Haaga-Helia
ammattikorkeakoulu Oy

Digitaalisen tuottavuuskehittämisen luonne ja digitalisaation ennakoiminen

Palvelu- ja asiantuntija-alojen työtehtävien koneellistamisen ennakointimalli organisaatioiden strategiatyön tueksi

Lilja Tamminen

Opinnäytetyö
Tietojenkäsittelyn koulutusohjelma
2017

Tiivistelmä

Haaga-Helia ammattikorkeakoulu Oy

Tekijä(t)
Lilja Tamminen

Koulutusohjelma
Tietojenkäsittely

Raportin/Opinnäytetyön nimi
Digitaalisen tuottavuuskehittämisen luonne ja digitalisaation ennakoiminen

Sivu- ja liitesivumäärä
62 + 1

Opinnäytetyössä haetaan kirjallisuudesta keinoja arvioida tietoteknisiin sovelluksiin perustuvan tuottavuuskehittämisen luonnetta ja siihen perustuvaa digitalisaation etenemiskaavaa. Opinnäytetyön keskeinen taustaoletus on, että ihmistyötä automatisoivien sovellusten kehittämistä ei ohjaa ainoastaan niiden edellyttämä teknologian taso vaan myös taloudelliset kannustimet kohdentaa niiden kehittämiseen tutkimusresursseja. Aihetta tarkastellaan sekä perinteisen ohjelmistokehityksen että neuroverkkoihin tukeutuvan koneoppimispohjaisen sovelluskehityksen lähtökohdista.

Kirjallisuuskatsauksen pohjalta esitetään työmarkkinoilla keskeisten tehtävien tai kykyjen digitalisaation ennakointimalli, jota organisaatiot voivat hyödyntää arvioidessaan liiketoiminta-alansa palvelu- ja asiantuntijatyön automatisoinnin mahdollisuuksia. Mallin tarkoitus on tukea organisaatioiden strategiatyötä ja auttaa ennustamaan markkinatoimijoiden välistä kilpailutilannetta.

Ennakointimallin toimivuutta testataan yksinkertaisella arviointimenetelmällä ja vertaillaan arvioinnin tuloksia muiden tutkimusryhmien ennakointimallien arviointimenetelmien löydöksiin. Löydösten pohjalta esitetään analyysi palvelu- ja asiantuntijatyön digitaalisen tuottavuuskehittämisen etenevistä työmarkkinavaikutuksista.

Opinnäytetyö ei pyri ennustamaan milloin tietyt työtehtävät, osatehtävät tai kyvyt tullaan automatisoimaan, vaan mallin tarkoitus on havainnollistaa niiden digitalisaatiopotentiaalia vertailemalla niiden keskinäistä järjestystä ja suhteellista etäisyyttä toisistaan. Näin opinnäytetyö tarjoaa teknologiselta ja liiketaloudelliselta pohjalta keinon ennakoida sovelluspohjaisen tuottavuuskehittämisen soveltamisalan ennustettavia muutoksia, joita koneoppimispohjaisen sovelluskehityksen kehittyminen mahdollistaa.

Asiasanat
digitalisaatio, ohjelmistokehitys, automaatio, tekoäly, neuroverkot, tulevaisuudentutkimus

Sisällys

1 Johdanto

2010-luvulla työn murros, robotisaatio ja digitalisaatio ovat käsitteinä olleet laajasti esillä julkisessa keskustelussa – erityisesti englanninkielisissä maissa, mutta myös Suomessa. Tässä keskustelussa taloustieteilijä John Maynard Keynesin vuonna 1930 esittelemä teknologisen työttömyyden ilmiö on ollut yleinen spekulaation aihe. Vastaavanlainen innostus automaation ja teknologisen työttömyyden käsitteen ympärillä koettiin viimeksi 1980- ja 1990-luvuilla tietotekniikan nopeasti edullistuessa, mutta innostus laantui, kun suuria ja nopeita murrosilmiöitä ei ilmennyt. Keskeinen syy tähän oli tekoälytutkimuksen stagnaatio.

1990-luvulta asti informaatioteknologiaan tukeutuva tuottavuuskehittäminen on täysautomaation tavoittelun sijaan käsitetty palvelu- ja asiantuntija-aloilla synonyymiksi ohjelmistoihin ja tietojärjestelmiin tukeutuvalle prosessikehittämiselle mm. toiminnanohjausjärjestelmähankkeiden muodossa. Tämän kehityksen seurauksena on syntynyt laaja kirjo erilaisia erityisesti toimistotyötehtävien tehokkuutta kasvattamaan pyrkiviä ohjelmistoratkaisuja. Osana samaa kehityskulkua organisaatioiden tietohallintotoiminnot sellaisena kuin me ne nykyään käsitämme ovat syntyneet. Ohjelmistoihin pohjaava tuottavuuskehittäminen on lisäksi tuonut mukanaan tarpeen erilaisille tietohallintomalleille ja -viitekehyksille sekä tarpeen sovelluskehittäjien työkaluston ja menetelmien kehittämiselle.

Keinotekoisiksi neuroverkoiksi kutsuttuihin laskentamalleihin pohjaava ns. syvä oppiminen (engl. *deep learning)* on tehnyt ensimmäisiä populaarimedian uutiskynnyksen ylittäneitä läpimurtoja vasta 2010-luvulla ja erityisesti vuodesta 2015 alkaen. Näitä pitkään odotettuja tekoälytutkimuksen läpimurtoja mahdollistanut keskeinen tekijä on ollut hajautetun laskentatehon markkinahinnan kova lasku ja sen saatavuuden paraneminen. Viime vuosina tapahtuneiden läpimurtojen myötä neuroverkkoihin pohjaava oppimispohjainen sovelluskehitys on muuttunut tutkimuksellisesta kuriositeetista nopeaa kasvua kokevaksi liiketoiminnan alaksi, jonka tuotekehityksen kilpajuoksuun osallistumista monet ICT-alan suuryhtiöt kuten IBM, Google ja Microsoft tavoittelevat jatkuvasti kasvavilla tuotekehitysbudjeteilla. Lisäksi IT-alan startup-verkostoissa voidaan havaita huomattava kilpajuoksu oppimispohjaisen sovelluskehittämisen liiketoiminnan ohjelmistojen markkinoille.

Erityisesti 2010-luvun puolivälin jälkeen sovelluskehityksen työllisyysvaikutukset esimerkiksi kuljetus- ja logistiikka-alalla ovat aiheuttaneet keskustelua sekä ilmiöön liittyvää tutkimustyötä. Silti teknologialähtöiset yritykset strukturoidusti tutkia ohjelmointi- ja oppimispohjaisen sovelluskehityksen automaatiopotentiaalin eli digitalisoinnin soveltamisalan kasvua palvelu- ja asiantuntija-aloilla ovat jääneet harvinaisiksi.

Opinnäytetyössä pyritään analysoimaan koneoppimis- ja ohjelmointipohjaisen sovelluskehittämisen luonnetta, niiden erityispiirteitä ja työtehtävien automaation etenemisjärjestystä perustuen yhtäältä työtehtävien automatisoitavuuden arvioihin tutkimuskirjallisuudessa ja toisaalta taloudellisiin kannustimiin automatisoida ihmisten suorittamaa työtä. Keskeinen opinnäytetyön taustaoletus on, että ihmistyötä automatisoivien sovellusten kehittämistä ei ohjaa ainoastaan sovellusten toimintojen edellyttämä teknologinen kehitysaste vaan myös taloudelliset kannustimet ja kiinnostus kohdentaa tutkimusresursseja erilaisten sovellusten ja niiden edellyttämän teknologian kehittämiseen.

Analyysin pohjalta esitetään työtehtävien suorittamiseen vaadittavien komponenttisten kykyjen digitalisaatiojärjestyksen ennakoimiseen abstrakti mutta kvantifioitavaksi tarkoitettu malli, jota yritykset ja julkisyhteisöt voivat hyödyntää strategiamäärittelyssään ja päätöksenteossaan esimerkiksi markkinatoimijoiden välisen kilpailutilanteen ennakoimisessa tai toimintansa kehittämisessä vastaamaan tulevaisuuden markkinaolosuhteita.

Digitalisoinnin työmarkkinavaikutusten arviointia varten mallia arvioidaan yksinkertaisella kvantifiointimenetelmällä, jonka avulla voidaan approksimoida komponenttisten kykyjen keskinäistä digitalisaatiojärjestystä. Arvioinnin tulosten ja jo tapahtuneen teknologisen kehityksen valossa esitetään lyhyesti arvioita helpoista ja toistaiseksi tavoittamattomissa olevista digitalisoinnin kohteista sekä näiden laajan digitalisoinnin eli digitalisaation työmarkkinavaikutusten luonteesta.

2 Käsitteistö

Tuottavuus on taloustieteen käsitteistössä tuotanto-panos -suhdetta kuvaava mitta. Jos hyödykkeen kuten palvelun, energian tai materiaalisen tuotteen tuotantoon kuluvia resursseja voidaan vähentää ilman, että lopputuotteen käyttöarvo vastaavasti heikkenee – tai käyttöarvoa voidaan nostaa ilman resurssien vastaavaa lisäystä – sanotaan, että tuottavuus kasvaa.

Hyödykkeen tuottajalla on yleensä taloudellinen kannustin parantaa tuottavuuttaan, koska näin tuottaja voi pyrkiä kasvattamaan itselleen jäävää osuutta hyödykkeen myyntihinnasta. Kilpailluilla markkinoilla tuottajat kilpailevat myös markkinaosuudesta siten, että he pyrkivät kilpailijoita alempiin myyntihintoihin erilaisilla tuottavuuden parannuskeinoilla – ja päätyvät siten myös maksimoimaan kuluttajan hyödyn suhteessa maksettuun hintaan, kun markkinahinta laskee. Tuottavuuden tavoitteellista parantamista voidaan kutsua **tuottavuuskehittämiseksi**.

Automaatiolla tarkoitetaan tietyn prosessin saattamista tapahtumaan koneellisesti, eli ilman ihmisen myötävaikutusta. Käytännössä automaatio toteutuu yleensä asteittain jo käytettävissä olevalla teknologialla. Kukin automaatioaskel edellyttää investointeja suunnitteluun, koneisiin ja laitteisiin, jotka pyritään rahoittamaan automaation tuomilla hyödyillä. Tällaisia tyypillisiä hyötyjä ovat säästöt työvoimakustannuksissa, suurempi tuotantokapasiteetti tai parempi laatu. Hyötyjen avulla pyritään kasvattamaan tuotannon tuottavuutta.

Prosessikehittäminen on tuottavuuskehittämisen muoto, joka perustuu tuotannon hahmottamiseen peräkkäisinä työvaiheina, joita voi tarkastella eri tarkkuustasoilla. Prosessien kehittäminen on tavoitelähtöistä toimintaa, jolla pyritään parantamaan organisaation tuottavuuden mittareita pilkkomalla ylätason prosessien vaiheet osavaiheiksi ja sitten järjestämällä tunnistetut osavaiheet uudelleen tehokkaammiksi ketjuiksi eli korvaaviksi prosesseiksi. Tähän tarkoitukseen prosesseja mallinnetaan prosessikaavioin, joissa prosessilla on määritellyt alku- ja loppupisteet, mutta prosessin osavaiheet eivät välttämättä etene janamaisesti järjestyksessä.

Prosessikehittämistä voi tehdä yksinkertaisesti muokkaamalla ihmistyön työvaiheita tehokkaammiksi prosesseiksi. 1900-luvun alussa tähän perustuva Frederick W. Taylorin ns. *tieteellinen liikkeenjohto* eli taylorismi oli teollisuusyhteiskunnan nousussa merkittävä tuottavuusinnovaatio. Tieteellisen liikkeenjohdon ajattelussa teollisen tuotannon käsityövaiheet purettiin yksityiskohtaisesti pieniin osatehtäviin, joiden järjestystä, tarvetta ja toteutustapaa

arvioitiin tarkkaan. Tämän tarkastelun pohjalta työvaiheet järjestettiin uusiksi sarjoiksi eli työntekijöiden työtehtäväkokonaisuuksiksi. Myös työvälineet suunniteltiin tehokkaammiksi.

Tietoteknisiä **sovelluksia** ovat esimerkiksi pikaviestinohjelmat, pelit ja verkkopankkisovellukset. Myös robottien käyttäytymistä ohjaavat järjestelmät koostuvat sovelluksista. Sovelluksista käytetään nykyään joskus myös termiä applikaatio (engl. *application).* Sovellukset saattavat toiminnoissaan tukeutua esimerkiksi tietokantapalveluihin, jolloin sovellusten toiminnallisuuden kaikkia osia ei välttämättä suoriteta tietyssä tietokoneessa, vaan laskenta voi olla hajautettua. Tällöin puhutaan nykyään usein pilvipalveluista.

Tietoteknisille sovelluksille on olemassa monenlaisia markkinoita. Osa sovelluksista kehitetään tietyn organisaation tarpeisiin, kun taas erityisesti pilvipalveluiden yleistymisen myötä on alettu tarjota asiakkaille yleiskäyttöisiä mutta kunkin asiakkaan tarpeisiin hienosäädettäviä valmisratkaisuja, jotka ovat skaalaetunsa vuoksi huomattavasti räätälöityjä sovelluksia edullisempia asiakkaille. Edulliset valmisratkaisut saattavat luonteensa vuoksi vuorostaan asettaa rajoitteita organisaatioiden prosessien kehittämisen vapaudelle, kun taas räätälöidyt sovellukset ja niiden muuttaminen edellyttävät suuria investointeja.

Palvelu- ja asiantuntija-alojen prosessikehitys tukeutuu nykyään usein tietoteknisiin sovelluksiin, joilla pyritään ohjaamaan ihmisten työskentelyä tehokkaammiksi prosesseiksi tai korvaamaan pieniä osia työvaiheista sovellusten toiminnoilla. Erilaiset toiminnanohjausjärjestelmät ja niitä muistuttavat sovellukset ovat nykyään merkittävä osa organisaatioiden prosessikehittämishankkeita.

Sovelluskehityksellä tarkoitetaan tavoitehakuista tiettyyn tarpeeseen vastaavan tietoteknisen ohjelmiston kehittämistä jollain menetelmällä. Sovelluskehittämisestä käytetään myös termiä ohjelmistokehitys ja ohjelmistotuotanto. Sovelluskehityksen lopputuotteena syntyy tietotekninen sovellus, jolla pyritään vastaamaan johonkin tunnistettuun tarpeeseen eli ratkaisemaan jokin ongelma.

