää off-premises-ohjelmistolle tietyt oikeudet, jotka rajoittavat ohjelmiston käyttöä vain tietyille osapuolille. Oikeudet off-premises-ohjelmiston käyttöön on jaettu kahteen osaan: verkkoyhteyteen ja henkilökohtaisiin tunnuksiin. Pilvipalvelussa voidaan määrittää mistä verkosta voidaan päästä käsiksi ohjelmistoon. Yksittäiset käyttäjät saavat oikeuden käyttää ohjelmistoa henkilökohtaisilla tunnuksilla, kun heidän tunnuksensa on lisätty käyttäjäryhmän rooliin järjestelmässä. Esimerkiksi yrityksen MES-järjestelmässä on operatiivinen rooli operatiivisille tuotantotyöntekijöille ja hallintarooli johtohenkilöstölle. Off-premises-ohjelmistoon pääsyyn tarvitaan siis yhteys määritetystä verkosta tunnuksilla, jotka ovat lisättynä ohjelmiston käyttäjäryhmän rooliin.

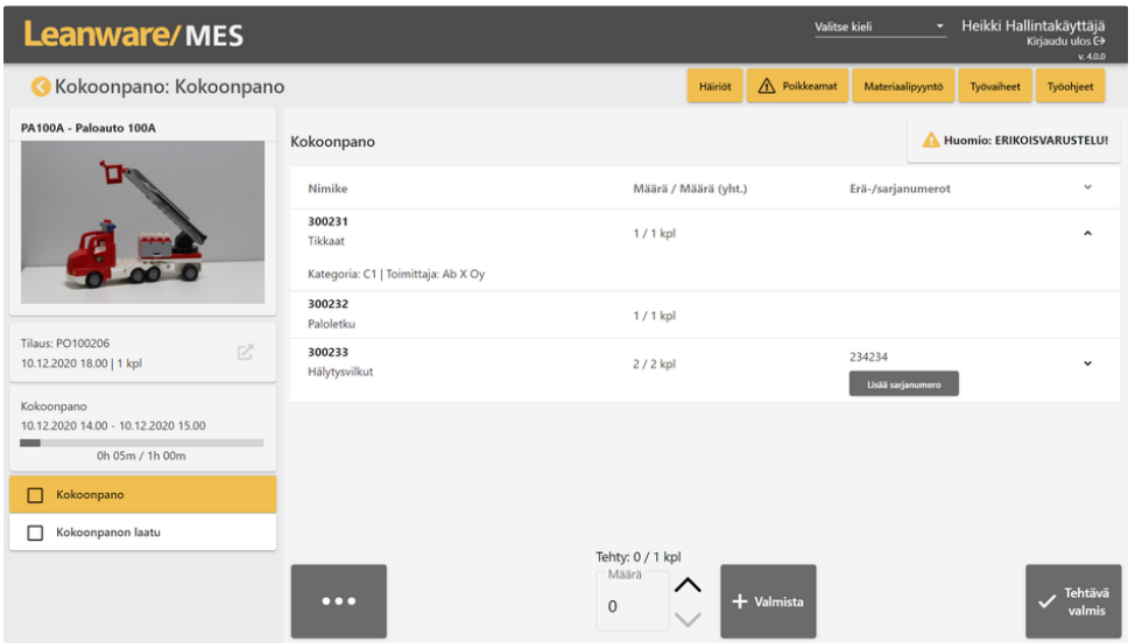
Off-premises-ohjelmistoa voidaan käyttää verkon ulkopuolelta VPN-yhteyden avulla, kuten on-premises-ohjelmistoa. VPN-yhteyden muodostaminen onnistuu riippumatta laitteesta, mikäli kyseinen laite tukee VPN-yhteyden muodostamista tai tarjoaa siihen liittyviä ohjelmistoja tai palveluita.

6. KOHDEYRITYKSEN MES-JÄRJESTELMÄ

Tässä luvussa esitetään yleisellä tasolla tutkimuksen kohdeyrityksen MES-järjestelmä sekä sen tekninen toteutus. Näiden tietojen avulla voidaan ymmärtää, millaiseen järjestelmään sekä arkkitehtuuriin mobiilikäyttöliittymää ollaan toteuttamassa.

6.1 LeanwareMES-järjestelmä

LeanwareMES on visuaalinen ja ohjaava MES-järjestelmä, jonka avulla voidaan ohjata ja jäljittää tuotannon operatiivisia toimintoja. LeanwareMES hyödyntää tuotantoprosessin kaikkia osapuolia organisaation jokaiselta tasolta. Tuotannon operatiiviset työntekijät voivat nähdä MES-järjestelmästä mitä ja koska heidän tulee tehdä tietyn tuotantoprosessin eri työvaiheilla tehtäväkuvauksien, työohjeiden ja laatutarkastusten avustamana. Tuotannon operatiivinen johto saa tietoa tuotannon tilasta ja sen toteutumisesta työjonojen näkymistä, kunnossapidon hälytyksistä ja tuotannon kojelaudoilta. Liikejohto voi tarkastella ja suunnitella tuotantoprosessin toteutumisen tehokkuutta PowerBI-raporttinäkymiltä ja valmistumisten raportoinnista. (LeanwareMES 2023)



Kuva 12. *LeanwareMES-järjestelmän operatiivinen vaihenäyttö (Leanware 2023).*

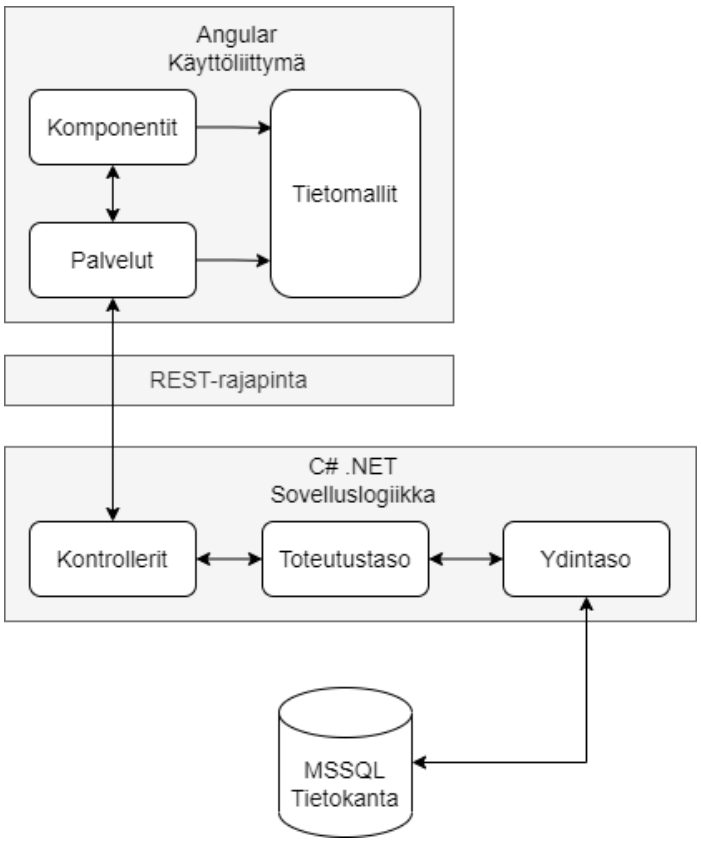
LeanwareMES toteuttaa myös integroinnin muihin järjestelmiin rajapintojen avulla. Integraatiot mahdollistavat tuotantoprosessista saatavan tiedon välittämisen organisaation joka tasolle. ERP-järjestelmäintegraation avulla voidaan tuoda MES-järjestelmään suun-

nitellut tuotantotilaukset ja niissä tarvittavat artikkelit. MES-järjestelmästä voidaan lähettää ERP-järjestelmään MES-järjestelmän kautta tehdyt tuotannon valmistuskuittaukset ja poikkeamatapaukset, joita voidaan tarkemmin analysoida jälkeenpäin ERP-järjestelmässä ja muissa analytiikkaohjelmistoissa. (LeanwareMES 2023)

Vakiorajapinta mahdollistaa integraation myös automaatiotason laitteisiin kone- ja laiteintegraatioiden avulla. Tämän avulla voidaan saada lähes reaaliaikaista tietoa tuotannon fyysisestä tilasta. LeanwareMES-järjestelmä voidaan myös integroida LeanwareWMS-varastonhallintajärjestelmän kanssa, joka mahdollistaa materiaalitietojen yhdistämisen tuotantoprosessin hallintaan ja näkyvyyteen. LeanwareMES-WMS-integraatio mahdollistaa monta eri logistiikan toiminnallisuutta, kuten valmistuksessa tarvittavien materiaalien keräyspyynnöt, materiaalisaldon tuoton ja kulutuksen, varastointipyynnöt valmistuneille tuotteille ja saldotilanteen näyttämisen tietyllä työpisteellä. (LeanwareMES 2023)

6.2 LeanwareMES-järjestelmän tekninen toteutus

LeanwareMES-järjestelmän tekninen toteutus perustuu kerrosarkkitehtuurin suunnitteluperiaatteeseen. Kerrosarkkitehtuurissa jokainen kerros tarjoaa edellistä korkeamman abstraktiotason tietorakenteiden käsittelyyn. Tämä tarkoittaa, että järjestelmä muodostuu useiden kerrosten muodostamasta luokkahierarkiasta, jossa ylimmät tasot vastaavat mahdollisimman suoraan järjestelmän toimintoja. Tällä tavoin pyritään minimoimaan suoraa tietokannan käsittelyä SQL-lauseiden avulla, mikä ei ole järjestelmälle tehokas tapa hakea tarvittavaa tietoa. Kerrosarkkitehtuuri edistää myös järjestelmän ylläpidettävyyttä ja laajennettavuutta, kun jokaisen kerroksen toiminnallisuus voidaan toteuttaa erikseen muista kerroksista. Tämä on mahdollista kerrosten välisillä määritellyillä rajapinnoilla, jotka standardisoivat kommunikoinnin kerrosten välillä. (Leanware 2023)



Kuva 13. *LeanwareMES-järjestelmän kerrosarkkitehtuuri mukaillen lähteestä (Leanware 2023).*

Kuvassa 13 esitetään LeanwareMES-järjestelmän kerrosarkkitehtuuri, joka koostuu käyttöliittymä-, sovelluslogiikka- ja tietokantakerroksista. Käyttöliittymäkerros on toteutettu Angular-web-ohjelmistokehyksellä. Käyttöliittymäkerroksessa esitetään sen käyttämät komponentit ja palvelut, jotka kuvattiin kohdassa 4.2. Kerroksessa esitetään myös tietomallit (*models*), jotka sisältävät kerroksessa käytettävät tietomallit. Tietomalleja käytetään kerroksen sisäisessä toiminnallisuudessa sekä REST-rajapinnassa mallintamaan tietoa, jota lähetetään käyttöliittymä- ja sovelluslogiikkakerroksen välillä. Kun tietomallit vastaavat toisiaan käyttöliittymä- ja sovelluslogiikkakerroksissa, on kerrosten välinen kommunikointi tehokasta ja tarkoituksenmukaista, kun molemmat kerrokset tietävät millaista tietoa käsitellään. (Leanware 2023)

Sovelluslogiikkakerros toteutetaan C#-ohjelmointikielellä ja .NET-ohjelmistokomponenttikirjastoilla. Kerros jaetaan karkeasti kolmeen eri tasoon: kontrollereihin (*controllers*), toteutustasoon (*middle layer*) ja ydintasoon (*MESCore*). Kontrollereiden tarkoituksena on vastaanottaa käyttöliittymän palveluilta saapuvia pyyntöjä ja ohjata niitä eteenpäin sovelluslogiikan toteutuskerrokselle. Toteutuskerros sisältää tietyn näytön tai palvelun tarvitseman logiikan ja käyttää apunaan ydintason toteutusta. Ydintaso toimii arkkiteh-

tuurin toiminnallisuuden keskuksena, käsitellen esimerkiksi bisneslogiikkaa (*business logic*) ja tietokannan hallintaa. Bisneslogiikassa suoritetaan käyttötapauksien toteutukset, joita voi olla esimerkiksi järjestelmän perustapaukset, kuten käyttöliittymällä tapahtuvat vaihekuittaukset. Ydintason kautta tietokantaa käsitellään DTO-luokkien (*Data Transfer Object*) ja Entity Framework -kehyksen (EF) avulla. DTO:t ovat arkkitehtuurissa yksinkertaisia objekteja, jotka säilyttävät ja välittävät tietoa tietokannasta käytettäväksi sovelluslogiikalle ja myös päinvastoin tallentaen tietoa tietokantaan. EF taas mahdollistaa tietokannan käsittelyn käyttämällä .NET-objekteja, kuten DbContext-kontekstiluokkaa, jota kutsumalla voidaan suoraan kirjoittaa ja suorittaa kyselyitä tietokantaan. (Leanware 2023)

Tietokantakerros toteutetaan käyttäen Microsoftin SQL Server-tietokantapalvelinta. Tietokannalla on keskeinen rooli järjestelmässä käytettävien tietomallien määrittämisessä. Tietokannassa määritetty tietomalli toimii perustana järjestelmän muiden kerroksien toteutuksien käyttämälle tiedolle. Tietokannan rakenteet on jaettu itsenäisiin moduuleihin, joilla kuvataan järjestelmän suoritusta sen eri vaiheissa. (Leanware 2023)

7. MOBIILIKÄYTTÖLIITTYMÄN TOTEUTUS

Tässä luvussa esitetään tutkimuksen empiirisessä osuudessa toteutetun mobiilikäyttöliittymän vaiheet ja toteutusprosessi. Kohdassa 7.1 käsitellään mobiilikäyttöliittymälle asetetut vaatimukset ja tavoitteet. Kohdassa 7.2 esitetään luodun toteutuksen vaiheet ja toteutusprosessi.

7.1 Vaatimukset ja tavoitteet

Tutkimuksen suunnitteluvaiheessa määriteltiin tietyt vaatimukset tutkimuksessa toteutettavalle konstruktiolle eli MES-järjestelmään luotavalle mobiilikäyttöliittymälle. Suunnitteluvaiheessa tutkijan ja kohdeyrityksen edustajien kanssa yhdessä suunniteltiin mitä tullaan konkreettisesti toteuttamaan tutkimuksen empiirisessä osuudessa. Taulukossa 1 esitetään mobiilikäyttöliittymän toteutukselle asetetut vaatimukset ja niiden tarkemmat kuvaukset.

Taulukko 1. *Mobiilikäyttöliittymän toteutukselle asetetut vaatimukset.*

Vaatimus	Kuvaus
Mobiilikäyttöliittymä MES-järjestelmään	Luodaan MES-järjestelmään MVP työnjohtajannäkymä sekä parannetaan navigointinäkymien mobiilikäytettävyyttä.
Mobiilikäyttöliittymän käyttäminen on-premises- ja off-premises-ohjelmistoissa	Tarkistetaan mobiilikäyttöliittymän käyttömahdollisuus on- ja off-premises-ympäristöissä paikasta riippumatta.
Mobiilikäyttöliittymän käynnistys kuvakkeen kautta	Lisätään MES-järjestelmän käyttöliittymätasoon PWA-toiminnallisuus.

Ensimmäinen vaatimus keskittyy toteutettavan työnjohtajanäkymän kuvaamiseen. Näkymästä luodaan MVP, jonka tavoitteena on luoda pohja näkymän mahdollista jatkokehitystä varten. Työnjohtajanäkymään suunniteltiin kuvan 2 kojelaudan tyyppisiä tietoja eri resurssien tehokkuuksista ja tilannetiedoista tuotannon nykytilasta. Tietojen avulla tavoitteena on optimoida resurssien käyttöä ja mahdollistaa nopeampaa päätöksentekoa. Näkymä suunniteltiin toteutettavaksi omana komponenttinaan, mikä tarkoittaa, että näkymä toimii itsenäisenä sivunaan eikä ole osa jo olemassa olevaa näkymää.

Ensimmäiseen vaatimukseen liittyy myös aiempien navigointinäkymien mobiiliystävällistäminen. Työnjohtajanäkymän lisäksi toteutetaan käytettävyys- ja responsiivisuusparannuksia navigointinäkymiin mobiilikäytettävyyden parantamiseksi. Nykyiset MES-järjestelmän navigointinäkymät eivät ole helppokäyttöisiä pienemmillä näyttökoilla, kuten älypuhelimilla, joille työjohtajanäkymää suunnitellaan erityisesti käytettäväksi. Näiden parannusten tarkoituksena on helpottaa käyttäjän pääsyä työnjohtajannäkymään mobiililaitteilla sekä myös luoda pohjaa tulevien mobiilikäyttöliittymien toteutuksille muissa MES-järjestelmän osissa.

Toinen vaatimus keskittyy mobiilikäyttöliittymän käyttöön on- ja off-premises-ympäristöissä. Vaatimuksen mukaan mobiilikäyttöliittymän tulee olla käytettävissä molemmissa ympäristöissä. Lisäksi mobiilikäyttöliittymää tulee voida käyttää myös ympäristöjen ulkopuolelta tai etäisyyden päästä. Esimerkiksi on-premises-ympäristössä suoritettavaa mobiilikäyttöliittymää tulee voida käyttää myös toimitilojen ulkopuolelta.

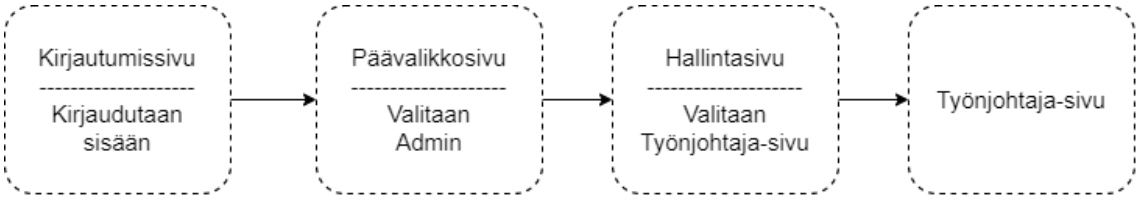
Kolmas vaatimus liittyy mobiilikäyttöliittymän käynnistämiseen kuvakkeen kautta. Vaatimuksen mukaan mobiilikäyttöliittymän tulee olla käynnistettävissä kuvakkeen kautta myös normaalin selaimen lisäksi. Tämän toteuttamiseksi MES-järjestelmän käyttöliittymätasoon lisätään PWA-toiminnallisuus, joka mahdollistaa web-sovelluksen asentamisen kohdelaitteeseen samalla luoden sovelluskuvakkeen.

7.2 Toteutus

Mobiilikäyttöliittymän toteutus aloitettiin näkymien suunnittelusta. Suunnittelussa noudatettiin Sensenbachin (2022) mobiilikäyttöliittymän suunnittelun parhaita käytäntöjä kulkukaavion ja rautalankamallien teolla. Näkymien suunnittelu keskittyi ensimmäiseen vaatimukseen näkymien toteutuksesta. Suunnitelmat toteutettiin draw.io-kaavionluontisovelluksella.