Sovelluskehittämisen perimmäinen tavoite on parantaa jonkin tai joidenkin työvaiheiden tehokkuutta joko työajallisesti tai parantamalla työn tarkkuutta. Tässä mielessä sovelluskehittäminen on menetelmä, jota voidaan käyttää tuottavuus- ja prosessikehittämisen tukena.

Ohjelmointipohjainen sovelluskehitys perustuu tietokoneen ohjelmointiin sovelluskehittäjän eli ohjelmoijan toimesta. Sovelluskehittäjä ilmaisee ohjelmakoodilla sovelluksen toiminnallisuudet toteuttavat algoritmit eli tehtävien koneelliset suoritusohjeet, jotka muutetaan tulkkiohjelman eli kääntäjän avulla konekieliseen muotoon, jonka tietokoneen prosessori (suoritin) voi suorittaa.

Ohjelmointipohjainen sovelluskehitys eroaa luonteeltaan **oppimispohjaisesta sovelluskehittämisestä**, jossa sovelluksen toiminnallisuudet toteuttavat algoritmit synnytetään **koneoppimisen** avulla, eli ohjelmisto muokkaa itse algoritmiaan, kunnes algoritmi vastaa sovelluksen tavoitteita. Koneoppimisen erilaisia rakenteellisia lähestymistapoja on käytössä useita, mutta tässä opinnäytetyössä keskitytään keinotekoisiin neuroverkkoihin.

Digitalisointi (tai laajana yhteiskunnallisena ilmiönä **digitalisaatio**) viittaa tietoteknisten sovellusten hyödyntämiseen automaatiossa. Digitalisointia on perinteisesti toteutettu erilaisilla toimistotyön tietovirtoja sähköisesti käsittelevillä sovelluksilla ja laitteilla, jotka ovat vähentäneet rutiininomaisten toimistotyövaiheiden tarvetta. Organisaatioissa ei esimerkiksi ole enää pitkään aikaan käytetty juoksupoikia viestiliikenteeseen, vaan suuri osa tiedon kulusta on sähköistetty ja viestiliikennettä sen myötä digitalisoitu.

Digitalisoivia sovelluksia ovat esimerkiksi:

- Yritysten toiminnanohjausjärjestelmät (ERP)
- Markkinointiautomaatiojärjestelmät (MAP)
- Sisällönhallintajärjestelmät (CMS)
- Automaattiset raideliikennejärjestelmät (ATC)
- Verkkokaupat (eCommerce)
- Automaattiset sensoridataan reagoivat ravintolalaitteet
- Uber-taksikeskuspalvelu
- Jätehuollon esineiden internet (IoT) -ratkaisut

Digitalisointi organisaatioissa perustuu muun automaation tavoin liiketaloudellisiin arvioihin sovelluspohjaisen tuottavuuskehittämisen kannattavuudesta. Yksi huomattava liiketaloudellinen kannustin kehittää digitalisoinnin tarpeisiin tuotteistettavia sovelluksia on näiden sovellusten äärimmäisen edullinen kopioitavuus internetin kautta. Sovellusten jakelun edullisuus mahdollistaa suuren tutkimus- ja tuotekehityspanostuksen tuloksena syntyneen sovelluksen laaja-alaisen ja nopean käyttöönoton verrattuna esimerkiksi mekaanisiin tuotantokoneisiin, jotka edellyttävät valmistusmateriaaleja, tuotantolaitoksia, kuljetusta ja asennusta.

Jakelun edullisuuden vuoksi digitalisoinnin yhteydessä organisaatiot ovat usein valmiita luopumaan liiketoimintansa perinteisistä fyysisistä tai kosmeettisista piirteistä suuren tuottavuusloikan houkuttelemina. Verkkokaupat ovat tästä klassinen esimerkki: myymälän henkilökuntaa ei yritetä korvata palveluroboteilla, vaan kaupasta jää jäljelle vain edullisella maalla sijaitseva varasto, ja asiakas asioi kaupan kanssa yksinomaan tietokonekäyttöliittymän kautta postimyyntikatalogin tavoin. Saadut kustannussäästöt mahdollistavat hintojen alentamisen, minkä myötä tarpeeksi monet asiakkaat ovat valmiita luopumaan myymälässä asioinnin traditiosta. Asiakkuusvalinnoillaan he myös ohjaavat muita kilpailevia toimijoita parantamaan tuottavuuttaan.

3 Digitalisoinnin menetelmät

3.1 Ohjelmointipohjainen sovelluskehitys

Sovelluskehityksellä on perinteisesti tarkoitettu ohjelmointipohjaista sovelluskehitystä, jossa jonkin ihmiselle oleellisen ongelman ratkaisun löytävä algoritmi on kuvailtu tietokoneelle tekstimuotoisena ohjelmakoodina tai muunlaisina käskyinä, kuten reikäkorttiohjelmina. Ohjelmointipohjaisen sovelluskehittämisen keskeinen haaste liittyy tunnistetun sovellustarpeen pilkkomiseen algoritmisiksi ongelmiksi, joiden ratkaisut voidaan ilmaista tyhjentävästi ohjelmakoodina. Tässä mielessä ohjelmointi on ongelmanratkaisua. Jotta sovelluskehittäjä voisi ilmaista ongelmien tarkat algoritmiset ratkaisut, hänen on siis tunnettava ratkaistava ongelma erittäin hyvin.

Nykyään sovelluskehittäjien tarpeisiin on kehitetty suuri määrä ohjelmointityötä helpottavia työkaluja, jotka ehkäisevät ohjelmointivirheitä, nopeuttavat ohjelmointia ja auttavat syntyneiden ohjelmien testaamisessa. Näistä yhtenä esimerkkinä toimivat sovelluskehitysympäristöt eli IDE:t (engl. *integrated development environment*). Suuren osan tai kaikki työkaluista sovelluskehittäjä saa käyttöönsä internetin kautta veloituksetta. Lähes kaikissa kehittyneiden maiden kotitalouksissa on nykyään sovelluskehittämiseen soveltuvia tietokoneita. Internetin kautta on saatavilla myös ohjelmoinnin verkkokursseja aloittelijoille sekä kattavat dokumentaatiot ohjelmistokirjastoista. Sovelluskehittäminen ei siis vaadi suuria alkupanostuksia työvälineisiin, vaikka se edellyttääkin opettelua.

Ohjelmointipohjaisen sovelluksen kehityskustannukset tapaavat kasvaa sovelluksen ohjelmakoodin rakenteellisen monimutkaisuuden kasvaessa (Banker, Datar, Kemerer & Zweig 1993, 89). Tämä johtunee siitä, että ohjelmakoodin monimutkaisuuden kasvaessa sen ymmärrettävyys ohjelmoijalle laskee, jolloin sovelluksen suunnittelu, ohjelmointi ja oikeellisen toiminnan todentaminen vaikeutuu (Ogheneovo 2014, 2).

Ohjelmointikokemus vaikuttaa kuitenkin ohjelmoijan ohjelmointiin kuluvaan aikaan lyhentävästi, eli tuottavuutta kasvattaen. Yksittäisen ohjelmoijan tuottavuuteen voidaan havaita vaikuttavan sekä ohjelmistokehityskokemus samasta sovelluksesta jota hän kehittää että kokemus vastaavanlaisten sovellusten kehittämisestä, mutta myös kokemus kehitettävään sovellukseen liittymättömistä sovelluksista. Sovelluskehittäjätyöryhmien tuottavuuden kohdalla on vastaavanlaiset mutta heikommat yksilöiden kokemusvaikutukset. (Boh, Slaughter & Espinosa 2007, 1324-1328).

Teknisesti laadukas sovelluskehitys on siis hyvin aikaa vievää ja sen kustannustehokkuus edellyttää asiantuntijoita. Erityisesti kun sovelluksilta edellytetään laajaa toiminnallisuutta, sovelluskehitys tarkoittaa suuria henkilöstömenoja ja siten suuria taloudellisia investointeja sovelluskehittämiseen. Investointien vastineeksi sovelluksilta odotetaan suurta tuottavuushyötyä.

Ohjelmointipohjaisen sovelluskehittämisen kustannukset saattavat muodostua sovelluspohjaisen tuottavuuskehittämisen esteeksi organisaatioissa. Esimerkiksi prosessikehittämisen edellytyksenä olevat toiminnanohjausjärjestelmien muutostarpeet saattavat johtaa ylläpitokulujen merkittävään nousuun (Sandberg 2008, 3). Lisäksi toiminnanohjausjärjestelmien muutosprojektit tapaavat laskea toiminnanohjausjärjestelmähankkeiden onnistumisprosenttia, eli kasvattavat kustannusriskiä (Hong & Kim 2001, 37). Organisaatiolle voi siis käydä niin, että sillä ei ole varaa uudistaa toimintatapojaan sovelluspohjaisen prosessikehittämisen puitteissa.

Ohjelmointipohjaisella sovelluskehityksellä tuotetun sovelluksen oikeellinen toiminta voidaan todentaa testaamalla sovellusta eri syötteillä. Lisäksi ratkaisu johonkin tiettyyn ongelmaan voidaan todentaa oikeelliseksi ja turvalliseksi esimerkiksi viranomaishyväksyntää varten testaamalla sovellusta sekä lukemalla ratkaisun ohjelmakoodi, mutta tämä edellyttää merkittävää asiantuntijatyöpanosta ja tarkkuutta. Ohjelmointipohjaisen sovelluksen testaamisessa tai ohjelmakoodin tarkastamisessa löydetty virhe voidaan korjata muuttamalla ohjelmakoodia.

Työmarkkinoilla on hyvä tarjonta muodollisesti ohjelmointitaitoisia ihmisiä, mutta asiantuntijoiden saatavuutta rajoittaa kuitenkin työmarkkinapula erityisosaajista samaan aikaan, kun erikoistumattomista on ylitarjontaa. Esimerkiksi Iso-Britanniassa erikoistuneiden IT-asiantuntijoiden pula selitti 21 % työpaikoista, joihin ei löydetty työntekijöitä vuonna 2011 (Hollingworth & Harvey-Price, 2013, 8). Syyskuussa 2016 Suomen IT-alan työttömistä työnhakijoista (pl. opistotason tutkinnot) 74,6 %:lla oli vain tietojenkäsittelyn ammatillinen tutkinto, kun taas alemman korkeakoulututkinnon suorittaneet edustivat vain 12,4 % työttömistä (Neittaanmäki & Kinnunen 2016, 6).

IT-sektorin kasvusta huolimatta Iso-Britanniassa vuosien 2002 ja 2010 välillä tietojenkäsittelyn korkeakoulutuksiin hakeutuvien lukumäärä oli laskenut 28 %, kun korkeakoulutukseen ylipäätään hakeneiden lukumäärä oli samaan aikaan kasvanut 51 %. 2010-luvun vaihteessa tietojenkäsittelyn korkeakoulututkinnon suorittaneilla Iso-Britanniassa oli kor-

kein työttömyysaste kuusi kuukautta valmistumisesta. Iso-Britanniassa IT-alan työnantajien järjestämän tehtävään perehdyttävän koulutuksen määrä oli 2010-luvun vaihteessa laskussa, jota työnantajat perustelivat koulutuksen järjestämiseen tarvittavan rahoituksen puutteella. (Hollingworth & Harvey-Price 2013, 9-10.)

Ohjelmointipohjaisen sovelluskehityksen työntekijöiden työmarkkinasaatavuus ei kuitenkaan ole täysin riippuvainen koulutuksen määrästä, ja työuran aloittaminen edellyttää monia asiantuntija-ammatteja lyhyempää muodollista kouluttautumista. Sovelluskehittäjien StackOverflow -portaalin (2016) ammattilaiskyselyn mukaan suurimmat alan ikäryhmät muodostivat 25-29 -vuotiaat ja 20-24 -vuotiaat, enemmistö (69 %) ammattilaisista on ainakin osittain itseoppineita, alle puolella (43,2 %) on suoritettuna jokin tietojenkäsittelyyn liittyvä alempi korkeakoulututkinto ja vain noin puolella ammattilaisista (49,7 %) on yli viisi vuotta alan työkokemusta. Monet alan ammattilaiset ovat siis hankkineet taitonsa muualta kuin muodollisesta koulutuksesta.

3.2 Koneoppimispohjainen sovelluskehitys

Toisen maailmansodan jälkeen vuonna 1950 Alan Turing julkaisi artikkelin ajattelevasta tietokoneesta ja kehitti ns. Turingin testin tällaisen todentamista varten: jos tietokone voisi ihmisen kanssa keskustelemalla saada ihmisen uskomaan, että hän keskustelee ihmisen kanssa, tietokone olisi läpäissyt testin ja olisi siten älykäs. 1950-luvulla tekoälytutkimukseen suhtauduttiin optimismilla ja tehtiin ennusteita, että vain muutaman vuoden kuluttua tietokoneet voisivat olla yhtä älykkäitä kuin ihmiset, ja esimerkiksi toimia kielenkääntäjinä ja ohjata autoja. Nämä ennusteet osoittautuivat pian liiotelluiksi. 1950-luvun alkuinnostuksen jälkeen tekoälytutkimus siirtyi hiljalleen kohti realistisempaa näkökulmaa, jossa keskityttiin kehittämään ihmisaivojen toimintaperiaatetta jäljitteleviä algoritmeja ja menetelmiä, joilla voitaisiin ratkaista ihmismäisiä ongelmia. (Coppin 2004, 7-9.)

Viimeistään 1970-luvulla alkoi tulla selväksi, että Turingin kuvitteleman yleistekoälyn kehittäminen oli erittäin haastava tavoite, eikä suuria tuloksia aiempien lupausten mukaisesti syntynyt (BBC 2014). Sen sijaan tekoälyä pyrittiin 1970- luvulla soveltamaan ns. asiantuntijajärjestelmien kehittämiseen sääntölogiikan avulla (Foote 2016). 1970- ja 1980-luvuilla tutkimusala koki kuitenkin pysähtyneisyyttä (Niu, Tang, Xu, Zhou & Song 2016, 2.)