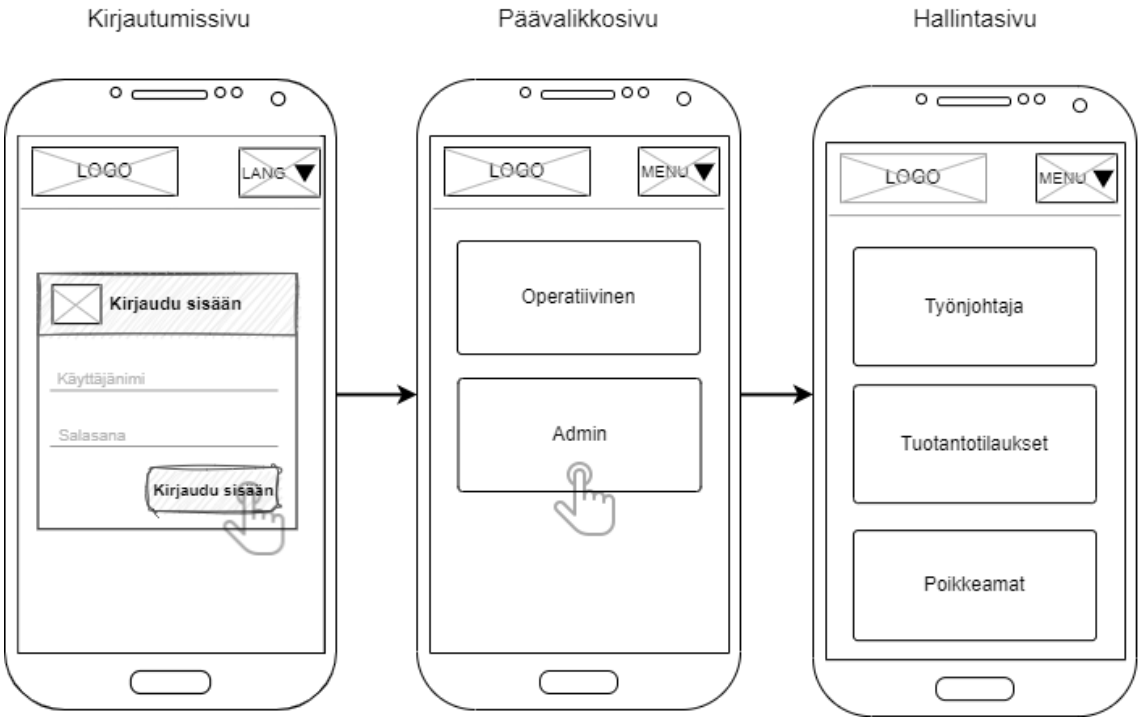
7.2.1 Suunnittelu

Kuvassa 14 esitetään kulkukaavio työnjohtajanäkymään. Kuvan laatikot edustavat erillisiä sivuja, ensimmäisenä kertoen sivun nimen ja viivan alapuolella mitä sivulla tehdään, jotta edetään seuraavalle sivulle. Kyseiset sivut valittiin ensimmäisen vaatimuksen navigointinäkymien mobiilikäytettävyyden parantamisesta, joiden kautta voidaan navigoida Työnjohtaja-sivulle.



Kuva 14. *Kulkukaavio työnjohtajanäkymään.*

Kulkukaaviossa esitetyistä sivuista tehtiin seuraavaksi rautalankamallit. Rautalankamallit tehtiin MES-järjestelmän olemassa olevien käyttöliittymien pohjalta. Malleissa pyrittiin noudattamaan Budiun (2014) mahdollisimman paljon tietojen ja toiminnallisuuden esittämistä myös pienemmällä näyttökoolla kuin isolla näyttökoolla. Kuvat 15 ja 16 esittävät toteutetut mallit sivuista.

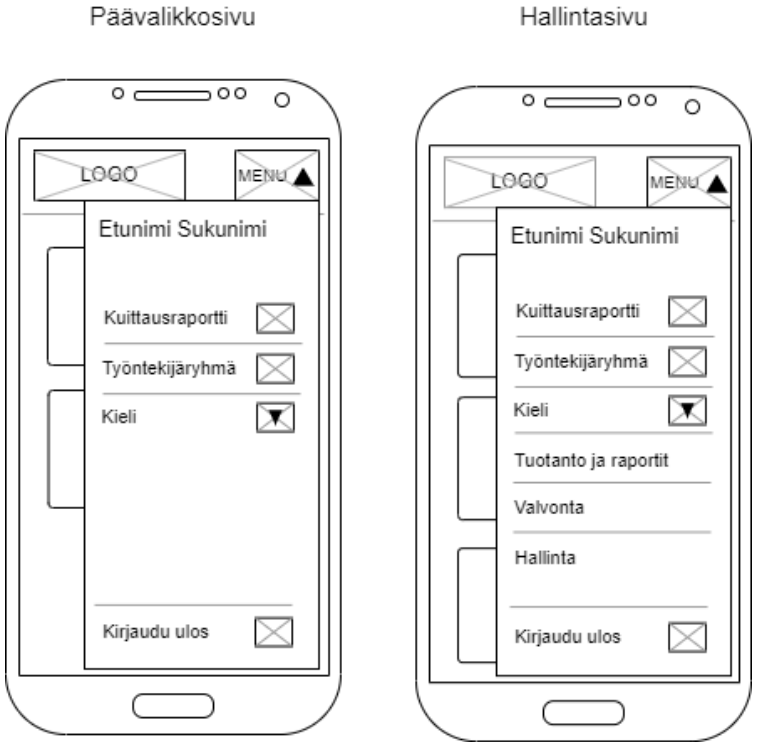


Kuva 15. *Rautalankamallit navigointinäkymistä.*

Kuva 15 esittää rautalankamallit kuvan 14 kulkukaavion kirjautumissivusta, päävalikkosivusta ja hallintasivusta. Kirjautumissivu koostuu yläpalkista ja kirjautumissäiliöstä. Yläpalkissa esitetään LOGO- ja LANG-elementit, jotka ruksauksen avulla esittävät lopullisessa toteutuksessa kuvaa tai ikonia. LOGO-elementti toimii MES-järjestelmän kotipainikkeena (*home button*), joka vie käyttäjän päävalikkosivulle. Tämä toiminto on aktiivinen vasta, kun käyttäjä on kirjautunut järjestelmään. LANG-elementti puolestaan on kuva tai ikoni, josta avautuu alasvetovalikko, josta voidaan valita järjestelmän kieli. Kirjautumissäiliö sisältää yläpalkin, syötekentät (*input*) ja painonapin. Kirjautumissäiliön yläpalkissa

on ikoni ja ohjeteksti, jotka auttavat tunnistamaan säiliön muusta sivun sisällöstä. Päävalikkosivulle siirrytään, kun käyttäjä on syöttänyt käyttäjätietonsa ja painaa ”Kirjaudu sisään”-painiketta.

Päävalikkosivulla näytetään myös yläpalkki, mutta LANG-elementin sijaan esitetään MENU-elementti. MENU-elementti on myös kuva tai ikoni, josta avautuu alasvetovalikko, joka koostuu monesta eri valinnasta. Yläpalkin alapuolella esitetään painettavat navigointivalinnat. Kulkukaavion mukaisesti käyttäjä painaa Admin-valintaa, josta siirrytään hallintasivulle. Hallintasivulla näkymä on samankaltainen kuin päävalikkosivulla, mutta nyt käyttäjälle esitetään enemmän navigointivaihtoehtoja yläpalkin alapuolella. Navigointivaihtoehdot jatkuvat näytön alapuolella, joihin käyttäjä pääsee käsiksi vierittämällä sivua alaspäin.



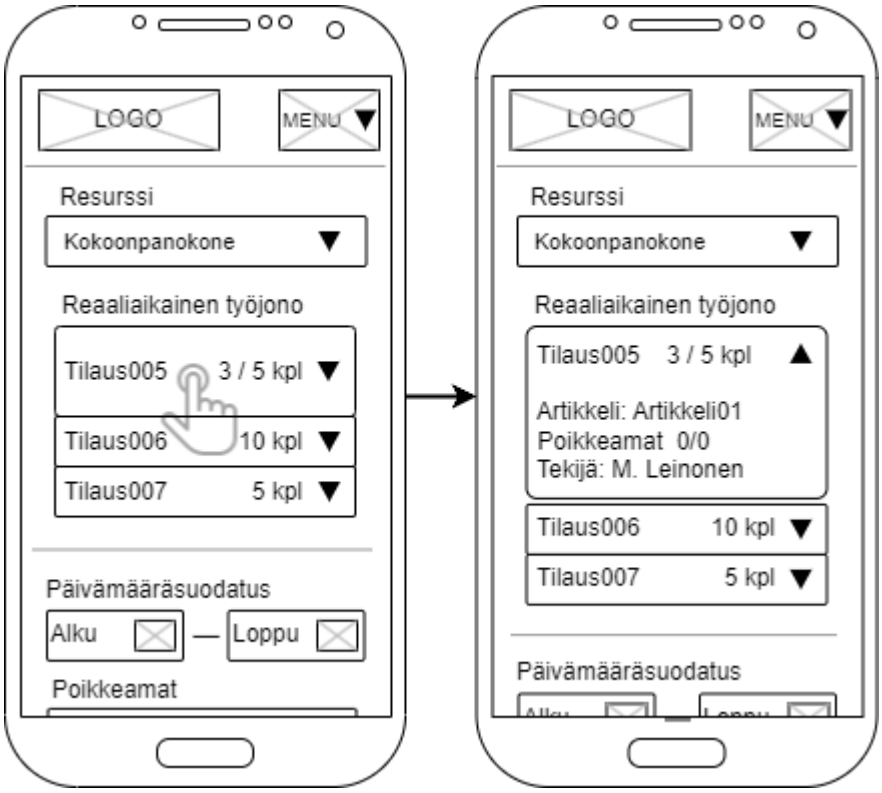
Kuva 16. *Rautalankamallit päävalikkosivun ja hallintasivun alasvetovalikoista.*

Kuvassa 16 esitetään rautalankamallit päävalikkosivun ja hallintasivun MENU-alasvetovalikoista. Alasvetovalikoissa näytetään käyttäjälle eri navigointi- ja toiminnallisuusvalintoja. Valinnat erotetaan horisontaaleilla viivoilla, jotta ne voidaan ryhmitellä tietyn valinnan kuvaustekstin ja ikonin kanssa yhteen. Tämä parantaa käyttöliittymän tunnistettavuutta, sillä ryhmittelyn avulla voidaan tunnistaa valinnat paremmin kuin valintojen ollessa erottelemattomina. Jokainen valinta toimii painikkeena, josta voi valita haluamansa vaihtoehdon painamalla sitä mistä tahansa kohdasta.