Vuonna 1980 filosofi John Searle julkaisi artikkelin, jossa hän kyseenalaisti Turingin testin käyttökelpoisuuden ja Turingin visioiman ihmismäisesti ajattelevan yleistekoälyn (engl. *strong artificial intelligence* tai *artificial general intelligence*) olemassaolon mahdollisuuden

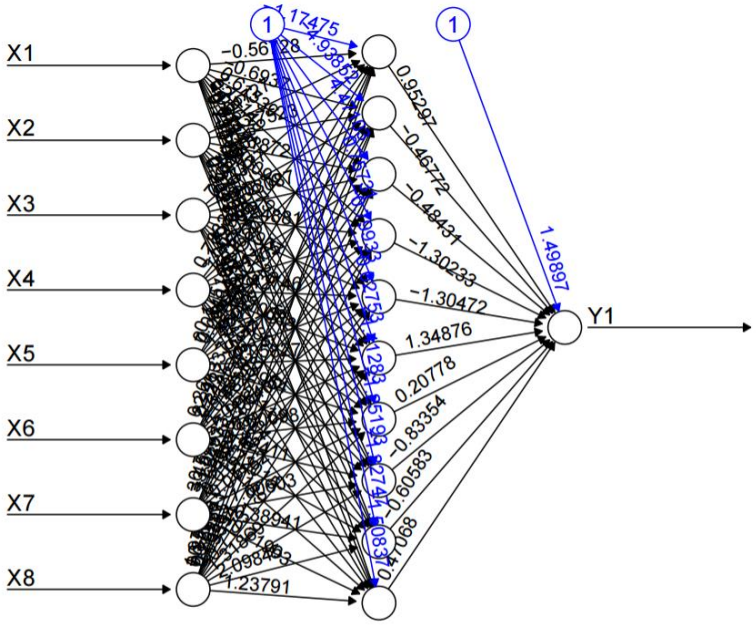
ns. kiinalaisen huoneen ajatusleikillä. Searlen keskeinen väite oli, että tosielämän sovellustarpeita varten voitaisiin kehittää sovelluskohtaisia ohjelmia, jotka läpäisisivät Turingin testin, mutta todellisuudessa vain mekaanisesti ratkaisisivat sovelluskohtaisen ongelman ymmärtämättä itse ongelmaa ja kykenemättä ratkaisemaan muita kuin valitun sovelluksen ongelmia. Searlen ajatusleikkiä vastaan on esitetty useita vasta-argumentteja. (Coppin 2004, 20-21.)

1980-luvun lopussa Yhdysvaltain asevoimien tutkimuslaitos DARPA julisti, ettei tekoäly olisi seuraava suurilmiö ja mm. tämän seurauksena tekoälytutkimuksen rahoitusta leikattiin kovasti. Tämän jälkeen tekoälytutkimus luopui asiantuntijajärjestelmien loogisen päättelyn tutkimussuuntauksesta ja keskittyi osin kehittämään ns. älykkäitä agentteja (engl. *intelligent agent*) tai botteja (engl. *bot*), eli ohjelmia, jotka avustavat ihmisiä esim. tietomäärän seulomistarpeissa. (Foote 2016.)

Tietojenkäsittelytieteen tutkija Hans Moravecin mukaan yleistekoälyn kehittäminen olikin edennyt väärin yrittämällä sitä ihmisajattelun näkökulmasta ”ylhäältä-alas” (engl. ”*top-down*”) eli keskittymällä suoraan älyllisten taitojen kuten loogisen päättelyn koneellistamiseen sääntöjen ohjelmoinnilla. Tekoälytutkimuksen eteneminen edellyttäisi Moravecin mukaan ”alhaalta-ylös” -ajattelua (engl. ”*bottom-up*”) eli ensin ympäristön havainnointiin ja ympäristössä liikkumiseen liittyvien ongelmien ratkaisua, koska samat kognition ongelmat evoluutio joutui ensin ratkaisemaan maapallon varhaisten eläinten kohdalla. (Moravec 1988, 16-17.)

Moravecin tavoin, autonomisen robotiikan tutkija Rodney Brooks kritisoi vuonna 1990 artikkelissaan *Elephants don’t play chess* asiantuntijajärjestelmien ja tekoälytutkimuksen edellisvuosikymmenten rationaalisuuteen pyrkivää ylhäältä-alas -ajattelua ja suositteli aisteihin ja vaistoihin pohjaavaa alhaalta-ylös -ajattelua, jota hän kutsui ”uudenlaiseksi tekoälyksi” (engl. ”*nouvelle AI*”) ja perusteli sitä tekoälyn tarpeella tuntea fyysinen maailma, jossa sen tulisi toimia (Brooke, 1990, 3, 5). Brooken kritiikin myötä myös kiinnostus ns. keinotekoisiin neuroverkkoihin alkoi palautua (BBC 2014).

Keinotekoiset neuroverkot (engl. *artificial neural network, ANN*) ovat matematiikan ns. yhdistävän laskennan funktiomalleja, jotka imitoivat aivojen synaptista, verkottuvaa rakennetta. On vaikea sanoa, miten paljon nykyisenlaiset keinotekoiset neuroverkot todellisuudessa muistuttavat aivojen luonnollisia neuroverkkoja eli aivosolujen keskinäisen viestinnän rakennetta, mutta niillä pyritään löytämään mm. inhimillistä ratkaisua muistuttavia algoritmeja.



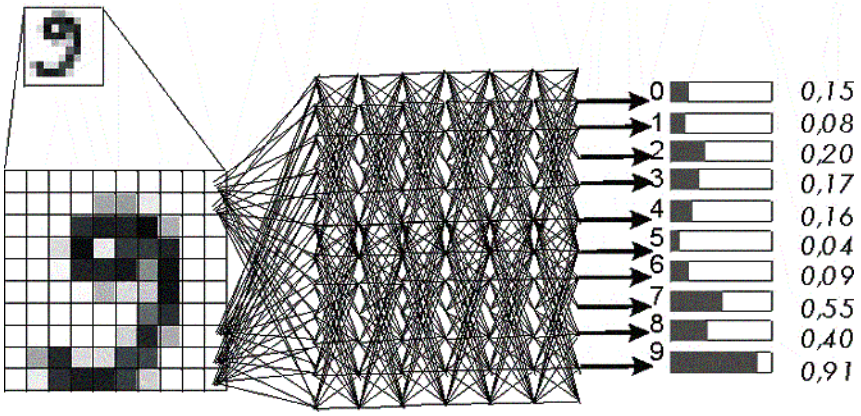
Kuvio 1. Visuaalinen esitys neuroverkosta painotusarvoineen (Beck 2013)

Keinotekoinen neuroverkko vastaanottaa syötedataa syöteneuroneiksi kutsuttujen solujen kautta. Syöteneuronien lukumäärä ja tekniset rajoitteet määräävät syötedatan tarkkuuden suhteessa alkuperäiseen lähteeseen aivan kuten ohjelmoinnissakin funktion syötteen täytyy sisältää se informaatio, jonka pohjalta tuloste syntyy. Syöteneuronit on kuviossa 1 merkitty X-kirjaimella alkavalla numeroinnilla. X-numeroitua aluetta kutsutaan syötekerrokseksi. Syöteneuroneista on nuolten kuvaamat yhteydet eli synapsit seuraavan kerroksen neuroneihin. Synapsit suorittavat välittämälleen datalle jonkin matemaattisen funktion.

Tyypillisesti keinotekoisissa neuroverkoissa syötekerrosta seuraa yksi tai useampi ns. piilokerros, joskin sellaisiakin verkkoja voidaan muotoilla, joissa ei ole piilokerrosta lainkaan. Lopuksi on tuloskerros (Kuvio 1, solu Y1), joka ilmaisee jollain tarkkuudella edellisten yhteyksien funktioiden tuottaman informaation. Tuloskerroksen neuronien määrä vaihtelee verkkotyypistä toiseen, mutta niitä on verkossa vähintään yksi. Kokonaisuutena verkko muodostaa jonkin matemaattisen funktion syötearvoille, joka monimutkaisempien verkkojen kohdalla on vaikea tarkasti selvittää. Erilaisia neuroverkkojen rakenteita on kehitetty, nimetty ja tutkittu useita.

Hyöty tällaisten verkkojen kehittämisessä saadaan siitä, että muuttamalla niiden synapsien suorittamia funktioita eli painotuksia syötedatalle kunnes verkon tila tuottaa jotain ennalta määrättyjä toivottuja tuloksia tuloskerrokseen, verkko voidaan ajan mittaan saattaa tilaan, jossa se muodostaa testaamalla hyödylliseksi todennettavan funktion syötedatalle. Tällöin verkko on ratkaissut ongelman. Tätä verkon funktion kehittämistä kutsutaan verkon opettamiseksi.

Kuviossa 2 on esitetty havainnollistus yksinkertaisesta neuroverkosta, joka tulkitsee käsin piirrettyjä numeroita siten, että kuva numerosta on jaettu 10*10 pikseliruudukoksi, jonka jokaista pikseliä vastaa yksi syöteneuroni. Syöteneuronin syötteenä on pikselin tummuusarvo vaihteluvälillä 0-255. Verkon opettamisen jälkeen se on saatettu hyödylliseen tilaan. Tulosneuronikerroksessa (Kuvio 2, oikealla) verkko ilmaisee jokaista numerosymbolia (0-9) kohden itsevarmuusarvion siitä, että kyseessä on symbolia vastaava numero. Tässä tapauksessa verkko esittää, että kyseessä todennäköisesti on numero 9. (Ingo 2002.)



Kuvio 2. Käsinkirjoitettujen numeroiden tunnistaminen monikerroksisella neuroverkolla (Ingo 2002)

Keinotekoisiin neuroverkkoihin tukeutuvan tekoälytutkimuksen kriittiseksi haasteeksi muodostui kauan niiden opettamiseen vaadittavan laskentatehon saatavuus ja markkinahinta. Laskentatehon hinnan lasku ja saatavuuden paraneminen saattoivat olla keskeisiä tekijöitä tutkimusalan tutkimusartikkeleiden vuositason kasvun takana 2000-luvun puolivälistä alkaen, joka mahdollisti laskennallista vaativaa datan käsittelyä, analyysiä ja visualisointia. Toinen selitys liittyy 2000-luvun puolivälin aikana kasvaneeseen mielenkiintoon ns. syvään oppimiseen, jossa neuroverkkoja alettiin tutkia lisäämällä niihin useampia kuin yksi tai kaksi piilokerrosta. Muita vaikuttavia tekijöitä saattoivat olla tutkimusalan kehittyminen ratkaisukeskeisemmäksi, tutkimuksen tieteellisen laadun paraneminen sekä poikkitieteelliset synergiahyödyt eri tutkimusalojen mutta samoja ongelmia tutkivien välillä. (Niu, Tang, Xu, Zhou & Song 2016, 2, 11-14.)

Keinotekoisten neuroverkkojen tutkimuksessa oltiin 2000-luvulle tultaessa hylätty pyrkimys imitoida biologisia neuroverkkoja ja keskitytty kehittämään sellaisia neuroverkkoja, joiden avulla onnistuttiin ratkaisemaan haluttuja ongelmia. Kuitenkin vielä 2000-luvun alussa keinotekoisten neuroverkkojen käyttökelpoisuuteen tekoälytutkimuksessa suhtauduttiin skeptisesti. (Littmann 2002, 1-2, 9.)

Keinotekoisten neuroverkkojen vahvuus on siinä, että niiden avulla voidaan toteuttaa sovelluksia, joita ei muuten voitaisi kehittää joko tiedon puutteen vuoksi vaadittavasta algoritmista tai koska ohjelmointipohjainen ratkaisu olisi kallis tai monimutkainen toteuttaa (Littmann 2002, 1). Keinotekoiset neuroverkot ja niitä muistuttavat koneoppimistekniikat vastaavat luonteeltaan Moravecin ja Brooksin 1990-luvun vaihteessa peräänkuuluttamaan alhaalta ylös -ajatteluun, joka sittemmin on esimerkiksi neuroverkkojen muodossa menestynyt huomattavasti 1970- ja 1980-luvun ohjelmoituja asiantuntijajärjestelmiä paremmin menetelmänä aikaansaada käyttökelpoisia sovelluksia. Kuitenkin vasta vuosina 2015 ja 2016 neuroverkkoteknologia on tehnyt populaarimedian uutiskynnyksen ylittäviä läpimurtoja.

Näistä yksi merkittävä liittyi AlphaGo-nimiseen Google DeepMind -hankkeen tekoälysovellukseen, joka maaliskuussa 2016 voitti Go-lautapelin maailmanmestarina pidetyn Lee Sedolin. Strategisen ja luovan luonteensa vuoksi, go-lautapeliä pidettiin kauan erittäin haastavana automaatiokohteena tekoälytutkimuksessa. (Tamminen & Pesälä 2016, 41-44.)

AlphaGo -sovelluksen neuroverkko koostui 17 328 syöteneuronista ja vähintään 13 piilokerroksesta (Silver & Huang 2016, liite, 2). Hyödyllisten neuroverkkojen suuri koko ja niiden toimintojen emergentti luonne tekee niistä vaikeasti havainnollistettavia ja vaikeasti ymmärrettäviä ns. mustia laatikoita (Coppin 2004, 292).

Kuten kuvioista 1 ja 2 käy ilmi, neuroverkon esittäminen painotuksineen ihmiselle ymmärrettävässä muodossa käy solumäärän kasvaessa hyvin haastavaksi, kun taas ohjelmakoodin rivimäärän kasvu ei aiheuta vastaavanlaista hahmottamisen haastetta; ohjelmistoteollisuus on onnistuneesti kehittänyt ohjelmakoodirivimäärässä mitattuna hyvinkin laajoja ohjelmistoja ja sovelluksia, joita käytämme arkisesti.

Neuroverkon kehittämän algoritmin hahmottamisen vaikeus voi tuottaa haasteita sellaisten sovellusten toteuttamisessa, jotka tarvitsisivat esimerkiksi ohjelmakoodin viranomaissertifiointia, kuten jotkut lääketieteelliseen käyttöön hyväksyttävät laitteet saattavat edellyttää. Tällöin toiminnan oikeellinen todentaminen voitaisiin tehdä vain testaamalla sovellusta erilaisilla syötteillä.

On esimerkiksi havaittu, että ainakin visuaalisen informaation tulkintaan käytetyt neuroverkot saattavat käyttäytyä arvaamattomasti, kun niiden syötteeseen tehdään hyvin pieniäkin muutoksia, joita ihminenkään ei välttämättä huomaa. Turvallisuushuolet neuroverkkojen käytöstä tehtävissä kuten auton ajamisessa on nostanut tutkimusalueeksi oppimispohjais-

ten sovellusten tarpeen selittää omaa päättelyään jälkikäteen. Oppimispohjaisten sovellusten turvallisuuden todentaminen on kuitenkin edelleen alkutekijöissään oleva tutkimusala. (Huang, Kwiatkowska, Wang & Wu 2017, 2-3.)

Oppimispohjaisen sovelluskehityksen hyöty on kuitenkin se, että se tarjoaa ratkaisujen tavoittelumenetelmän sellaisissa ongelmissa, joissa ihmisen on erittäin vaikea tai mahdotonta määritellä ongelman hyvin ratkaiseva algoritmi ohjelmallisesti, mutta ratkaisun oikeellisuus voidaan kuitenkin todentaa jollain tapaa. Esimerkkejä tällaisista neuroverkkojen sovelluskohteista ovat mm. ns. sumean logiikan ja tilastollisen päättelyn tehtävät, joiden ongelmien ratkaisuun sovelluskehittämisessä on kasvava tarve (Harman 2012, 2). Digitalisoinnin näkökulmasta oppimispohjaisella sovelluskehityksellä voidaan siis pyrkiä automatisoimaan sellaisia työvaiheita, joiden ohjelmointi on vaikeaa, mutta jotka ihmiset osaavat jo itse suorittaa ja siksi heillä on jonkinlainen käsitys siitä, miltä oikea tulos näyttää. Silloin neuroverkon tulosta voidaan verrata tähän oikean tuloksen ennakkokäsitykseen.

Vaikka tällaiset oppimispohjaisen sovelluskehityksen tuotokset eivät olisi Turingin testin mielessä älykkäitä, ne voivat silti ratkaista ohjelmistokehityksessä sellaisia osaongelmia, joita emme ohjelmoinnillisesti ole syystä tai toisesta kyenneet ratkaisemaan tyydyttävästi tai ollenkaan. Oppimispohjainen sovelluskehitys voi siis avata täysin uusia mahdollisuuksia ohjelmistoteollisuudelle ja siten sovellusperustaiselle tuottavuuskehittämiselle.

Perinteisestä ohjelmointiin perustuvasta sovelluskehittämisestä ja tekoälytutkimuksesta poiketen, oppimispohjainen sovelluskehitys on täysin riippuvaista oppimisesta. Oppimispohjainen sovelluskehitys siis liittyy keskeisesti sovelluksen verkon opettamiseen, eli algoritmin oikeaan tilaan virittämiseen. Oppimispohjaisen sovelluskehityksen käyttämät tekniikat ovat edelleen monella tapaa kehittymättömiä ja vaativat suuria määriä laadukasta dataa hyödyllisen funktion aikaansaamiseksi. Siksi oppimispohjaisen sovelluskehittämisen rajoitteeksi muodostuu laskentatehon hinnan ja saatavuuden lisäksi laadukkaan opetusdatan hinta ja saatavuus. (Tamminen & Pesälä 2016, 38.)

Laadukkaan opetusdatan tuottaminen ihmistyönä olisi hyvin kallista tarvittavaan datan määrään nähden, mutta erilaisista joukkoistamis- ja mikrotyöpalveluista saattaa olla hyötyä tässä. Keinotekoisten neuroverkkomallien ja näiden oppimisparadigmojen kehittäminen saattaa tulevaisuudessa merkittävästi poistaa oppimispohjaisen sovelluskehittämisen nykyisiä resurssihaasteita.

Oppimispohjaisen sovelluskehittämisen uutuus liiketoiminnan alana ja voimakas kasvu vuosina 2015-2016 vaikuttaa asiantuntevan työvoiman rekrytointiin. Taitavista datatutkijoista ja koneoppimisen asiantuntijoista on merkittävä pula työmarkkinoilla, joka kasvaa edelleen (Ghosh 2017). Oppimispohjaisen sovelluskehittämisen työkalut ovat alkutekijöissään ja kehittyvät jatkuvasti, joten liiketoiminnallisesti hyödyllisten ja kilpailukykyisten sovellusten kehittäminen edellyttää alan ammattilaisilta korkeaa asiantuntemusta ja jatkuvaa perehtymistä. Se siis vertautuu paremmin tutkimustyöhön kuin tyypilliseen ohjelmointityöhön.

Taulukossa 1 on tiivistelmä ohjelmointipohjaisen ja oppimispohjaisen sovelluskehityksen tässä läpikäynnissä tunnistetuista eroavaisuuksista. Menetelmien vertailusta voinee päätellä, että oppimispohjainen sovelluskehitys ei ainakaan välittömässä lähitulevaisuudessa ole menetelmä, jolla voitaisiin vastata kaikkiin sovelluskehittämisen tarpeisiin, mutta jolla kuitenkin voidaan ratkaista tiettyjä ohjelmoinnillisesti haastavia ongelmia.

Taulukko 1: Ohjelmointipohjaisen ja oppimispohjaisen sovelluskehittämisen eroja

	Ohjelmointipohjainen sovelluskehitys	**Oppimispohjainen sovelluskehitys**
Kehitystyön aloituskustannus ja investoinnit	Kehitystyö voidaan aloittaa ilman merkittäviä alkuinvestointeja. Liiketoiminnallisesti arvokkaat suuret sovellukset edellyttävät kuitenkin asiantuntevaa ohjelmistosuunnittelutyötä, joka vaikuttaa myöhemmin kehityskustannuksiin.	Kehitystyö edellyttää alkuinvestointia ratkaistavan ongelman tutkimukseen, oppivan sovelluksen suunnitteluun sekä soveltuvan opetusdatan paikantamiseen tai tuotantoon. Nämä alkuvaiheen tekijät määräävät sovelluskehityksen lopputuloksen laadun.
Henkilöstökustannukset	Henkilöstömenot ovat suurin menoerä. Henkilöstömenot kasvavat sovelluksen ohjelmakoodin laajuuden kasvaessa	Kehitystyö vaatii korkeaa osaamista, muttei välttämättä suurta henkilöstöä. Työ muistuttaa tutkimustyötä.
Muut merkittävät kustannukset	Kehitystyön aikaisen menorahoituksen korkokustannukset ja kustannusriski	Laskentateho ja laadukas opetusdata
Asiantuntijoiden saatavuus	Erikoistuneista asiantuntijoista on pulaa, mutta tarjonta on muutoin hyvä	Heikko ja heikentyvä
Ongelmanratkaisun vahvuusalue	Ongelmat, joiden algoritmisia ratkaisuja tunnetaan entuudestaan tai joiden ratkaisu voidaan kustannustehokkaasti muotoilla ohjelmakoodiksi	Ongelmat, joille ei tunneta algoritmista ratkaisua tai sen kehittäminen olisi kallista ja vaikeaa
Erityishaasteet	Kehittämisen työvoimaintensiivisyys sekä siitä seuraavat menetelmälliset ja organisaatiolliset haasteet	Kehitettyjen algoritmien musta laatikko -piirre, joka tekee algoritmien oikeellisen toiminnan todentamisesta ja jälkikäteisestä muuttamisesta vaikeaa. Oikeellisen toiminnan todentaminen on merkittävä turvallisuustekijä.

4 Työtehtävien digitalisoitavuuden arvioiminen

Voidaksemme arvioida nykyisen työelämän työtehtävien digitalisoitavuutta sovelluskehityksen menetelmin tarvitsemme vastauksia seuraaviin kysymyksiin:

Millaisia toisistaan rakenteellisesti poikkeavia tehtäviä työelämässä on?
Mikä on niiden automaation kvantifioitava haastavuus?
Mikä on tehtävien digitalisoimisen taloudellisuus?

4.1 Laskennallinen tehtäväteoria

Thórisson ym. (2016, 4-5) on arvioinut, että kattavan laskennallisen tehtäväteorian tulisi kattaa kaikki tehtävien piirteet ja ratkaisuympäristöt. Tämän pohjalta Thórisson ym. esittää, että tehtäväteorian tulisi antaa eväitä (suom. Lilja Tamminen):

- Samanlaisten ja erilaisten tehtävien vertailuun
- Tehtävien elementtien ja kokonaisuuksien abstraktioon ja konkretisointiin
- Tehtävän suorittamisen resurssivaatimuksien kustannusten kuten ajan, energian ja virheiden kustannuksen ym. arviointiin sekä resurssien saatavuuden arviointiin
- Tehtävien kompleksisuuden luonnehdintaan tiedon saatavuuden rajallisuuden, oppimispalautteen saamisen keston sekä tehtävänannon luonteen mukaan
- Tehtävien purkamiseen osatehtäviin ja näiden atomisiin elementteihin
- Tehtäväkokonaisuuksien luomiseen eri tavoin

Mitään tällaista tarkkaa ja kattavaa laskennallista tehtäväteoriaa ei ole toistaiseksi kehitetty, joten Thórissonin listaamia ehtoja täytyy pyrkiä tyydyttämään approksimoimalla.

Työtehtäviä on tutkittu laajasti sosiologiassa ja käyttäytymistieteissä. Tästä huolimatta tutkimuskirjallisuuteen ei ole syntynyt konsensusta työtehtävän teoreettisesta luonteesta. Silti useissa tehtävien vaativuutta koskevissa tutkimuksissa on havaittu, että tehtävien vaativuudesta esitetyt arviot ovat ennustaneet ihmisten suoriutumisastetta erinäisistä tietotyön tehtävistä. Osana aiempaa työtehtävien tutkimusta on kehitetty erilaisia malleja työtehtävien vaativuuden arvioimiseksi. Mallit ovat keskittyneet tehtävien keskinäiseen vertailuun ja työn kuormittavuustekijöiden tunnistamiseen. Osa tutkimuksesta on keskittynyt hahmottamaan tehtävän vaativuutta itse tehtävän rakenteellisten seikkojen kautta. Toinen tutkimuksen haara on ollut hahmottaa vaativuutta resurssivaativuuden kautta, kuten kognitiivisten, fyysisten ja henkisten kyky- ja kapasiteettivaatimusten kautta. Kolmas tutkimushaara on tutkinut tehtävien ja niiden suorittajien välistä suhdetta, johon on vaikuttanut

aiempi kokemus tehtävästä, tieto, tehtävän tuttuus ja yksilökohtainen vaihtelu. Tutkimustyön seurauksena on syntynyt sekä subjektiivisuuteen perustuvia keinoja arvioida työn kuormittavuutta että kirjallisuutta siitä, mihin työn vaativuuden kokemus voisi objektiivisesti perustua. Työn kuormittavuuden taustalla piileviä rakenteellisia (objektiivisia) tekijöitä on lukuisissa tutkimuksissa yritetty hahmottaa vaativuuden ulottuvuuksia tunnistamalla. (Liu & Li 2012, 553-557.)

Kirjallisuuden subjektiivisista arviointimenetelmistä yhtenä esimerkkinä toimii kyselypohjainen Yhdysvaltain ilmailu- ja avaruushallintovirasto NASAn kehittämä tehtävien kuormittavuusindeksi (engl. *task load index*), tunnetummin NASA-TLX.

NASA-TLX perustuu työntekijän arvioihin työnkuvan vaativuudesta eri osa-alueilla: älyllinen kuormittavuus, fyysinen kuormittavuus, aikapaine, turhautumisaste, vaivannäkö ja onnistuneisuuden kokemus. NASA-TLX:ää on opetettu laajasti korkeakouluissa ja sitä sovelletaan vaativissa olosuhteissa suoritettavien tehtävien kuormittavuuden arviointiin. Vuonna 2006 lähes neljännes NASA-TLX:ää silloisesta tutkimuskirjallisuudesta käsitteli kyselyjen tuottaman datan pisteyttämiskeinoja sekä niiden vertailua ja arviointia. Yksi tutkimuskirjallisuudessa esiin noussut tekijä on, että monet kyselyn kuormittavuustekijöistä korreloivat keskenään, eli mittaavat osin samoja asioita. (Hart 2006, 1-4.)

Kuvio 3. Liu & Li:n (2012, 564) arvio tehtävän vaativuuden ulottuvuuksista

Complexity dimensions.

Dimensions	Definition
Size	Number of task components.
Variety	Diversity in terms of the number of distinguishable and dissimilar task components.
Ambiguity	Degree of unclear, incomplete, or non-specific task components.
Relationship	Interdependency (e.g., conflict, redundancy, dependency) between task components.
Variability	Changes or unstable characteristics of task components.
Unreliability	Inaccurate and misleading information.
Novelty	Appearance of novel, irregular and non-routine events (e.g., interruption) or tasks that are not performed with regularity.
Incongruity	Inconsistency, mismatch, incompatibility, and heterogeneity of task components.
Action complexity	Cognitive and physical requirements inherent in human actions during the performance of a task
Temporal demand	Task requirement caused by time pressure, concurrency between tasks and between presentations, or other time-related constraints.

Kuviossa 3 on esitetty laajaan kirjallisuuskatsaukseen perustuva arvio työn vaativuuden ulottuvuuksista. Näistä Liu & Li:n tehtävien rakenteellisista ulottuvuuksista tehtävän laajuus (engl. *size*), tehtävän epämääräisyys (engl. *ambiguity*), riippuvuussuhteet (engl. *relationship*), epäluotettavuus (engl. *unrealiability*), kognitiivinen haastavuus (engl. *action complexity*) ja aikavaativuus (engl. *temporal demand*) kytkeytyvät jokseenkin Thórissonin ym. listaamiin tehtäväteorian resurssivaatimusten ja kompleksisuuden arvioinnin tekijöihin kuten tiedon saatavuuden rajallisuuteen, tehtävänannon luonteeseen ja virheiden ja ajan kustannuksiin. Liu & Li:n lista (Kuvio 3) tunnistaa tehtävien keskinäisen erilaisuuden (engl. *variety*) merkityksen tehtävän suorittajan kannalta näyttäytyvänä vaativuustekijänä, muttei ota kantaa erilaisten tehtävien vertailuun tai atomiseen pilkkomiseen, jota Thórisson peräänkuuluttaa.

Lisäksi työtutkimuksen hyödyntämisen haasteeksi digitalisoitavuuden arvioinnissa muodostuu keskeisesti se, että olemassa oleva kirjallisuus työtehtävien vaativuudesta keskittyy tunnistamaan tekijöitä, jotka saavat tehtävät tuntumaan raskailta ihmisten näkökulmasta. Ihmisen subjektiivinen kokemus työvaiheen tai tehtävän kuormittavuudesta tai vaativuudesta ei kuitenkaan ole suoraan vertailukelpoinen tehtävän koneelliseen tietojenkäsittelylliseen haastavuuteen eli laskennalliseen kompleksisuuteen. Haasteen taustalla on se, että ihmisaivot muiden eläinten tavoin ovat evoluution ja viime kädessä varttumisensa myötä erikoistuneet ratkaisemaan teoreettisen ongelma-avaruuden tiettyjä osa-alueita paremmin kuin toisia, mikä voidaan havaita esimerkiksi vertailemalla ihmisen päässälaskun kykyjä laskimeen ns. raa'assa laskennassa.