Hallintasivulla on kolme valintaa enemmän verrattuna päävalikkosivuun. Nämä lisävalinnat näytetään vasta hallintasivusta eteenpäin, koska ne liittyvät järjestelmän alueelle,

johon pääsy edellyttää rajattuja käyttäjätunnuksia. "Kirjaudu ulos" -valinta sijoitetaan erilleen muista valinnoista mallin alaosassa, koska se on harvoin tarvittava toiminto. Käyttäjän tulee tietoisesti painaa kyseistä valintaa muiden valintojen ulkopuolella käyttääkseen toimintoa.

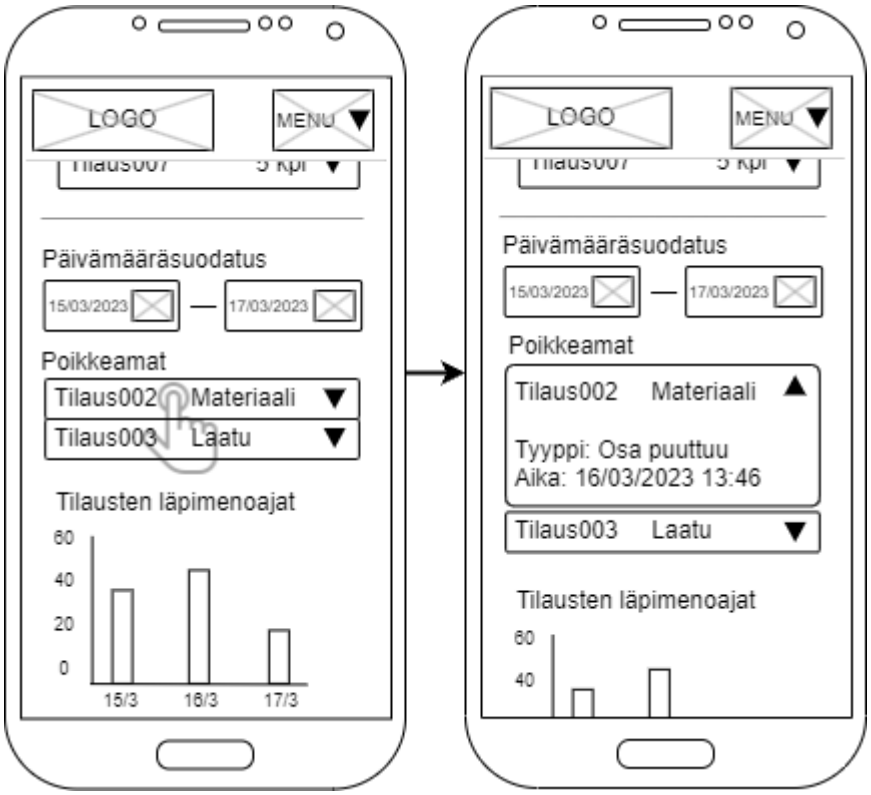
Seuraavaksi luotiin rautalankamalli työnjohtajanäkymästä, jotka esitetään kuvissa 17 ja 18. Kuvan 17 malleissa esitetään näkymä, joka näytetään käyttäjälle, kun saavutaan työnjohtajasivulle. Sivun yläosassa on Resurssi-alasvetovalikko, josta käyttäjä valitsee MES-järjestelmässä määritetyn tuotantokoneen tai -työpisteen. Resurssin valittuaan näkymässä näytetään valitun resurssin Reaaliaikainen työjono-listaus. Listauksessa näytetään kolme työn alla tai työn alle tulevaa tilausta allekkain. Tilauksen tunnisteen vieressä esitetään tilauksen reaaliaikainen tuotetun määrän tilanne, ensin esittäen jo tuotetun määrän ja toisena tavoitteellisen määrän. Painamalla tilausta esitetään käyttäjälle tarkemmat tiedot tilauksesta, kuten artikkeli, jota valmistetaan, avoimet ja korjatut poikkeamat sekä tilauksen nykyinen tekijä.



Kuva 17. *Työnjohtajanäkymän yläosa.*

Kuva 18 esittää työnjohtajanäkymän alaosan, johon käyttäjä pääsee vierittämällä sivua alaspäin. Alaosassa esitetään yläosassa valitun resurssin poikkeama- ja tilausten läpimenoaikatietoja. Ensimmäisenä käyttäjä syöttää haluamansa ajanjakson päivämäärä-suodatus-kenttiin. Tämän jälkeen näytetään käyttäjälle valitsemalta ajanjaksolta resurs-

silla esiintyneet poikkeamat listana ja tehtyjen tilausten läpimenoajat pylväsdiagrammissa. Poikkeamat-listan toiminnallisuus on sama kuin kuvan 17 työjono-listan toiminnallisuus. Painamalla poikkeamaa esitetään poikkeaman tyyppi ja aika, jolloin poikkeama on raportoitu. Tilausten läpimenoajat-diagrammissa esitetään valitulta ajanjaksolta valmistuneiden tilausten keskiarvoiset läpimenoajat päivittäin.



Kuva 18. *Työnjohtajanäkymän alaosa.*

Näytettävät tiedot valittiin pääasiassa kohdan 2.4 tutkittujen tietojen sekä kuvan 2 kojelaudan tietojen perusteella siitä, mitä työnjohtajan olisi hyvä nähdä näkymässä. Tietojen valintaan myös vaikutti tiedot mitä tutkimuksen MES-järjestelmä pystyy tarjoamaan. Näkymän tietomäärä rajautui tutkimuksen MVP-toteutuksen raameihin, jonka tarkoituksena on luoda alustava toteutus mahdollista tulevaa jatkokehitystä varten.

7.2.2 Kehittäminen

Kehittämisalustana toimi Visual Studio Code -tekstieditori ja Google Chrome -verkkoselain. Chrome-verkkoselaimen Device Mode -ominaisuutta hyödynnettiin mobiilikäyttöliittymän testaamisessa kehittämisen aikana. Ominaisuus mahdollisti käyttöliittymän simuloinnin eri näyttöresoluutioilla.

Mobiilikäyttöliittymän kehittäminen aloitettiin kulkukaavion ja rautalankamallien pohjalta. Ensimmäisenä siirrettiin yläpalkin navigointivalinnat Menu-valikon alle. Menu-valikkokomponentti luotiin mes-common-moduulin sisälle Angular CLI komennolla `ng generate component mobile-menu`. Komento loi automaattisesti HTML-, TypeScript- ja CSS-tiedostot, joiden pohjalta oli tehokasta lähteä toteuttamaan komponentin toiminnallisuutta.

Menu-valikkokomponentin mobile-menu.component.html-tiedostoon luotiin valikon avauspainike sekä valikon sisällön rakenne. Avauspainikkeeksi luotiin `button`-elementti, jonka ikoniksi valittiin kolmen viivan hampurilaisikoni (*hamburger menu*), joka on laajasti käytetty ikoni kuvastamaan valikkoa. Muiden elementtien valinnassa hyödynnettiin MES-järjestelmässä jo käytössä olevaa Angular Material -tyylikirjastoa. Tyylikirjasto tarjoaa kansainvälisesti soveltuvia ja helppokäyttöisiä komponentteja käyttöliittymän elementtien esittämiseen. (Angular Material 2023) Valikkokomponentin sisällön esittämiseen valittiin tyylikirjaston tarjoama `mat-menu`-elementti. Aluksi myös tarkasteltiin `mat-drawer-container`-elementtiä, jota käytetään vastaavien valikoiden näyttämisessä, mutta `mat-menu`-elementti nähtiin tarkoitukseen sopivampana sen yksinkertaisuuden puolesta.

Seuraavaksi siirryttiin yläpalkin toteutuksen omaavaan app-komponenttiin. Komponenttiin tuotiin Angularin BreakpointObserver-moduuli, joka mahdollistaa näytön koon muutosten reaaliaikaisen seurannan. Komponentin app.component.ts-tiedoston `ngOnInit`-alustusfunktioon lisättiin näytön koon seuranta, joka esitetään ohjelmassa 4.

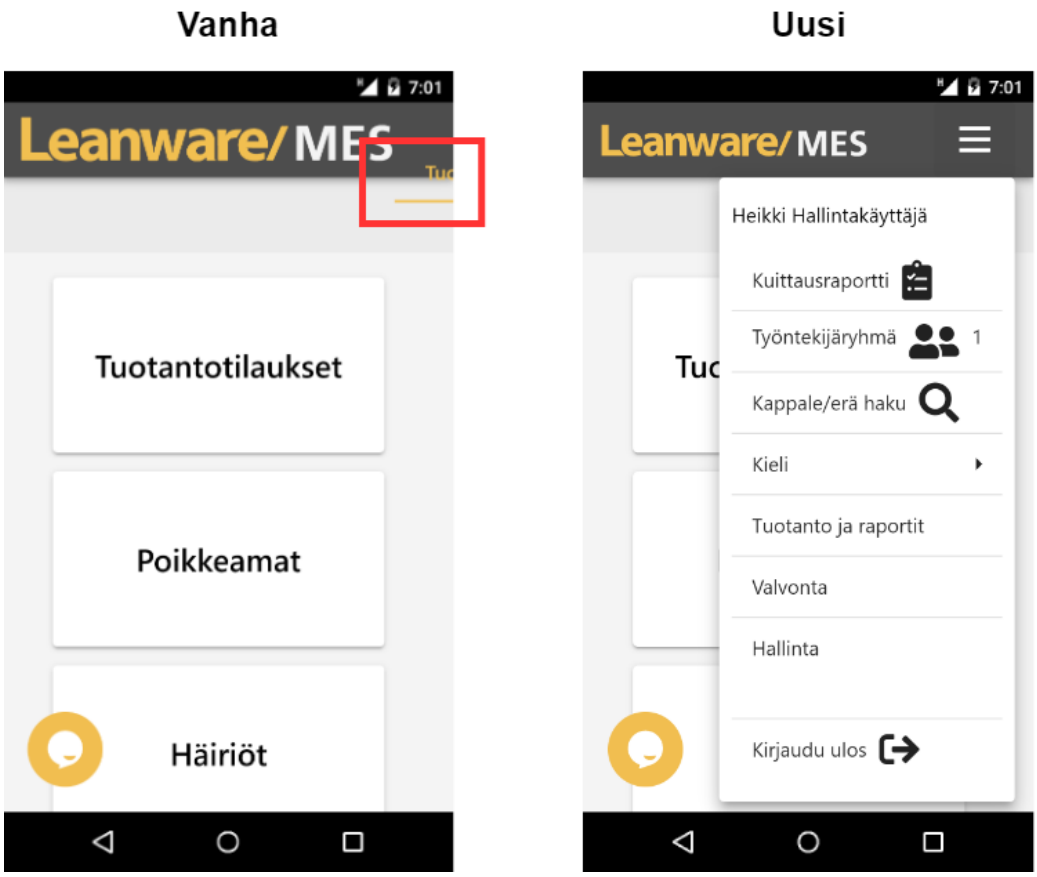
```
1     import { BreakpointObserver, Breakpoints } from '@angular/cdk/layout';
2
3     public isXSmallLayout: boolean;
4     constructor(private breakpointObserver: BreakpointObserver) {}
5
6     ngOnInit() {
7         this.breakpointObserver.observe(Breakpoints.XSmall)
8         .subscribe(result => {
9           if (result.matches) {
10            this.isXSmallLayout = true;
11          } else {
12            this.isXSmallLayout = false;
13          }
14        });
15    }
```