Työtehtävien rakenneanalyysejä digitalisoitavuuden kontekstissa ovat aiemmin esittäneet Lampelto (2013) ja McKinsey & Company -konsulttiyhtiön tutkijat Chui, Manyika ja Miremadi (2017).

Chui, Manyika ja Miremadi tutkivat teknologisen automaation potentiaalia tunnistamalla ammattitehtävistä komponenttisia työtehtäviä, joiden suorittamisesta työntekijöille maksetaan. Näiden työtehtävien pohjalta eriteltiin 18 kykyä (vastedes CMM-kykyä), joita arvioitiin tarvittavan näiden työtehtävien suorittamiseksi (Kuvio 4). CMM-kyvyt jaettiin kategorioihin, jotka olivat aistiperusteinen havainnointi, kognitiiviset kyvyt, luonnollisen kielen prosessointi, sosiaaliset ja emotionaaliset kyvyt sekä fyysiset kyvyt. CMM-kyvyistä työelämässä vaadittavan suoriutumisen tason selvittämiseksi arvioitiin tyypillisen ihmisen maksimipotentiaalia suoriutua CMM-kyvyistä ja vertailtiin tätä arviota siihen tasoon, jolla niistä yleensä täytyy suoriutua työelämässä. Tämän jälkeen CMM-kykyjä vertailtiin olemassa olevan teknologian sovelluspotentiaaliin automaatiossa. (Chui, Manyika & Miremadi 2017, 4, 34-35).

Kuvio 4. 18 työelämässä arvokkaan kyvyn ihmisvertoinen automatisoitavuus vuoden 2017 teknologialla (Chui, Manyika & Miremadi 2017, 35)

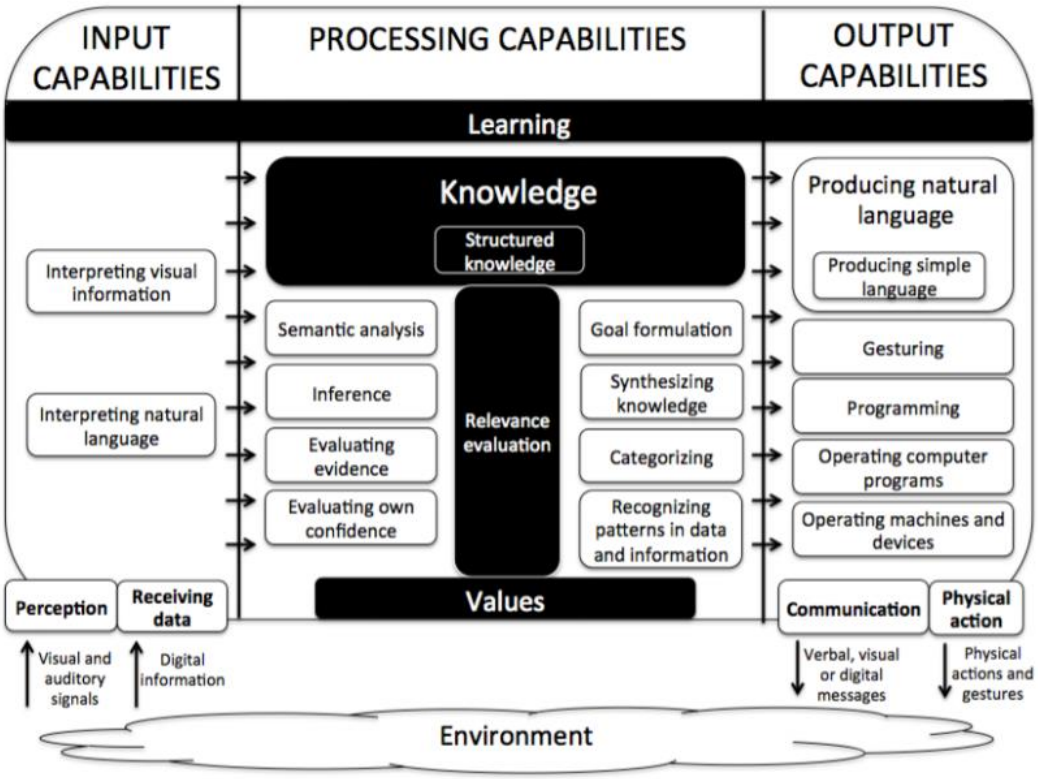
Below median Median Top quartile

	Automation capability	Capability level[1]	Description (ability to …)
Sensory perception	Sensory perception		Autonomously infer and integrate complex external perception using sensors
Cognitive capabilities	Recognizing known patterns/categories (supervised learning)		Recognize simple/complex known patterns and categories other than sensory perception
	Generating novel patterns/ categories		Create and recognize new patterns/categories (e.g., hypothesized categories)
	Logical reasoning/ problem solving		Solve problems in an organized way using contextual information and increasingly complex input variables other than optimization and planning
	Optimization and planning		Optimize and plan for objective outcomes across various constraints
	Creativity		Create diverse and novel ideas, or novel combinations of ideas
	Information retrieval		Search and retrieve information from a large scale of sources (breadth, depth, and degree of integration)
	Coordination with multiple agents		Interact with others, including humans, to coordinate group activity
	Output articulation/ presentation		Deliver outputs/visualizations across a variety of mediums other than natural language
Natural language processing	Natural language generation		Deliver messages in natural language, including nuanced human interaction and some quasi language (e.g., gestures)
	Natural language understanding		Comprehend language, including nuanced human interaction
Social and emotional capabilities	Social and emotional sensing		Identify social and emotional state
	Social and emotional reasoning		Accurately draw conclusions about social and emotional state, and determine appropriate response/action
	Social and emotional output		Produce emotionally appropriate output (e.g., speech, body language)
Physical capabilities	Fine motor skills/dexterity		Manipulate objects with dexterity and sensitivity
	Gross motor skills		Move objects with multidimensional motor skills
	Navigation		Autonomously navigate in various environments
	Mobility		Move within and across various environments and terrain

1 Assumes technical capabilities demonstrated in commercial products, R&D, and academic settings; compared against human performance.

Kuvion 4 erittely muistuttaa sisällöllisesti paljon Lampelton vastaavaa (Kuvio 5).

Lampelto kehitti tutkimuksensa tarpeisiin tehtäväteorian ensin haastattelemalla viittä eri tietotyön työnkuvaa edustavaa ammattilaista, minkä jälkeen näiden työnkuvista tunnistettiin 104 tehtävää, jotka ryhmiteltiin 25 tehtävätyyppiin. Näitä tehtävätyyppejä analysoitiin ja tunnistettiin 24 kykyä (Kuvio 5), jotka tarvittiin niiden suorittamiseen. Nämä jaettiin syötteen vastaanottamisen, prosessoinnin ja tulosten ilmaisun kategorioihin. (Lampelto 2013, 112-134.)



Kuvio 5. Työnkuvahaastatteluista tunnistetut keskeiset kyvyt (Lampelto 2013, 86)

Lampelton tunnistamat kyvyt (vastedes Lampelto-kyvyt) ovat kootusti (Lampelto 2013, 80-85) (suom. Lilja Tamminen):

- Datan vastaanottaminen (*receiving data)*
- Ympäristön havainnointi (*perception)*
- Luonnollisen kielen tulkinta (*interpreting natural language*)
- Visuaalisen informaation tulkinta (*interpreting visual information*)
- Semanttinen analyysi (*semantic analysis*)
- Päättely (*inference*)
- Näytön arviointi (*evaluating evidence*)
- Itseluottamuksen arviointi (*evaluating own confidence*)
- Opettelu (*learning*)
- Strukturoitu tieto (*structured knowledge*)
- Tietämys (*knowledge*)
- Priorisointi (*relevance evaluation*)
- Arvo- ja tunneäly (*values*)
- Tavoitteiden määrittely (*goal formulation*)
- Tiedon synteesi (*synthesizing knowledge*)
- Luokittelu (*categorizing*)
- Kuvioiden tunnistaminen (*recognizing patterns in data and information*)
- Luonnollisen kielen tuottaminen (*producing natural language*)
- Yksinkertaisen kielen tuottaminen (*producing simple langiage*)
- Elehdintä (*gesturing*)
- Ohjelmointi (*programming*)
- Tietokoneiden käyttäminen (*operating computer programs*)
- Koneiden ja laitteiden käyttäminen (*operating machines and devices*)
- Tulosten ilmoittaminen (*communication*)

- Fyysiset toimet (*physical action*)

Lampelto arvioi tutkimuksessaan mitkä näistä kyvyistä olivat sellaisia, joista ihmisten ohjelmoima IBM Watson -asiantuntijasovellus voisi suoriutua. Nämä tunnistetut kyvyt olivat opettelu, datan vastaanottaminen, luonnollisen kielen tulkinta, semanttinen analyysi, päättely, näytön arviointi, itseluottamuksen arviointi, strukturoitu tieto (esim. taulukkodatasta johdettu tietopohja, joka tarjoaa vastauksia annettuihin kysymyksiin), kuvioiden tunnistaminen, yksinkertaisen kielen tuottaminen ja tulosten ilmoittaminen. (Lampelto 2013, 134-136.)

Thórissonin ym. peräänkuuluttaman tehtävien rakenteelliseen luonteeseen ja pilkkomiseen keskittymisen sijaan voimme kirjallisuudesta tunnistaa näiden tehtävien suorittamiseen vaadittavia abstrakteja tietojenkäsittelyllisiä tai kognitiivisia kykyjä, joita voimme vertailla olemassa olevien tekoälysovellusten kykyihin. Ilman näitä kykyjä ihminenkään ei voisi suoriutua tehtävistään. Nämä toisistaan erotellut tietojenkäsittelylliset tai kognitiiviset kyvyt ovat toistaiseksi paras taso, jolla voimme yrittää vastata Thórissonin ym. (2016, 5) vaatimuksiin ”samanlaisten ja erilaisten tehtävien” vertailusta, tehtävien ”purkamisesta osatehtäviin ja näiden atomisiin elementteihin” ja ”tehtäväkokonaisuuksien” luomisesta. Joudumme siis tarkastelemaan asiaa työtehtävien suorittamiseen vaadittavien kykyjen tunnistamisen kautta.

Lyhyellä ja keskipitkällä aikavälillä (0-5 vuotta) yksi tehtävien rakenteen ja vaativuuden analysoinnille vaihtoehtoinen tapa ennakoida työnkuvien osatehtävien tai kykyjen liiketoiminnassa tapahtuvaa digitalisoitumista olisi tukeutua pelkästään esimerkkeihin sellaisista työnkuvien osatehtävistä, jotka on jo onnistuneesti automatisoitu joko tutkimusympäristössä tai tuotteistettuina sovelluksina, koska suuri osa organisaatioista ei ole vielä ottanut läheskään kaikkia näitä sovelluksia käyttöönsä. Automaatioteknologian käyttöönotto ei tapahdu hetkessä. Esimerkiksi terveydenhoito, jolla työvoimavaltaisena ja erittäin suurena sektorina piilee suuri palvelutyön automatisoinnin taloudellinen kannustin, on ollut hyvin hidas ottamaan käyttöön tuottavuutta kasvattavaa teknologiaa (Pearl 25.9.2015). Laajaa teknologian hyödyntämistä ei vaikuttaisi enää rajoittavan niinkään teknologian kehitystahti kuin liiketaloudellisten hyödyntämispäätösten ja hankkeiden aikataulut; näiden voidaan olettaa laahaavan perässä teknologisesta kehityksestä merkittävästikin (Tamminen & Pesälä 2016, 178).

Chui, Manyika ja Miremadi:n lähestymistapa oli juuri tämä, eli he pyrkivät vertailemaan liiketoiminnassa hyödynnettyjä kykyjä jo olemassa olevan teknologian soveltamispotentiaa-

liin. Tutkimalla näiden kykyjen välistä digitalisoitumisen järjestystä saattaisi olla mahdollista tehdä päätelmiä näiden tehtävien vaativuudesta. Hyödyllistä tietoa oppimispohjaisten sovellusten soveltamisalan kasvun ennustamisen pohjaksi voisimme siis saada tutustumalla tutkimushankkeissa jo tuotettuihin sovelluksiin.

Tällaisista esimerkkejä ovat mm. (Tamminen & Pesälä 2016, 147-148):

- Sovellus, joka kuvailee ihmisen tavoin sanoin mitä videokuvassa tapahtuu
- Sovellus, joka tuntemillaan kielillä lukee ääneen tekstiä uskottavalla puheäänellä, aksentilla ja hengitysäänin.
- Sovellus, joka värittää mustavalkovalokuvia
- Sovellus, joka osaa tunnistaa saman ihmisen eri kuvista
- Sovellus, joka osasi kehittää teorian planaria-laakamatojen biologisesta mekanismista, jonka avulla ne kykenevät kasvattamaan uudelleen ruumiinsa osia

ja

- Sovellus, joka tunnistaa röntgenkuvista tauteja (Shin, Roberts ym. 2016)
- Sovellus, joka osaa ratkaista aloittelijatason ohjelmointikilpailujen ohjelmointitehtäviä (Balog ym. 2017)

Näiden sovellusten toiminnallisuuksista voidaan päättelemällä tunnistaa tarpeita suoriutua joistain Lampelto-kyvyistä. Tällaisia voisivat olla esimerkiksi strukturoitu tieto, visuaalisen informaation tulkinta, kuvioiden tunnistaminen, luokittelu, elehdintä, päättely ja yksinkertaisen kielen tuottaminen. Esimerkkien perusteella vaikuttaisi myös siltä, että jo olemassa olevien teknologiademonstraatioiden käyttöönotossa olisi monella liiketoiminnan alalla niiden soveltamiseen perustuvaa automaatiopotentiaalia. On kuitenkin tarpeen ymmärtää sitä, miksi juuri nämä sovellukset ovat jo mahdollisia ja miksi toiset eivät vielä ole, jotta voisimme vertailla näiden toteuttamisen haastavuutta ja sen pohjalta tehdä ennustuksia ja Lampelto-kykyjen automaatioon tarvittavan teknologian kypsyysasteesta.

Algoritmien tutkimuksessa on havaittu jo pitkään, että joidenkin ongelmien ratkaiseminen vaikuttaisi olevan rakenteellisesti vaikeampaa kuin toisten. Tätä laskennallisen kompleksisuusteorian tutkimus oleellisesti tutkii. Havainto toistuu esimerkiksi tietojenkäsittelytieteen ratkaisemattomassa peruskysymyksessä ”P=NP?”, eli jos jonkin ongelman ratkaisun oikeellisuuden tarkistaminen algoritmilla on mahdollista polynomiaalisessa ajassa syötteen määrän funktiona (vrt. eksponentiaalisessa ajassa), voidaanko myös itse ongelma ratkaista polynomiaalisessa ajassa? Tietojenkäsittelytieteilijöiden enemmistön keskuudessa havaintoperusteinen vastaus kysymykseen olisi, että ei ole (Rosenberger 2012). Matemaattista todistusta tälle oletukselle ei ole esitetty, mutta esimerkiksi kryptografia perustuu

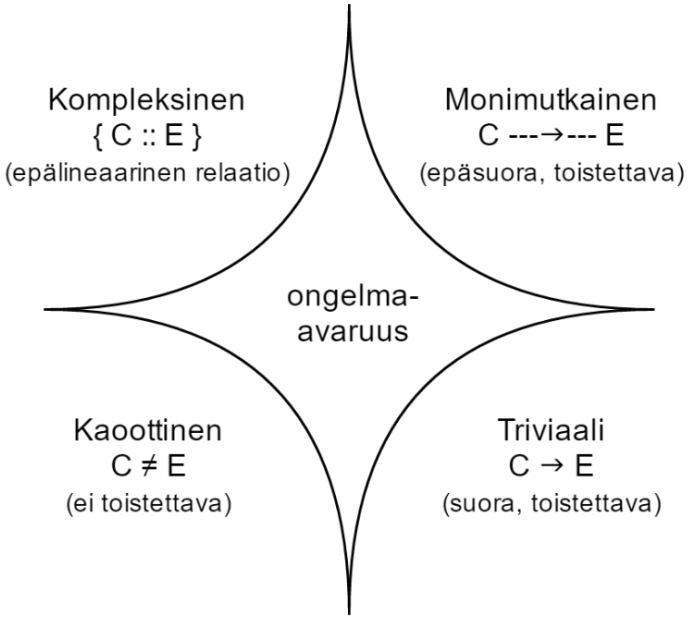
empiiriseen havaintoon, että vastaus on kielteinen. Tämän ongelman valossa laskennallisen kompleksisuuden tarkasteluun on kehitetty erilaisia nimettyjä kompleksisuusluokkia. Näiden yhteensovittaminen Lampelton tai Chui, Manyika ja Miremadin tehtäväteorioiden kanssa sellaisenaan on kuitenkin olemassa olevan tiedon perusteella mahdotonta, koska Thórissonin ym. peräänkuuluttamaa keinoa arvioida Lampelto- tai CMM-kykyjen resurssivaatimuksia ei ole. On siis tarve kehittää alkeellinen approksimoiva keino yhdistää kompleksisuusteoria ja tehtäväteoria.

Steve Wozniak muotoili vitsinä eräässä haastattelussa ihmisen kaltaisen yleistekoälyn todentamiseen ns. kahvitestin, joka samalla valottaa ihmisen toimintaympäristön ongelmaavaruuden luonnetta (Wozniak & Moon 2007) (suom. Lilja Tamminen):

> *”Koneelle annettaisiin tehtävä mennä tyypilliseen amerikkalaiseen kotiin ja sen täytyisi selvittää miten se voi keittää kahvit. Sen täytyisi löytää kahvikone, löytää [ja annostella] kahvipurut, lisätä oikea määrä vettä, löytää kuppi ja valmistaa kahvi painamalla oikeita nappeja.”*

Jos tieteessä tutkimuskysymys ohjaa tutkimuksen löytämiä vastauksia, kenties myös ihmisajattelun ihmisten tunnistamat haastavuuden lajit kertovat jotain niistä inhimillisen ajattelun ongelmista, joita yritämme algoritmisesti ratkaista? Kenties niiden ratkaisu ihmiselle hyödyllisellä tavalla edellyttää samanlaisiin haasteisiin pureutumista? Voisiko olla niin, että Moravecin ja Brooksin peräänkuuluttamaa alhaalta ylös -ajattelua voisi viedä näin pitkälle?

Eräs yksinkertainen tapa hahmottaa ihmisen toimintapiirin ongelmien ratkaisun haasteita on IBM:ssä vuonna 2000 alun perin hallinnon kehittämisen tarpeisiin kehitetty Cynefin-viitekehys (Kuvio 6), jolla hahmotettiin haasteita vastata erilaisiin ongelmiin liiketoimintaympäristössä. Cynefin-viitekehyksessä ihmisten toimintaympäristön ongelmat jaetaan triviaaleihin (engl. *simple),* monimutkaisiin (engl. *complicated*), kompleksisiin (engl. *complex*) ja kaoottisiin (engl. *chaotic*) (Callahan 2009) (suom. Lilja Tamminen).



Kuvio 6. Cynefin-viitekehys ja algoritmisen ratkaisun haasteellisuuden lajit (Callahan 2009, muokannut Graves 2010, suomentanut Lilja Tamminen)

Cynefin-viitekehyksen **triviaaliuden alueella** ongelman ratkaisuympäristö on tunnettu ja se käyttäytyy ennustettavasti. Triviaaliuden alueen ongelmat ovat niitä, joita ohjelmakoodirivit yleensä ratkaisevat. Triviaaleja ongelmia ovat yksinkertaisimmillaan esim. kysymykset kuten ”onko 1 + 5 yhtä kuin 6?”, joita saatetaan toistaa peräkkäin suurempien ongelmien asteittaiseksi ratkaisemiseksi. Triviaalit ongelmat ovat luonteeltaan yksitilaisia, eli ongelmaan sisältyy koko tehtävänanto ja kaikki sen ratkaisemiseen tarvittava informaatio. Tyypilliset ohjelmointipohjaiset sovellukset tekevät pääasiassa suoraviivaisia lasku- ja tietokantaoperaatioita ja esittävät näiden pohjalta tietoa ihmiselle. Triviaaleja ongelmia ratkaisemalla sovellukset ovat usein pyrkineet tukemaan ihmisen työskentelyä monivaiheisten ongelmien kanssa.

Triviaali – monimutkainen -janalla siirrytään ristinolla-pelin tapaisista ongelmista kohti ongelmia, joissa toivotun lopputuloksen saavuttaminen edellyttää kykyä ylläpitää tilannekuvaa, jotta voitaisiin ottaa oikeita askeleita kohti toivottua tavoitetta. Cilliersin (1998, 3) mukaan jos järjestelmä (tai ongelma) koostuu suuresta määrästä osia, mutta se voidaan kuvailla tyhjentävästi pilkkomalla se koostumusosiinsa, se on monimutkainen (engl. *complicated*) järjestelmä.

Monimutkaisuuden alueella ratkaisuympäristön vaihtelut sekä ratkaisut esiin tuleviin tilanteisiin voivat olla tunnettuja, kunhan tilannekuva on selvä. Monimutkaisuuden alue edellyttää ongelmakohtaisia taitoja, eli rajallisen ratkaisuympäristön tuntemusta. Tätä aluetta voisi kutsua taktiikan alueeksi. Monimutkaisten ongelmien ratkaisun esimerkkinä voidaan

pitää tietokonepelien tekoälyvastustajien tai robottien logiikan ohjelmointia. Koska koneen ymmärtämä ohjelmakoodi voi kuvata vain triviaaleja ongelmia ja ratkaisuja, ohjelmoinnillinen monimutkaisuuden ratkaisu tapahtuu siis pilkkomalla monimutkainen ongelma triviaaleiksi ehtologiikan ongelmiksi ohjelmoijan toimesta samalla, kun ohjelmoija katsoo, että ohjelmalla on tilannekuva siitä, missä kohdassa (tilassa) ongelmaa se on.

Triviaali – kompleksi -janalla siirrytään rajallisten tunnettujen muuttujien ympäristöstä kohti äärimmäisen monen muuttujan emergenttien ilmiöiden ympäristöä. Cilliersin (1998, 4-5) mukaan systeemi on kompleksi, jos sen lukuisat osat ovat vuorovaikutuksessa keskenään, toimivat rajallisen tiedon puitteissa ja järjestelmällä on historia tätä emergenttiä toimintaa.

Kompleksisuuden alueella löydettävissä olevat ratkaisut eivät todennäköisesti ole täydellisiä ratkaisuja, ja ongelmiin saattaa olla useitakin ratkaisuja. Kompleksisuuden alueen tosielämän ratkaisuympäristöjä ovat esimerkiksi taistelutantereet, markkinat ja ekosysteemit, jotka ovat hyvin monen muuttujan yhteisvaikutusten tuloksia eli emergenttejä ilmiöitä, mutta joiden muuttujista voidaan kuitenkin saada jotain tietoa. Tämän ongelma-alueen ratkaisu edellyttää strategisia taitoja. Kompleksisten ongelmien ratkaisu on ollut erittäin haastavaa toteuttaa ohjelmoimalla, koska täydellisen ratkaisun implementointi edellyttäisi kaikkien ratkaisuympäristön muuttujien tuntemusta ja käsittelykykyä ohjelmoijalta sekä kykyä redusoida ratkaisualgoritmi konekieliseksi triviaaliksi ratkaisuksi. Samasta syystä ohjelmointipohjainen tekoäly esimerkiksi strategiapeleissä on yleensä pärjännyt huonosti ihmispelaajia vastaan, ellei sille ole annettu strategiapelin idean vastaisesti enemmän tietoa pelin tilasta kuin ihmispelaajalle. Ohjelmointipohjaisissa sovelluksissa kompleksisia ongelmia on siis yritetty ratkaista kuin monimutkaisia ongelmia, eli hyödyntämällä valmiita hyviä käytäntöjä eri tilanteisiin sen sijaan, että ongelmaa olisi yritetty syvällisesti ymmärtää.

Triviaali – kaoottinen -janalla siirrytään tunnettujen ongelmien tunnetuista tehokkaiksi tiedetyistä ratkaisuista kohti tilaa, missä valittavien ratkaisukeinojen yhteydet toivottuihin lopputuloksiin ovat epävarmoja, koska ratkaisuympäristö ei voi muuttua ennakoimattomasti. Kaoottisen ratkaisuympäristön historiasta ei voi havaita syy- ja seuraussuhteita, tai ainakaan näiden analyysi ei tarjoa apuvälineitä ongelman ratkaisuun; lainalaisuuksien esiin nousemisen odottaminen on joko ajanhukkaa tai johtaa katastrofiin (Dettmer 2011, 16).

Vaikka kaoottisuuden ongelmat olisivat vain seurausta suuresta muuttujamäärästä eli kompleksisuudesta, kaoottisuuden erilaisuus liittyy siihen, ettemme voi saada näistä

muuttujista tietoa ainakaan ajoissa, joten kaoottinen ympäristö vaikuttaa epädeterministiseltä, vaikkei se olisikaan sitä. Kaoottisuuden alueen ongelmat edellyttävät siis toimijan sopeutumiskykyä ratkaisuympäristön eriasteiseen vaihteluun niin, että ongelma kyetään silti ratkaisemaan – ja ratkaisemaan ajoissa. Käytännössä tämä voisi tarkoittaa esimerkiksi robotin luovaa liikkumista kohti maalia, kun sen liikkumista yritetään tahallisesti estää, tai tietoja seulovan sovelluksen kykyä käsitellä datan tai informaation esitysmuodon ennakoimatonta vaihtelua esimerkiksi tutkimusartikkeleiden välillä.

Vaikka Cynefin-viitekehys on tarkoitettu kuvaamaan ihmisten ongelmanratkaisun haasteellisuuden lajeja eikä tiettyjen ongelmien laskennallista kompleksisuutta, se tarjoaa kuitenkin viitekehyksen hahmottaa ihmisille oleellisten ihmisten toimintaympäristön kognitiivisten ongelmien ratkaisun haasteita, joita koneellisesti sovelluskehittämisessä jouduttaisiin ratkaisemaan jollain korvaavalla tavalla.

Tässä mielessä, jotta tietotekninen sovellus kykenisi ihmisen tavoin ratkaisemaan liiketoiminnan toimintaympäristön ongelmia, sen tulisi siis kyetä ratkaisemaan monimutkaisia, kompleksisia ja kaoottisia ongelmia, joita ihmisetkin joutuvat ratkaisemaan. Tässä mielessä Cynefin-viitekehys tarjoaa yhden keinon tarjota vastauksia Thórissonin ym. (2016, 5) ”samanlaisten ja erilaisten tehtävien vertailun” vaatimukseen, ainakin ihmisen näkökulmasta ”tehtävän suorittamisen resurssivaatimusten” arvioinnin vaatimukseen sekä vaatimukseen ”tehtävien kompleksisuuden luonnehdinnasta tiedon rajallisuuden, oppimispalautteen saamisen keston sekä tehtävänannon luonteen mukaan”.

Toisen yksinkertaisen tavan hahmottaa ongelmien ratkaisuympäristöä esittävät Russel & Norvig (1995). Russel & Norvig kuvaavat ympäristöjä kuudella tekijällä (suom. Lilja Tamminen): tiedon saatavuus (engl. *accessible, inacessible)*, deterministisyys (engl. *deterministic, nondeterministic*), hetkittäisyys (engl. *episodic, nonepisodic*), muuttumattomuus (engl. *static, dynamic*) ja selkeärajaisuus (engl. *discrete, continuous*), joille annetaan arvo tosi tai epätosi. (Russel & Norvig 1995, 46.)

Tiedon saatavuudella Russel & Norvig tarkoittavat samaa kuin Thórisson ym. (2016), eli jos tekoälytoimijan anturit tai muut tiedonsaantivälineet antavat sille mahdollisuuden saada täydellinen kuvaus ympäristöstä, ympäristön tieto on saatavilla. Cynefin-viitekehyksessä tämä vertautuu kysymykseen siitä, voidaanko ratkaisuympäristö tuntea, eli onko kyse joko triviaaleista tai monimutkaista ongelmista, vai kompleksisista tai kaoottisista.

Deterministisyydellä viitataan siihen, onko ratkaisuympäristön seuraava tila täysin sen edellisen tilan ja tekoälytoimijan siinä tekemien toimenpiteiden seurausta. Jos näin on, ratkaisuympäristö olisi Cynefin-viitekehyksen mukaan silloin monimutkainen, koska triviaaleissa ongelmissa ei ole erillisiä tiloja. Jos ratkaisu ei ole deterministinen, se olisi Cynefin-viitekehyksen mukaan joko kaoottinen tai kompleksinen. Russel & Norvig erottavat kuitenkin aidosti epädeterministisen sellaisesta tilanteesta, jossa ympäristön tieto ei ole täysin saatavilla, jolloin ympäristö ainoastaan vaikuttaa epädeterministiseltä. Cynefin-viitekehyksen kaoottisuuden näkökulmasta asian laidalla ei ole käytännön väliä, jos asiaa tulkitaan Dettmerin (2011, 16) kaoottisuuden määritelmän kautta.