Ohjelma 4. *BreakpointObserver-moduulin käyttäminen näytön koon seuraamiseen.*

Riveillä 1, 3 ja 4 alustettiin tarvittavat ominaisuudet näytön koon tarkkailua varten. Rivillä 6 esitetään `ngOnInit`-metodi, jota kutsutaan ensimmäisenä, kun komponentti luodaan. Rivillä 7 aloitetaan näytön koon tarkkailu kutsumalla `this.breakpointObserver.observe`-metodia. Metodi ottaa parametrinaan yhden tai useampia näytön koon arvoja, joita se tarkkailee. Tässä tapauksessa metodille annettiin tiedostoon rivillä 1 tuodusta Breakpoints-objektista ennalta määritetty näytön keskeytysarvo `(max-width: 599.98px)`, joka aktivoituu, kun näytön leveys alittaa tai ylittää 599.98 pikselin leveyden.

Rivillä 8 käytetään `subscribe`-kuuntelijaa, joka reagoi `observe`-metodin muutoksiin. Kun `observe`-metodi havaitsee, että annettu arvo on saavutettu, se palauttaa objektin, joka sisältää `matches`-totuusarvon, joka kertoo, onko arvo alitettu (*true*) vai ylitetty (*false*). Kun tieto saadaan, se päivitetään `isXSmallLayout`-totuusarvoon, jota voidaan käyttää sisällön näyttämiseen vain tietyillä näytön koilla.

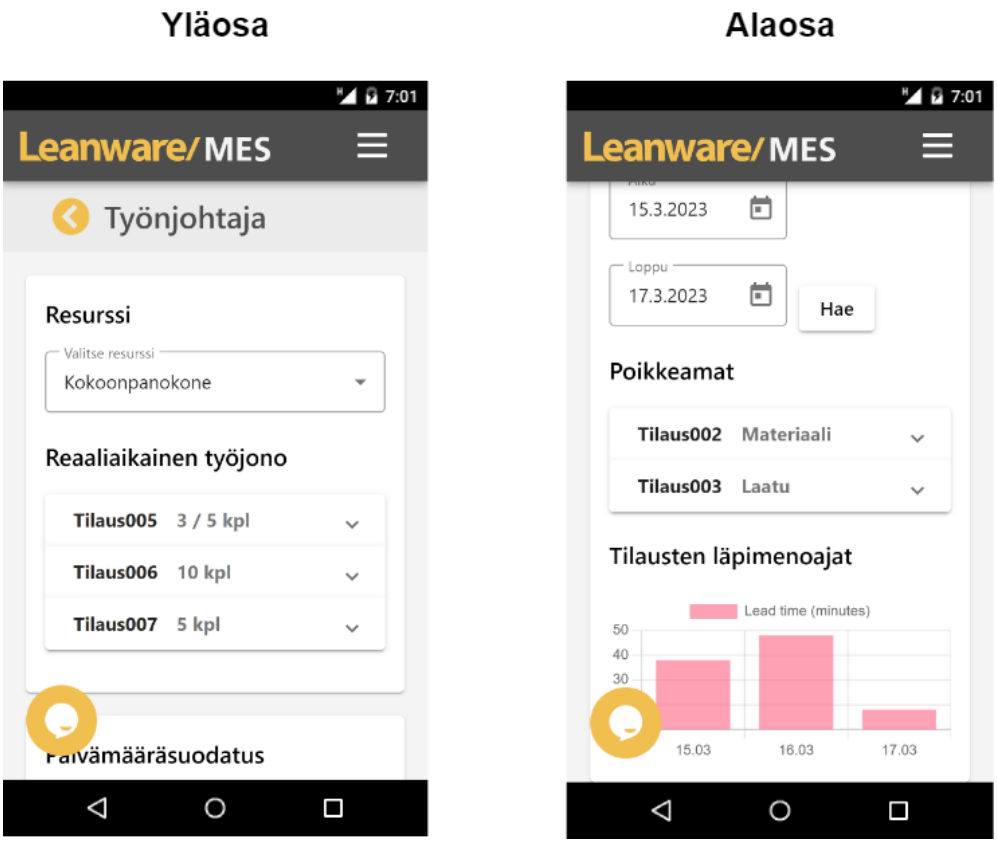
Yläpalkin navigointivalinnat ovat toteutettuina omissa komponenteissaan, joiden toteutusta voidaan kutsua niille määritetyillä HTML-selektoreilla. Navigointikomponentteja ei voitu kutsua sellaisenaan mobile-menu-komponentin toteutuksessa, vaan niiden muotoilua täytyi muokata. Tässä tarkoituksessa hyödynnettiin BreakpointObserver-moduulia. Moduulin ansiosta voitiin luoda navigointikomponenttien HTML-tiedostoihin toinen toteutus käyttämällä Angularin `*ngIf`-attribuuttia ja ominaisuussitomista. Attribuutin avulla voidaan esimerkiksi määrittää mitä elementtejä komponentti näyttää, kun attribuutille annettu arvo on saavutettu. BreakpointObserver-moduulin ja `*ngIf`-attribuutin avulla voitiin määrittää, milloin esitetään nykyinen toteutus ja milloin esitetään Menu-valikossa esitettävä toteutus. Tämän ansiosta voitiin kutsua olemassa olevia navigointikomponentteja molemmissa app.component-yläpalkissa sekä mobile-menu-component-valikossa vähentäen koodin uudelleenkirjoitusta.



Kuva 19. *Navigointivalintojen toteutukset.*

Kuvassa 19 esitetään vanha ja uusi toteutus navigointivalinnoista mobiilikäyttöliittymässä. Vanhasta toteutuksesta nähdään punaisella korostettu navigointivalikon Tuotanto ja raportit -valinnan alkukirjaimet ja loput navigointivalinnoista jää näytön ulkopuolelle, josta niitä ei voida käyttää. Uudessa toteutuksessa rautamallin mukaan yläpalkin käyttäjätiedot ja navigointivalinnat sijoitettiin Menu-valikon alle, kun näytön leveys pienenee alle 600 pikselin. Uusi toteutus mahdollistaa MES-järjestelmän navigoinnin myös mobiililaitteen pienellä näytöllä.

Seuraavaksi siirryttiin työnjohtajanäkymän toteutukseen. Työnjohtajanäkymälle päätettiin luoda uusi moduuli, koska useat nykyiset päänavigointivalintojen näkymät olivat toteutettuina omina moduuleinaan. Uusi moduuli luotiin komennolla `ng generate module foreman`, mikä loi mes-foreman.module.ts-tiedoston ja mes-foreman-kansion. Kansioon luotiin foreman-komponentti näkymän toiminnallisuutta varten.



Kuva 20. *Työnjohtajanäkymän ylä- ja alaosa.*

Kuvien 17 ja 18 rautalankamallien mukaisesti luotiin kuvassa 20 esitettävä näkymä. Käyttöliittymä jaettiin kahteen osaan rautalankamallien harmaan viivajakajan mukaan. Tämä tarkoitti sitä, että Resurssi-alasvetovalikko ja Reaaliaikainen työjono -listaus sijoitettiin ensimmäiseen säiliöön ja Päivämääräsuodatus-valinnat, Poikkeamat-listaus sekä Tilausten läpimenoajat -pylväsdiagrammi sijoitettiin toiseen säiliöön.

Säiliöiden elementiksi valittiin `mat-card`-säiliöelementti, jonka avulla voitiin erotella elementit selvästi omiin kokonaisuuksiinsa. Resurssi-alasvetovalikko toteutettiin `mat-select`-elementtillä, jonka sisällä käytettiin `mat-option`-valintaelementtejä. Listauksissa käytettiin `mat-accordion`-harmonikkaelementtiä pääelementtinä ja sen sisällä `mat-expansion-panel`-elementit mahdollistivat lisätietojen esittämisen valintaa painamalla. Tilausten läpimenoajat -pylväsdiagrammissa käytettiin MES-järjestelmässä käytettyä ng2-charts Angular-kaaviokirjastoa.

7.2.3 Käyttöympäristöt ja testaus

Kolmas vaatimus PWA-toiminnallisuuden lisäämisestä tehtiin seuraavaksi. Toiminnallisuuden lisäämisessä havaittiin että MES-järjestelmässä oli jo olemassa manifest.web-

manifest.json-tiedosto, jossa määritetään tiedot PWA-sovelluksen asentamiseksi. Tiedoista muutettiin `orientation`-ominaisuuden arvo `landscape`-arvosta `any`-arvoksi. Ominaisuuden `any`-arvo mahdollistaa PWA-sovelluksen käytön molemmissa vaaka- ja pystyasennossa.