Hetkittäisyydellä viitataan siihen, onko ongelman ratkaiseminen riippuvaista muista järjestelmän tiloista kuin juuri käsillä olevasta. Ongelmat, jotka eivät edellytä minkäänlaista tilannekuvaa tai jatkuvuutta, ovat yksinkertaisempia. Cynefin-viitekehyksessä hetkittäiset eli yhden tilan ongelmat rinnastuvat triviaaleihin ongelmiin. Ei-hetkittäiset olisivat silloin luonteeltaan monimutkaisia, mutta niillä saattaisi olla myös kompleksisisa tai kaoottisia piirteitä.

Muuttumattomuudella tarkoitetaan sitä, voiko ympäristö muuttua sinä aikana, kun tekoälytoimija pyrkii löytämään ratkaisua. Jos ympäristö on muuttumaton, toimijan ei tarvitse välittää ajan kulumisesta. Jos ympäristö on muuttuva, se vertautuu Dettmerin (2011, 16) tulkinnan myötä Cynefin-viitekehyksen kaoottiseen ympäristöön, jossa vaikka täydellinen ratkaisu voitaisiin ehkä löytää ajan saatossa, siihen ei ole aikaa.

Selkeärajaisuudella viitataan siihen, onko ratkaisuympäristö rajallinen, eli onko ratkaisuympäristössä selvästi määritelty määrä havaintoja ja toimenpiteitä, joita toimija voi tehdä. Selkeärajaiseksi esimerkiksi Russel & Norvig mainitsevat shakkipelin, ja epäselkeärajaiseksi taksin ajamisen. Selkeärajaisuus vertautuu tällöin Cynefin-viitekehyksen triviaaleihin ja monimutkaisiin ongelmiin, ja epäselkeärajaisuus kompleksisiin ongelmiin. Cynefin-viitekehyksen kaoottiset ongelmat voivat olla joko selkeärajaisia tai olla olematta, mutta kaoottisuuden haasteeksi muodostuu se, ettei ympäristön aiemmista tiloista voi päätellä sen seuraavia tiloja; tässä mielessä kaoottisuus olisi ainakin epäelkeärajaista eri tilojen välillä.

Russel & Norvig esittävät erittelynsä pohjalta oletuksen, että vaikeimmat näistä tekijöistä käsitellä ohjelmoinnillisesti ovat ratkaisuympäristön tiedon epäsaatavuus, jatkuvuus (epähetkittäisyys), muuttuvuus ja epäselkeärajaisuus (Russel & Norvig 1995, 46). Nämä vertautuvat käytännön kannalta Cynefin-viitekehyksen ulottuvuuksiin monimutkainen, komp-

leksi ja kaoottinen. Cynefin-viitekehys siis kattaa myös Russel & Norvigin tunnistamat ratkaisuympäristön haasteellisuuden lajit jos lajeja voi samanaikaisesti soveltaa samaan ongelmaan. Tässä mielessä Cynefin-viitekehys on yksinkertaisempi.

Yhdistämällä Cynefin-viitekehys esimerkiksi Lampelton tunnistamiin kykyihin, voidaan koota alkeellinen tehtäväteoria, jota tulisi testata.

4.2 Automaation taloudellisen potentiaalin arvioiminen

Automaation taloudellisuutta voi arvioida lukuisin eri menetelmin riippuen siitä, mitä tarkkuustasoa ja kuinka yleistettävää arviointimenetelmään tavoitellaan. Erilaisten liiketoiminta-alakohtaisten automaatioratkaisujen taloudellisuuden arviointiin on tutkimuskirjallisuudessa esitetty lukuisia menetelmiä, mutta niiden ollessa niin erikoistuneita, ei niistä välttämättä ole johdettavissa yleistettävää kaavaa.

Yleistettävää automaation taloudellisen potentiaalin arviointia organisaatioissa on tutkinut Kutay (1989). Kutay (1989, 4) toteaa, että vaikka automaation taloudellisuutta voi arvioida pelkästään vähentämällä tuotannon henkilöstökustannuksista automaatioratkaisun investointikustannusten poistot, tällainen teollisen massatuotannon (skaalatalouden) pääomainvestointien arviointitapa soveltuu huonosti automaatioratkaisujen arviointiin, koska se ei ota huomioon automaatioon liittyviä muita hyötypotentiaaleja.

Ainakin 1980-luvun Yhdysvalloissa yritysten vastahakoisuus ryhtyä automaatiohankkeisiin johtui osittain näiden yritysten kyvyttömyydestä arvioida automaation pitkän aikavälin strategisia hyötyjä. Organisaatioiden johdossa ei välttämättä ymmärretty uuden teknologian käyttöönoton hyötyjä, koska niitä on haastavaa esittää taloudellisina tunnuslukuina. (Kutay 1989, 6.)

Strategisia hyötyjä voi kuitenkin arvioida esimerkiksi tunnistamalla organisaation toiminnasta kulueriä, joihin automaation hyödyt kuten nopeampi läpimenoaika epäsuorasti vaikuttaisivat. Kutay antaa esimerkkeinä näistä varastointikulut ja materiaalivaraston arvonmuutokset. Yksi Kutayn tunnistama virhe tavanomaisissa menetelmissä arvioida automaation taloudellista potentiaalia organisaatioissa on vertailla automaatiohankkeen kannattavuusarviota status quo:hon, jossa on oletettu markkinaosuuden, myyntihinnan ja kulut muuttumattomiksi. Kuitenkin uuden teknologian tullessa kaupallisesti saataville, se muuttaa organisaatioiden välistä kilpailutilannetta, vaikka tietty organisaatio ei olisikaan tehnyt tuotantoonsa mitään muutoksia. Markkinaosuusvaikutus on siis myös automaation strateginen hyöty. (Kutay 1989, 11.)

Muun muassa tietohallinnon tarkoituksiin on kirjallisuudessa tunnistettu lukuisia strategisia hyötyjä ja riskejä tietynlaisten digitalisointisovellusten kehitys- ja käyttöönottohankkeista, erityisesti toiminnanohjausjärjestelmähankkeista, mutta niistä on vaikea tehdä yleistyksiä.

Riippumatta siitä, miten hyvin organisaatiot osaavat arvioida liiketoimintansa prosessien automaation taloudellista potentiaalia, liiketoiminnallisten automaatiopäätösten täytyy viime kädessä tukeutua olemassa olevaan teknologiaan, josta on kehitetty joko organisaation itsensä tai jonkun erikoistuneen toimijan toimesta sovelluksia.

Sovellusten kehittämisen taloudellisuus taas poikkeaa hankintapäätösten taloudellisuudesta. Tavoitettavissa olevaan tai jo kehitettyyn teknologiaan perustuvan sovelluskehittämisen keskeisenä taloudellisena kannustimena voidaan pitää kehitettävän sovelluksen myynnin markkinapotentiaalia. Jos esimerkiksi ihmistyöstä riippuvainen prosessi on harvinainen, digitalisoivan sovelluksen markkinapotentiaali on myös alhainen, ellei prosessin ihmistyönä suorittaminen ole äärimmäisen kallista. Vastaavasti, vaikka prosessin suorittaminen ihmistyönä ei olisikaan erityisen kallista tietylle organisaatiolle, sillä voi olla hyvinkin suuri markkinapotentiaali, jos prosessi on taloudessa erittäin yleinen. Oleellista markkinapotentiaalia arvioitaessa on siis kokonaistaloudellinen tehtävän suorittamisen arvo.

Arvon arvioimisen kannalta tärkeää erottaa toisistaan ihmistyöstä maksetut palkat ja työstä syntyvä arvonlisäys organisaatiolle (työnantajalle). Tuotanto ei ole kannattavaa, jos sen kulut ovat suuremmat kuin sen tulot. Niin sanotusti työvoimavaltaisessa tuotannossa kulut muodostuvat pääasiassa henkilöstökuluista ja henkilöstöön liittyvistä muista kuluista eikä tuotantolaitteisiin liittyvistä kuluista. Jotta tuotanto pysyisi kannattavana, tuotteiden markkinahinnan on katettava kaikki tuotannon realisoituvat kustannukset. Kilpailluilla markkinoilla, joissa tuotteet ovat täysin tai lähestulkoon identtisiä, markkinahinta tapaa asettua tuotannon rajakustannusten tasolle. Tästä seuraa, että työvoimavaltaisen ja kilpaillun alan suorittavassa työssä, jossa työvoiman tarjonta on usein suurta, kannattavien yritysten työntekijöiden kokonaispalkat tapaavat seurata työn tuottavuutta.

Vastaavasti sellaisilla markkinoilla, joissa tuotteita on vaikea vertailla keskenään, tuotteet ovat erikoistuneita ja työntekijät usein erikoistuneita asiantuntijoita. Tällöin tuotteiden markkinahinta ei tapaa seurata tuotannon rajakustannuksia vaan määräytyy niiden tuottaman arvioidun hyödyn ja asiakkaiden ostovoiman mukaan, joka saattaa huomattavasti ylittää tuotannon rajakustannukset. Näistä rajakustannuksista merkittävä osa saattaa olla erikoistuneiden asiantuntijoiden kokonaispalkkoja, joiden taso määräytyy kysynnän ja tarjon-

nan mukaan työmarkkinoilla. Näin ollen matalapalkka-aloilla työn tuottavuus (työnantajalle) on yleensä matalampaa kuin korkeasti palkatuilla asiantuntija-aloilla. Tällainen liiketaloudellinen työn tuottavuustarkastelu ei huomioi tuotannon ulkoisvaikutuksia, mutta riittää organisaatioiden automaation kannattavuuden arvioinnin tarkoitukseen.

Tästä seuraa, että jos jonkin asiantuntijatehtävän tuottavuus organisaatiolle on erittäin suuri ja sen tuotteiden kysyntä kasvaa voimakkaasti hinnan laskiessa (eli tuotteen kysyntä joustaa), jos sovelluksella voidaan tehdä tuotannosta edullista ja skaalautuvaa, markkinapotentiaali on myös suuri.

Lisäksi, kuten Kutay (1989, 11) toteaa, automaation taloudelliseen kannattavuuteen vaikuttavat myös strategiset hyödyt kuten epäsuorat kustannussäästöt ja markkinaosuusvaikutus. Jos nämä voidaan sovelluksen markkinoinnissa kvantifioida asiakkaille uskottavasti, markkinapotentiaaliin vaikuttavat myös strategiset hyödyt.

Esimerkkinä tästä toimii yleislääkärin vastaanottoprosessin automatisointi; vaikka yleislääkärin vastaanoton kokonaan korvaavan sovelluksen kehittäminen maksaisi miljardeja euroja, se olisi silti kannattavaa, koska sovellus voitaisiin myydä tuhansiin toimipaikkoihin ja sitä voisi käyttää vuorokauden ympäri lukemattomien ihmisten samanaikaiseen palveluun lähes ilmaiseksi. Tämä mahdollistaisi myös yleislääkärin palveluiden tarjoamisen niille, joilla ei siihen tänä päivänä ole varaa. Alhaiset rajakustannukset mahdollistavat myös sen, että potilaat voisivat käyttää huomattavasti enemmän aikaa oireidensa ja vaivojensa kuvailuun, mikä voisi tarkentaa diagnooseja. Jos järjestelmä osaisi oppia potilaistaan, se tulisi myös jatkuvasti paremmaksi. (Tamminen & Pesälä 2016, 149-150.)

Sovellustuotteen skaalautuvuus markkinapotentiaalin tyydyttämiseen on siis keskeinen digitalisoivien sovellusten kehittämispäätösten ja kehitystyön ohjaustekijä. Se antaa samalla huomattavan liiketoiminnallisen kilpailuedun skaalautuville digitalisoiville sovelluksille erikoistuneisiin tarpeisiin tuotettuihin mekaanisiin automaatiolaitteisiin verrattuna, mikä kannustaa muuttamaan tuotantoa aineettomammaksi tai ainakin muuttamaan sitä yleispätevämpiin laitteisiin tukeutuvaksi.

Todennäköisesti kehitettävän digitalisoivan sovelluksen piirteitä ovat siis, että:

- sillä on korkeaksi arvioitu markkinapotentiaali
- se on teknisesti toteutettavissa olemassa olevalla teknologialla
- se skaalautuu kysynnän mukaan, jotta se tavoittaisi markkinapotentiaalin

Tällaisen yleistettävän sovelluksen kehittämisen kustannukset voivat olla hyvin suuret,

mikä edellyttää kehittämistä yrittäviltä tahoilta huomattavaa riskinottokykyä. Jos taho kuitenkin onnistuu kehittämisessään, kustannukset ovat niin vähäiset suhteessa sovelluksen hyötyihin, että markkinapotentiaalin koko on merkittävin sovelluksen kehittämistä ennustava tekijä. Kustannusriskin kantaminen saattaa kuitenkin edellyttää suurta organisaatiota, kuten suuryritystä tai valtiota.
Kun arvokas sovellus on kehitetty, sen markkinapotentiaalia vastaavaa käyttöönottoa tuotannossa rajoittaa enää organisaatioiden kyky ja resurssit ryhtyä automaatiohankkeisiin. Mikäli sovelluksella automaation kannattavuus on huomattavaa, kilpailluilla markkinoilla ensimmäiset uusien sovellusten käyttöönottajat syrjäyttävät keskipitkällä aikavälillä markkinoilta ne, jotka eivät ryhtyneet automaatioon. Kilpailluilla markkinoilla digitalisaatio siis etenee nopeasti sen jälkeen, kun arvokas sovellus on kehitetty.

Yleistettävän sovelluksen kehittämistä voidaan siis sanoa ohjaavan lähinnä markkinapotentiaalin koko, sovelluksen edellyttämä teknologinen kehitysaste ja tutkimusresurssien kohdentamisvalinnat.

5 Malli tehtäväkomponenttien digitalisaation ennustamiselle

5.1 Ongelman redusointi malliksi

Hyödynnettävän mallin aikaansaamiseksi tehdään kolme keskeistä oletusta.

Ensinnäkin digitalisoivan sovelluksen kehittämisen kannustin riippuu teknologisen haastavuuden lisäksi siitä, mikä sovelluksen taloudellinen markkinapotentiaali on.