Toisen vaatimuksen mukaisesti uuden mobiilikäyttöliittymän toimivuutta testattiin kokeilevan testauksen muodossa on- ja off-premises-ympäristöissä. Ensin testattiin on-premises-ympäristö, johon luotiin uusi MES-järjestelmän instanssi yrityksen sisäisessä verkossa sijaitsevalle palvelimelle. Älypuhelimella (Samsung Galaxy A53, käyttöjärjestelmä Android 13) luotiin yhteys verkkoon VPN-ohjelman avulla, kun oltiin poissa yrityksen verkosta. Chrome-mobiiliverkkoselaimella navigoitiin instanssille määritettyyn verkko-osoitteeseen. MES-järjestelmä avautui normaalisti sekä uusi mobiilikäyttöliittymä näytettiin käyttäjälle. Uuden navigointivalikon kautta siirryttiin onnistuneesti myös työnjohtajanäkymään. On-premises MES-järjestelmän instanssia ja uutta mobiilikäyttöliittymää käytettiin onnistuneesti älypuhelimen kautta yrityksen verkon ulkopuolelta VPN-yhteyden avulla.

Verkon ulkopuolelta pääsyä off-premises MES-järjestelmään testattiin yrityksen olemassa olevalla off-premises instanssilla. Off-premises instanssi oli samassa versiossa kuin pohjana käytetty versio mobiilikäyttöliittymän kehittämisessä. Älypuhelimen VPN-ohjelman avulla yhdistettiin yrityksen verkkoon, josta oli pääsy off-premises instanssiin. Kuten on-premises tapauksessa, Chrome-verkkoselaimella navigoitiin instanssin määrittämään verkko-osoitteeseen. Off-premises MES-järjestelmä avautui älypuhelimella normaalisti. Vaikka uutta mobiilikäyttöliittymää ei ollut off-premises instanssissa, molemmissa on- ja off-premises ympäristöissä MES-järjestelmän instanssi toimi Windows Service -palvelusovelluksena samalla asennuspaketilla. Tällöin on-premises ympäristössä suoritetun mobiilikäyttöliittymän testauksen tulokset ovat sovellettavissa myös off-premises ympäristön testaukseen.

On-premises MES-järjestelmän instanssia ei voitu asentaa älypuhelimelle PWA-sovelluksena. Instanssia suoritettiin HTTP-yhteyden avulla, mikä tarkoitti, että se ei täyttänyt PWA-sovelluksen vaatimusta suojatusta HTTPS-yhteydestä. PWA-sovelluksen asentaminen ja avaaminen kuvakkeen kautta kuitenkin onnistui palvelinkoneella instanssin localhost-isäntäosoitteen kautta ja pienentämällä sovellusikkunaa alle 600 pikselin leveyteen voitiin mobiilikäyttöliittymää myös käyttää onnistuneesti. Off-premises instanssi puolestaan voitiin asentaa PWA-sovelluksena. Azure-palveluntarjoaja tarjosi off-premises instanssin verkko-osoitteelle sertifikaatin, joka loi suojatun HTTPS-yhteyden ja siten mahdollisuuden asentaa PWA-sovelluksen.

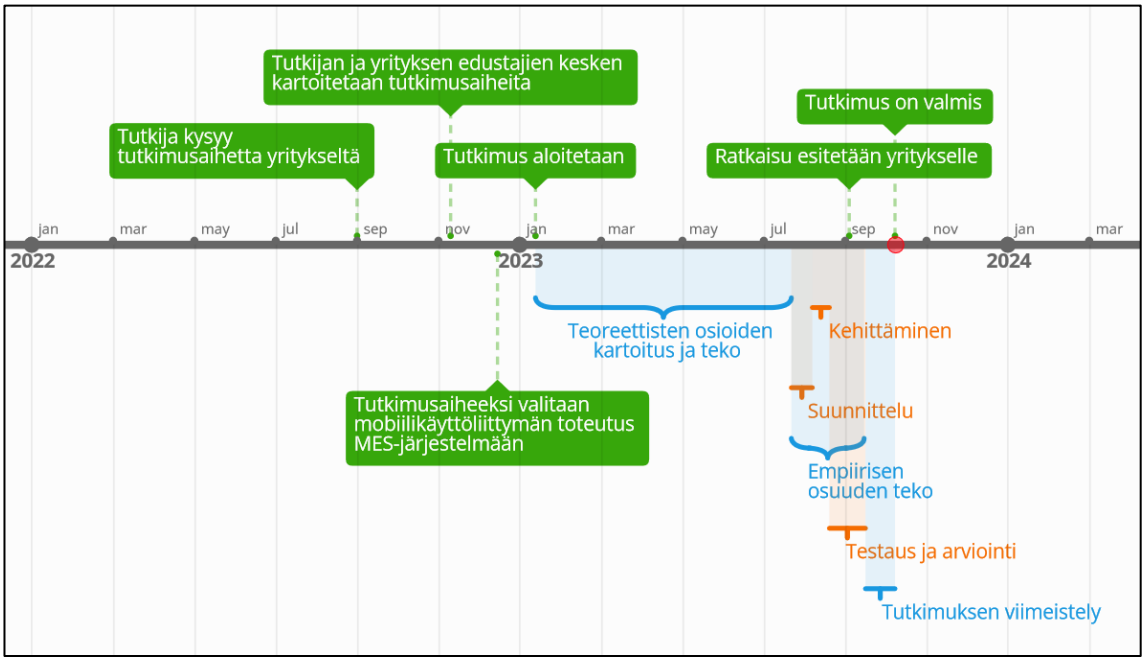
8. TUTKIMUKSEN ARVIOINTI

Hevnerin ja muiden (2004) mukaan suunnittelutieteestä syntyvää tulosta voidaan arvioida kahdessa vaiheessa. Ensimmäisessä vaiheessa pyritään tarkastelemaan täyttääkö luotu konstruktio sille asetetut vaatimukset ja toisessa vaiheessa tavoitteena on tunnistaa, ratkaiseeko luotu konstruktio kohdeyrityksen alkuperäisen ongelman. Suunnittelutieteen tutkimustuloksia tulee myös arvioida tieteellisestä näkökulmasta, jossa tavoitteena on tarkastella mitä uutuudellista tieteellistä tietoa on syntynyt konstruktion luomisessa.

Kohdassa 8.1 arvioidaan tutkimuksen prosessia ja tulosta tutkimuksessa käytetyn Peffersin ja muiden (2007) suunnittelutieteellisen tutkimuksen prosessin vaiheiden avulla sekä yllä Hevnerin ja muiden (2004) esittämien arvioinnin vaiheiden avulla. Kohdassa 8.2 pohditaan tutkimustulosten yleistettävyyttä ja luotettavuutta sekä mahdollisia jatkokehityskohteita.

8.1 Tutkimusprosessin ja tuloksen arviointi

Tutkimuksessa pyrittiin noudattamaan Peffersin ja muiden (2007) suunnittelutieteellisen tutkimuksen prosessin vaiheita ongelman ratkaisemisessa. Kuva 21 esittää karkean aikajanan tutkimuksen tapahtumista ja vaiheista.



Kuva 21. *Tutkimuksen aikajana.*

Ensimmäisen vaiheen ongelman tunnistaminen onnistuttiin tekemään tutkimuksen alkuvaiheissa syksyllä vuonna 2022. Syksyn aikana kohdeyrityksen edustajien kesken kartoitettiin mahdollisia tutkimusaiheita. Aiheeksi valittiin mobiilikäyttöliittymän kehittäminen MES-järjestelmään tutkijan aiheeseen osoittaman kiinnostuksen ja kohdeyrityksen tarpeen perusteella.

Toisen vaiheen tutkimuksen tavoitteiden asettaminen saatiin valmiiksi alkuvuonna 2023 ennen tutkimuksen aloittamista. Kohdeyrityksen edustajien kanssa käytiin läpi suunnitelma tutkimuksen rakenteesta ja etenemisestä alustavan sisällysluettelon pohjalta. Sisällysluetteloa vielä muokattiin edustajien toiveiden pohjalta ja tutkimus aloitettiin.

Kolmannen vaiheen konstruktion suunnittelu ja toteutus aloitettiin kevään edetessä teoreettisten osioiden teolla. Tässä kartoitettiin kirjallisuudesta olemassa olevan tiedon pohjalta tarvittavat tiedot tutkimuskysymyksiin vastaamiseen sekä luotavaa konstruktiota varten. Kesällä aloitettiin konstruktion varsinainen suunnittelu- ja toteutusprosessi. Luodut suunnitelmat mobiilikäyttöliittymänäkymistä lähettiin kohdeyrityksen edustajille kommentoitavaksi ja suunnitelmat nähtiin sellaisina mitä oltiin alun perin ajateltu tutkimuksen suunnitteluvaiheessa. Suunnitelmien pohjalta toteutettiin mobiilikäyttöliittymän kehittäminen.

Neljännen vaiheen ratkaisun todentaminen tehtiin alakohdassa 7.2.3 tehdyssä testauksessa, jossa todistettiin konstruktion luoma ratkaisu kohdeyrityksen ongelmaan. Viidennen vaiheen arviointi esitetään seuraavana Hevnerin ja muiden (2004) vaatimusten täyttämisen ja alkuperäisen ongelman ratkaisun todentamisella.

Ensimmäisen vaatimuksen mukaan luotiin mobiilikäyttöliittymän MVP-toteutus työnjohtajanäkymästä ja navigointivalintojen mobiilikäytettävyyden parantamisesta. Vaatimus todennettiin täytetyksi testauksessa, jossa MES-järjestelmää voitiin käyttää käytettävyydeltään hyvin älypuhelimella. Kohdeyrityksen edustajat myös vahvistivat vaatimuksen täyttymisen omassa kokeilevassa testauksessaan.

Toisen vaatimuksen osalta varmistettiin, että mobiilikäyttöliittymä oli käytettävissä kohdeyrityksen MES-järjestelmässä on- ja off-premises ympäristöissä paikasta riippumatta. Samassa testauksessa VPN-yhteyden avulla vaatimus todennettiin täytetyksi, kun toteutusta voitiin käyttää kohdeyrityksen ympäristöissä paikasta riippumatta. Toinen kohdeyrityksen edustajista pystyi todentamaan VPN-yhteyden kautta toteutuksen toimivuuden. Vaatimus nähtiin täyttyvän molempien kohdeyrityksen ja tutkijan toimesta, vaikka toinen edustajista ei testannut vaatimuksen toimivuutta.

Kolmas vaatimus liittyi mobiilikäyttöliittymän avaamiseen kuvakkeen kautta eli PWA-toiminnallisuuden lisäämiseen, joka myös todennettiin saman testauksen yhteydessä. Testauksessa PWA-sovellus voitiin asentaa, avata kuvakkeen kautta ja käyttää toteutuksen mobiilikäyttöliittymää. Kohdeyrityksen edustajat eivät testanneet vaatimusta erikseen, koska näkivät tutkimuksessa tehdyn testauksen riittäväksi mobiilikäyttöliittymän toimivuuden testauksena jo olemassa olleessa PWA-toiminnallisuudessa.

Konstruktio suurelta osin ratkaisee kohdeyrityksen alkuperäisen ongelman, joka koski MES-järjestelmän mobiilikäytettävyyden parantamista ja asiakkaiden kysyntään vastaamista MES-järjestelmän käytöstä älypuhelimilla. Konstruktio esittää MVP:n parantamaan MES-järjestelmän mobiilikäytettävyyttä hyödyntämällä RWD-kehitystavan periaatteita. Samoja periaatteita hyödyntämällä voidaan parantaa mobiilikäytettävyyttä myös muissa MES-järjestelmän osissa.

Konstruktio ei kuitenkaan heti vastaa asiakkaiden kysyntään, sillä esimerkiksi työnjohtajanäkymää täytyy ensin jatkokehittää hakemaan oikeaa tuotantotietoa tietokannasta, jolloin se tarjoaisi täyden käyttökokemuksen näkymän täydestä toimivuudesta. Lisäksi työnjohtajanäkymän ja yläpalkin navigointivalikon muutokset tulisi vielä sisällyttää MES-järjestelmän päätuoteversiohallintaan ja lopulta loppuasiakkaiden käyttöön tulevissa tuoteversiojulkaisuissa.

Konstruktion luomisessa syntyi hieman uutuudellista tieteellistä tietoa. Eräänä uutuudellisena tieteellisenä tietona voidaan nähdä, miten voidaan suunnitella ja toteuttaa mobiilikäyttöliittymä osaksi olemassa olevaa web-pohjaista MES-järjestelmää RWD-kehitystavan avulla. Tutkimuksessa tehdyssä teoreettisessa kartoituksessa ei löydetty kirjallisuudesta suoraan vastaavaa tutkimusta.

Kohdeyrityksen edustajat antoivat toteutuksesta palautetta avoimen keskustelun muodossa. Edustajat näkivät toteutuksen olevan hyödyllinen lähtökohta MES-järjestelmän mobiilikäytettävyyden jatkokehittämistä varten. Edustajat näkivät myös, että toteutus tarjoaa tukea kohdeyrityksen myynti- ja konsulttihenkilöstölle, kun he voivat kertoa asiakkaille, että tulevaisuudessa heillä on mahdollisuus käyttää MES-järjestelmää myös älypuhelimilla.

8.2 Pohdinta

Peffersin ja muiden (2007) suunnittelutieteellisen tutkimuksen prosessin kuudennessa vaiheessa viestitetään tutkimuksen tuloksen yleistettävyydestä. Tutkimuksen tulos on yleistettävissä muihin web-sovelluksiin. Varsinkin web-sovelluksissa, joissa käytetään

Angular-ohjelmistokehystä yhdessä Angular Material -tyylikirjaston kanssa, tulos on lähes kokonaan siirrettävissä. Samoja RWD-kehitystavan Angular BreakpointObserver -moduulin toteuttamia periaatteita voidaan hyödyntää lähes kaikissa CSS-tyylimäärittelyn omaavissa web-sovelluksissa. Esimerkiksi mediakyselyiden avulla voidaan reagoida moduulin tapaan näytönkoon muuttumiseen.

Tutkimuksessa esiintyy myös yleispäteviä piirteitä. Käytettävyyden ja mobiilikäyttöliittymän suunnittelun esitettyjä periaatteita voidaan hyödyntää erilaisissa mobiilisovelluksissa, kuten alustakohtaisten mobiilisovellusten käyttöliittymäkehityksessä. Kyseiset periaatteet ovat myös sovellettavissa eri toimialoille eivätkä rajoitu vain tuotannonohjaukseen, koska kyseisten lukujen lähteet pyrkivät juuri korostamaan niiden yleistettävyyttä.

Tieteellisestä näkökulmasta tutkimuksen tulokset olisivat olleet luotettavimpia, jos toteutukselle olisi luotu tarkempi testaussuunnitelma. Näin olisi voitu tarkemmin arvioida vaatimusten toteutumista konstruktiossa. Myös kohdeyritykseltä saatua palautetta olisi voitu kerätä systemaattisemmin ja luotettavammin, esimerkiksi haastatteluiden muodossa. Kaiken kaikkiaan tutkimuksessa kuitenkin onnistuttiin tutkimuksen tavoitteiden toteuttamisessa luomalla ratkaisu kohdeyrityksen kohtaamaan ongelmaan.

Tutkimuksen jatkokehityskohteena voi toimia työnjohtajan perusnäkymän täyden toiminnallisuuden loppukehittäminen sekä integroiminen yläpalkin navigointivalikon muutosten kanssa MES-järjestelmään ja tuoteversiojulkaisuihin. Integroimisen jälkeen voitaisiin kerätä kohdeyrityksen asiakkaiden käyttäjiltä palautetta mobiilikäyttöliittymän toimivuudesta ja käytettävyydestä. Palautteen pohjalta voitaisiin iteroidusti parantaa toteutusta sekä kasvattaa käyttäjien tyytyväisyyttä MES-järjestelmään.

9. YHTEENVETO

Tämän tutkimuksen tavoitteena oli suunnitella ja toteuttaa mobiilikäyttöliittymä kohdeyrityksen MES-järjestelmään. Tavoitteen saavuttamiseksi hyödynnettiin suunnittelutieteellistä tutkimusmenetelmää sekä sen prosessin eri vaiheita. Mobiilikäyttöliittymän toteuttamiseen tarvittava tieto kartoitettiin kirjallisuudesta tutkimuksen teoreettisissa luvuissa. Luvuista saatujen tietojen pohjalta vastattiin tutkimukselle asetettuihin tutkimuskysymyksiin ja luotiin tarvittava pohja mobiilikäyttöliittymän toteutukselle.

Tutkimuksen kirjallisuuden kartoituksen perusteella MES-järjestelmien käyttöä älypuhelimilla ei nykypäivänä vielä tueta laajasti. Syinä laajan tuen puuttumiseen nähtiin olevan MES-järjestelmän tuottaman suuren tietomäärän näyttämisen hankaluudet pienillä näytöillä, tuotantoympäristöjen ainutlaatuiset toiminnallisuustarpeet sekä vakiintuneen mobiilistandardin puuttuminen. Haasteista huolimatta on havaittu, että MES-järjestelmiin suunnitellut mobiiliratkaisut ovat nähty parantavan tuotantoprosessin läpinäkyvyyttä ja toimintojen ketteryyttä. Lisäksi Teollisuus 4.0 -konseptissa älypuhelimien tarjoamaa tiedonvälitystä pidetään tärkeänä suuntauksena tuotantoprosessin joustavuuden ja tehokkuuden edistämisessä.

Mobiilikäyttöliittymän suunnitteluun ja web-sovelluksen kehittämiseen keskittyvien teoreettisten lukujen avulla saatiin arvokasta tietoa mobiilikäyttöliittymän toteutusta varten. Kulkukaavion ja rautalankamallien avulla voitiin luoda yksinkertaiset, mutta samalla laadukkaat suunnitelmat mobiilikäyttöliittymän toimivuudesta ja sisällöstä. Teoreettiset kartoitukset käyttöliittymätason Angular-ohjelmistokehykseen ja RWD-kehitystapaan antoivat puolestaan kattavaa tietoa mobiilikäyttöliittymän käytännön kehitystä varten.

RWD-kehitystapaa noudattamalla luotiin älypuhelimille suunnattu työjohtajanäkymä ja yläpalkin navigointivalikko kohdeyrityksen MES-järjestelmään. Angularin BreakpointObserver-moduuli osoittautui tehokkaaksi tavaksi toteuttaa RWD-kehitystavan mukaista näytönkoon seurantaa ja sen perusteella eri sisältöjen dynaamista esittämistä. Näytönkoon seurannasta saatuja arvoja voitiin tuoda HTML-elementeille käyttöön Angularin ominaisuussitomisen avulla. Lisäksi Angular Material -tyylikirjaston tarjoamat valmiiksi muotoillut HTML-elementit mahdollistivat tehokkaan tavan esittää sisällön visuaalisesti ja toiminnallisesti.

LÄHTEET

Allen, L., Atkinson, J., Jayasundara, D., Cordiner, J., & Moghadam, P. Z. (2021). Data visualization for Industry 4.0: A stepping-stone toward a digital future, bridging the gap between academia and industry. *Patterns*, *2*(5), 100266.

Angular. (2022a). Introduction to Angular concepts. Viitattu: 29.05.2023. Saatavilla: https://angular.io/guide/architecture

Angular. (2022b). Introduction to modules. Viitattu: 29.05.2023. Saatavilla: https://angular.io/guide/architecture-modules

Angular. (2022c). Introduction to components and templates. Viitattu: 30.05.2023. Saatavilla: https://angular.io/guide/architecture-components

Angular. (2022d). Introduction to services and dependency injection. Viitattu: 30.05.2023. Saatavilla: https://angular.io/guide/architecture-services

Angular. (2022e). Angular service worker introduction. Viitattu: 05.06.2023. Saatavilla: https://angular.io/guide/service-worker-intro

Angular Material. (2023). Angular Material - Material Design components for Angular. Viitattu: 08.10.2023. Saatavilla: https://material.angular.io/

Bass, L., Clements, P., & Kazman, R. (2012). Software Architecture in Practice: Software Architect Practice_c3. Addison-Wesley.

Budiu, R. (2015). Mobile User Experience: Limitations and Strengths. Nielsen Norman Group. Viitattu: 16.05.2023. Saatavilla: https://www.nngroup.com/articles/mobile-ux/

Cleo. (2023). On Premise vs. Cloud: Key Differences, Benefits and Risks. Viitattu: 05.06.2023. Saatavilla: https://www.cleo.com/blog/knowledge-base-on-premise-vs-cloud

Gröger, C., Stach, C., Mitschang, B., & Westkämper, E. (2016). A mobile dashboard for analytics-based information provisioning on the shop floor. International Journal of Computer Integrated Manufacturing, 29(12), 1335-1354.

Gröger, C., & Stach, C. (2014). The mobile manufacturing dashboard. In 2014 IEEE International Conference on Pervasive Computing and Communication Workshops (PERCOM WORKSHOPS) (pp. 138-140). IEEE.

Harju, V. (2023a). Tuotannon digitalisoinnin järjestelmät. Leanware Oy. Viitattu: 14.03.2023. Saatavilla: https://leanware.fi/yhteiso/blogi/tuotannon-digitalisoinnin-jarjestelmat/

Harju, V. (2023b). Tuotannon läpinäkyvyys. Leanware Oy. Viitattu: 19.06.2023. Saatavilla: https://leanware.fi/yhteiso/blogi/tuotannon-lapinakyvyys-2/

Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design science in information systems research. Management Information Systems Quarterly, 28(1), 75-105.

Icons8. (2023). Natural User Interface 2. Viitattu: 14.07.2023. Saatavilla: https://icons8.com/icon/23540/natural-user-interface-2

ISA-101.01. (2015). Human Machine Interfaces for Process Automation Systems. ISA Standard.

InTech. (2019). New ISA101 HMI technical report focuses on usability and performance. Viitattu: 25.04.2023. Saatavilla: https://www.isa.org/intech-home/2019/july-august/departments/new-isa101-hmi-technical-report-focuses-on-usabili

ISA-95. (2005). Enterprise - Control System Integration, Part 3: Activity Models of Manufacturing Operations Management. ISA Standard.

ISO-9241-11. (2018). Ergonomics of human-system interaction — Part 11: Usability: Definitions and concepts. Viitattu: 01.05.2023. Saatavilla: https://www.iso.org/obp/ui/#iso:std:iso:9241:-11:ed-2:v1:en

Jaskó, S., Skrop, A., Holczinger, T., Chován, T., & Abonyi, J. (2020). Development of manufacturing execution systems in accordance with Industry 4.0 requirements: A review of standard-and ontology-based methodologies and tools. Computers in Industry, 123, 103300.

Kaczmarczyk, V., Zezulka, F., Benešl, T., Arm, J., Marcoň, P., Jirsa, J., & Venkrbec, L. (2022). Revisiting the Role of Manufacturing Execution Systems in Industry 4.0. IFAC-PapersOnLine, 55(4), 151-157.

Kranz, J. J., Hanelt, A., & Kolbe, L. M. (2016). Understanding the influence of absorptive capacity and ambidexterity on the process of business model change–the case of on-premises and cloud-computing software. Information Systems Journal, 26(5), 477–517.

LeanwareMES. (2023). LeanwareMES – visuaalinen ja ohjaava tuotannonohjausjärjestelmä. Viitattu: 18.04.2023. Saatavilla: https://leanware.fi/tuotteet/tuotannonohjausjarjestelma-mes/

Leanware. (2023). Leanware, Sisäinen dokumentaatio. Viitattu 24.04.2023.

Mantravadi, S., Jansson, A. D., & Møller, C. (2020). User-friendly MES Interfaces: Recommendations for an ai-based chatbot assistance in industry 4.0 shop floors. In Intelligent Information and Database Systems: 12th Asian Conference, ACIIDS 2020, Phuket, Thailand, March 23–26, 2020, Proceedings, Part II 12 (pp. 189-201). Springer International Publishing.

MDN. (2023). What is a progressive web app? – Progressive web apps (PWAs). Viitattu: 05.06.2023. Saatavilla: https://developer.mozilla.org/en-US/docs/Web/Progressive_web_apps/Guides/What_is_a_progressive_web_app

Microsoft. (2023a). Remote Desktop clients for Remote Desktop Services and remote PCs. Viitattu: 06.06.2023. Saatavilla: https://learn.microsoft.com/en-us/windows-server/remote/remote-desktop-services/clients/remote-desktop-clients

Microsoft. (2023b). What's new in the Remote Desktop client for Android and Chrome OS. Viitattu: 06.06.2023. Saatavilla: https://learn.microsoft.com/en-us/windows-server/remote/remote-desktop-services/clients/android-whatsnew

MESA. (2023). History of the MESA models. Viitattu: 18.04.2023. Saatavilla: https://mesa.org/topics-resources/mesa-model/history-of-the-mesa-models/

Nakkeeran, A., Niranga, M., & Wickramarachchi, R. (2020). A Model for On-Premises ERP System and Cloud ERP Integration. Proceedings of the International Conference on Industrial Engineering and Operations Management Dubai, UAE, March 10-12, 2020

Nielsen, J. (2012). Usability 101: Introduction to Usability. Nielsen Norman Group. Viitattu: 07.05.2023. Saatavilla: https://www.nngroup.com/articles/usability-101-introduction-to-usability/

Niemelä, H. (2020). Sovelluksen käytettävyys. Verkkolehti SeAMK. Viitattu: 07.05.2023. Saatavilla: https://lehti.seamk.fi/alykkaat-ja-energiatehokkaat-jarjestelmat/sovelluksen-kaytettavyys/

O'Brien, L. (2023). How ISA-101 Lifecycle Standard Improves Operator Effectiveness with Display Design and Lifecycle Management. Viitattu: 16.05.2023. Saatavilla: https://www.arcweb.com/industry-best-practices/how-isa-101-lifecycle-standard-improves-operator-effectiveness-display

Peffers, K., Tuunanen, T., Rothenberger, M. A., & Chatterjee, S. (2007). A design science research methodology for information systems research. Journal of management information systems, 24(3), 45-77.

Pessl, E., & Rabel, B. (2022). Digitization in Production: A Use Case on a Cloud-based Manufacturing Execution System. In Proceedings of the 2022 8th International Conference on Computer Technology Applications (pp. 206-210).

Procházka, T. (2012). Complete.png. Wikimedia Commons. Viitattu: 13.05.2023. Saatavilla: https://commons.wikimedia.org/wiki/File:Complete.png

Richardson, L., Amundsen, M., & Ruby, S. (2013). RESTful Web APIs: Services for a Changing World. O'Reilly Media, Inc.

Richard, S. & LePage, P. (2020). What are Progressive Web Apps?. Viitattu: 05.06.2023. Saatavilla: https://web.dev/what-are-pwas/

Rix, M., Kujat, B., Meisen, T., & Jeschke, S. (2016). An agile information processing framework for high pressure die casting applications in modern manufacturing systems. Procedia CIRP, 41, 1084-1089.

Rosala, M. (2020). User Control and Freedom (Usability Heuristic #3). Nielsen Norman Group. Viitattu: 21.05.2023. Saatavilla: https://www.nngroup.com/articles/user-control-and-freedom/

SAP. (2023). What is an MES (manufacturing execution system)?. Viitattu: 18.04.2023. Saatavilla: https://www.sap.com/finland/insights/what-is-mes-manufacturing-execution-system.html

Schade, A. (2014). Responsive Web Design (RWD) and User Experience. Nielsen Norman Group. Viitattu: 09.05.2023. Saatavilla: https://www.nngroup.com/articles/responsive-web-design-definition/

Sensenbach, R. (2022). Top Tips for Great Mobile Interface Design. Seminaarivideo. Inductive Automation. Viitattu: 19.05.2023. Saatavilla: https://www.inductiveautomation.com/resources/icc/2022/top-tips-for-great-mobile-interface-design

Shahzad, F. (2017). Modern and responsive mobile-enabled web applications. Procedia Computer Science, 110, 410-415.

Trivedi, J. (2016). Basic Architecture of Angular 2 Applications. Viitattu: 29.05.2023. Saatavilla: https://www.c-sharpcorner.com/article/basic-architecture-of-angular-2-applications/

Tokola, H., Gröger, C., Järvenpää, E., & Niemi, E. (2016). Designing manufacturing dashboards on the basis of a Key Performance Indicator survey. Procedia CIRP, 57, 619–624.

Usability.gov. (2023). User Interface Elements. U.S. General Services Administration Technology Transformation Services. Viitattu: 13.05.2023. Saatavilla: https://www.usability.gov/how-to-and-tools/methods/user-interface-elements.html

W3Schools. (2023). Responsive Web Design – Introduction. Viitattu: 13.05.2023. Saatavilla: https://www.w3schools.com/css/css_rwd_intro.asp

William. (2022). Web Application Architecture: The Latest Guide 2022. *ClickIT*. Viitattu: 27.05.2023. Saatavilla: https://www.clickittech.com/devops/web-application-architecture/

Wikimedia Commons. (2007). Usability-ca. Viitattu: 09.05.2023. Saatavilla: https://commons.wikimedia.org/wiki/File:Usability-ca.svg

Xing, Y., Huang, J., & Lai, Y. (2019). Research and analysis of the front-end frameworks and libraries in e-business development. In Proceedings of the 2019 11th International Conference on Computer and Automation Engineering (pp. 68-72).

Zarbo, R. J., Varney, R. C., Copeland, J. R., D'Angelo, R., & Sharma, G. (2015). Daily management system of the Henry Ford Production System: QTIPS to focus continuous improvements at the level of the work. American journal of clinical pathology, 144(1), 122-136.