Toiseksi, koska digitalisoinnin taloudellinen potentiaali ohjaa pyrkimystä löytää ratkaisuja erilaisiin ongelmiin, toimiva digitalisaation etenemisdynamiikan malli huomioi taloudellisen potentiaalin vaikutuksen tutkimustyön resursointiin ja fokukseen. Äärimmäinenkään taloudellinen potentiaali ei kuitenkaan voi yksin kumota erittäin haastavan ongelman ratkaisemisen haastavuutta mm. tutkimuskapasiteetin rajallisuuden vuoksi. Automaation taloudellinen potentiaali vaikuttaa siis lähinnä siihen, mitä vaihtoehtoisia tutkimuksen etenemispolkuja lähdetään resursoimaan tutkijoilla ja tutkimusfasiliteeteilla. Taloudellinen potentiaali ei siis vaikuta itse teknologiseen haasteeseen, vaan sen ratkaisemiseen liittyviin valintoihin organisaatioissa. On kuitenkin hyvin vaikeaa kvantifioida sitä, kuinka paljon nämä valinnat vaikuttavat teknologian kehitystahtiin.

Kolmanneksi oletetaan, että työtehtäviin vaadittujen kykyjen tai taitojen laskennallisen kompleksisuuden kvantifiointiin on löydettävissä keinoja, joiden tarvetta ja luonnetta Thórisson ym. (2016, 5) kuvaili.

Näiden oletusten pohjalta oletukset redusoidaan kahdeksi abstraktiksi mutta kvantifioitaviksi tarkoitetuiksi muuttujiksi. Yksi on tietyn tehtäväkomponentin tai kyvyn automaation teknologinen haastavuus (kustannus) *C* arvioinnin laadintahetkenä. Toinen on tehtäväkomponentin tai kyvyn automaation arvioitu taloudellinen potentiaali (hyöty) *B.* Ajallista etäisyyttä tehtäväkomponenttien automaation välillä voidaan ennustaa vertailuluvulla tai *digitalisaatiopotentiaalilla D*, joka saadaan kaavalla D = C – B.

D käyttäytyy tehtäväkomponenttien tai kykyjen vertailussa järjestysluvun tavoin ja sen avulla voidaan vertailemalla ennustaa eri tehtäväkomponenttien keskinäistä digitalisaatiojärjestystä. Mallin muuttujien laskentaan käytetty kvantifiointimenetelmä määrittelee ennustuksen tarkkuuden.

5.2 Mallin arviointimenetelmä

Digitalisaatiopotentiaalin havainnollistamiseksi, Thórissonin ym. (2016, 5) laskennallisen tehtäväteorian puitteiden approksimointiin valitaan automatisoitaviksi kyvyiksi Lampelto-kyvyt niiden määritelmien mukaan tulkittuina (Lampelto 2013, 80-85). Koska tällaisen laskennallisen tehtäväteorian laskennallisen vaativuuden arviointiin ei toistaiseksi ole kirjallisuudessa keinoja, muuttuja C muodostetaan NASA-TLX:n toimintaperiaatetta mukaillen siten, että opinnäytetyön tekijä etsii kullekin kyvylle sen soveltamiseen liittyvän palkkatilastoista löytyvän työtehtävänimikkeen ja arvioi työtehtävässä kyvyn *menestyksekkään* soveltamisen haasteita oleellisilla Cynefin-ongelma-alueilla (monimutkaisuus, kompleksisuus ja kaoottisuus) skaalalla 0-10 ja ynnäämällä pisteet yhteen. Kaoottisuuden painokertoimeksi asetetaan luku 2, eli pistearvo kerrotaan kahdella. Tällöin oletetaan käytännön vuoksi, että Cynefin-alueiden ongelmanratkaisun vaativuus ei riipu muista Cynefin-alueista. Lisäksi oletetaan käytännön vuoksi, että ongelmien ratkaisun ihmisen kokema haastavuus on suorassa suhteessa sen laskennalliseen kompleksisuuteen.

Monimutkaisuus C_{co} on tällöin arvioitu vaikeusaste ihmisen ratkaista kyvyn mukaisia ongelmia ohjelmointipohjaisella sovelluskehityksellä C_{co}. Mikäli kyvyn toteuttaminen edellyttää lähinnä triviaalien ongelmien ratkaisua, arvo on arvioidun haastavuuden perusteella välillä 0-3. Jos ongelmat ovat luonteeltaan haastavampia mutta asiantuntijoiden ratkaistavissa ohjelmoinnillisesti, arvo on 4-9. Jos taas ratkaisu edellyttää äärimmäistä taitavuutta tai ylittää ihmisen kyvyn, arvo on 10.

Kompleksisuus C_{cx} arvioidaan ihmisen kokemana kuormittavuutena tai vaikeutena ratkaista kompleksisuuteen liittyviä ongelmia. Mikäli kyky ei edellytä lainkaan kompleksisuuden ratkaisukykyä, arvo on 0 (nolla). Mikäli kyky on ihmisen ehdottoman maksimikapasiteetin kohdalla tai sen yli, arvoksi asetetaan 10. Tältä väliltä (1-9) arviointi jää hyvin subjektiivisesti.

Kaoottisuus C_{ch} arvioidaan kompleksisuuden tavoin eli ihmisen kokemana haastavuutena ratkaista kaoottisuuteen liittyviä ongelmia, mutta pistearvo kerrotaan kahdella, jotta voitaisiin kompensoida huomattavaa eroa ihmisen ja olemassa olevien sovellusten kyvyissä ratkaista kaoottisia ongelmia. Tehdään siis todennäköisesti voimakkaasti aliarvioiva oletus, että kaoottisuus on 100% haastavampi ongelmanratkaisun laji. Näin ollen vaihteluväli on 0-20 parillisin luvuin.

Automaation taloudellinen potentiaali arvioidaan seuraavalla menetelmällä:

1. Kyseessä olevan Lampelto-kyvyn soveltamiseen pääosin tai stereotyyppisesti perustuva työtehtävänimike valitaan opinnäytetyön tekijän subjektiivisen arvion perusteella Oikotie -verkkopalvelun keskipalkkatietokannasta, josta tarkistetaan tehtävänimikkeen kuukausikeskipalkka B_{kp}, joka kirjataan ylös euroissa vuoden 2017 alun hintatasossa (Oikotie 2017).
2. Palkkatiedon pohjalta laaditaan löyhä arvio työnantajalle tuotetusta arvonlisäyksestä kertomalla kuukausipalkka sen tuhannesosan neliöjuurella (kertoimen vaihteluväli 1,05 -2,5) perustuen oletukseen, että matalapalkka-aloilla arvonlisäys on lähellä työvoimakustannusta, kun taas korkeasti tuottavista tehtävistä maksetaan usein korkeampia palkkoja, mutta korkeampien palkkojen ero työnantajalle tuotettuun arvonlisäykseen myös on suurempi.
3. Kyseessä olevan kyvyn työmarkkinatarpeen yleisyysaste *Y* arvioidaan opinnäytetyön tekijän subjektiivisen arvion mukaan skaalalla 1-5, jossa arvo 1 viittaa kyvyn erinomaisen suorittamisen harvinaisuuteen ja 5 sen täysin itsestäänselväksi olettamiseen työnhakijoiden keskuudessa.
4. Koska kykyjen soveltamisen yleisyysaste todennäköisesti vaihtelee huomattavasti enemmän kuin lineaarisella skaalalla 1-5, *Y* korotetaan mallin testaamisessa potenssiin 1,5. Tällöin markkinapotentiaaliarvo arvioidaan kertomalla arvonlisäysarvio luvulla $Y^{1,5}$.
5. Lopuksi, koska teknologista haastavuutta ei voi suoraan vertailla rahamääräiseen markkinapotentiaaliin, tehdään käytännön vuoksi summittainen oletus, että suurikaan markkinapotentiaali ei kompensoi teknologisen haastavuuden pisteitä kuin maksimissan 10 pisteen verran. Tämä kompensoiva pistevaikutus arvioidaan siten, että kun kaikista kyvyistä on laadittu markkinapotentiaaliarvio, näistä muodostetaan vertailuluvut välillä 0-10 jakamalla kunkin kyvyn markkinapotentiaali kaikista kyvyistä suurimmalla markkinapotentiaalilla ja kertomalla tulos 10:llä. Tällöin suurimman markkinapotentiaalin omaava automatisoitava kyky saa vertailuluvun 10 ja muut vertailuluvut ovat tätä pienempiä.

Käytetyt laskentakaavat C:lle ja B:lle arviointimenetelmässä ovat siis:

$$C = C_{co} + C_{cx} + 2 * C_{ch}$$

$$B = \frac{B_{kp} * \sqrt{B_{kp}/1000} * Y^{1,5}}{\max_{(B_{kp} * \sqrt{B_{kp}/1000} * Y^{1,5}),\ldots}} * 10$$

Kun näin tehdään kaikille Lampelto-kyvyille, saadaan kykyjen keskinäiseen vertailuun taulukko, joka voidaan järjestää digitalisaatiopotentiaalin mukaan. Arviointimenetelmässä suurimmaksi arvoksi C:lle muodostuu 40 ja B:lle 10. Taulukossa 2 on havainnollistettu menetelmällä syntyvää syötedataa tietyn Lampelto-kyvyn kohdalla.

Taulukko 2. Havainnollistus tarkastelutaulun syötedatasta (kts. liite 1)

Kyky	Tehtävä	Keski-palkka	Yleisyys Y	C_{co} (0-10)	C_{cx} (0-10)	C_{ch} (0-10)
Semanttinen analyysi	Hallintoassistentti	2839	3	10	4	5

6 Tulokset ja pohdinta

6.1 Lampelto-kykyjen digitalisaatiopotentiaali

Taulukko 3. Lampelto-kyvyt, niiden verrokkityötehtävät ja digitalisaatiopotentiaalia kuvaavat digitalisaatioindeksit (DI) arviointimenetelmän mukaan (kts. liite 1)

Komponenttinen kyky	DI	C	B	Työnimike	Keskipalkka
Datan vastaanottaminen	−0,8	2	2,78	Tallentaja	1 805 €
Tulosten ilmoittaminen	1,1	3	1,85	Toimistoharjoittelija	1 102 €
Yksinkertaisen kielen tuottaminen	2,5	6	3,47	Kassamyyjä	1 675 €
Strukturoitu tieto	4,7	9	4,26	Lakiasiantuntija	3 197 €
Visuaalisen informaation tulkinta	9,0	19	10,00	Valvomopäivystäjä	3 390 €
Elehdintä	9,5	12	2,47	Asiakaspalvelija	2 224 €
Luonnollisen kielen tuottaminen	10,1	17	6,90	Tekninen kirjoittaja	3 310 €
Kuvioiden tunnistaminen datasta	12,1	22	9,90	Ketjumyyntipäällikkö	5 611 €
Päättely	12,3	15	2,69	Myynti- ja markkinointiassistentti	2 352 €
Luokittelu	14,5	19	4,53	Lajittelija*	2 000 €
Ohjelmointi	14,6	17	2,39	Ohjelmoija	3 266 €
Itseluottamuksen arviointi	16,6	21	4,43	Tutkija (tohtori)	4 923 €
Priorisointi	18,6	24	5,39	Projektijohtaja	5 615 €
Luonnollisen kielen tulkinta	19,6	23	3,42	Tekstinkäsittelijä (sanelunpurkaja)	2 071 €
Motoriset taidot	20,4	25	4,60	Apupoika	3 369 €
Semanttinen analyysi	20,4	24	3,56	Hallintoassistentti	2 839 €
Tietokoneiden käyttäminen	20,8	24	3,22	Kirjanpitäjä	2 654 €
Ympäristön havainnointi	21,3	26	4,67	Laadunvalvoja	2 549 €
Opettelu	21,8	27	5,16	Tohtorikoulutettava	2 727 €
Näytön arviointi	23,0	28	5,04	Käräjätuomari	5 366 €
Koneiden ja laitteiden käyttäminen	23,4	28	4,56	Asentaja / tekninen tuki	3 345 €
Arvo- ja tunneäly	25,2	29	3,82	HR-päällikkö	4 458 €
Tavoitteiden määrittely	26,3	34	7,75	Markkinointijohtaja	7 148 €
Tiedon synteesi	26,5	28	1,46	Vanhempi tutkija	4 698 €
Tietämys	29,9	32	2,12	Johdon konsultti	6 025 €

Kuten taulukosta 3 käy ilmi, ensimmäiset neljä tehtävää (alimmat indeksit) ovat luonteeltaan sellaisia, että ne voitaisiin digitalisoida ohjelmointipohjaisella sovelluskehittämisellä ainakin osittain, koska niihin onkin jo tarjottu ohjelmointipohjaisia sovelluksia. Näistä ensimmäiselle (datan vastaanottaminen) on muodostunut negatiivinen indeksi, mikä sopisi siihen tilannekuvaan, että se on jo automatisoitumassa ja paljolti automatisoitukin. Ensimmäisten neljän jälkeen alkaa ilmetä paremmin oppimispohjaisen sovelluskehittämisen ratkaistavaksi soveltuvia ongelmia, joissa kompleksisuus ja kaoottisuus alkavat muodostaa haasteita ohjelmoinnille. Taulukon 3 esittämät arviot ovat lähinnä suuntaa-antavia, koska monet vertailuluvut ovat hyvin lähellä toisiaan.

Taulukosta 3 käy myös ilmi, että kykyjen soveltamisesta työssä maksettava palkka ei aina vaikuttaisi vastaavan algoritmisen ratkaisun haastavuutta tai digitalisaatiopotentiaalia, vaikka haastavuus vaikuttaisikin yleisesti korreloivan palkkatason kanssa. Hyvä esimerkki tästä epäsuhdasta on ympäristön havainnointi (DI: 21,3), joka on ihmiselle käytännössä universaali kyky, johon esimerkiksi erilaiset valvonta- ja vartiointityötehtävät perustuvat. Strukturoidun tiedon kyvystä (DI: 4,7) vuorostaan maksetaan sen haastavuuteen nähden paljon palkkaa.

Tätä epäsuhtaa voi yrittää selittää sillä, että ympäristön havainnointi on hyvin haastava automatisoida käytännön työympäristöissä, koska työ tukeutuu ääni- ja näköhavainnoinnin lisäksi myös kulttuuriseen ja yhteiskunnalliseen ymmärrykseen ympäristöstä ja muiden ihmisten toimintatavoista, mikä auttaa tehtävää suorittavaa ihmistä hallitsemaan ympäristön kaoottisuutta redusoimalla osa kaoottisuuden ongelmista kompleksisiksi tai monimutkaisiksi ongelmiksi, joihin voi yrittää soveltaa muistisääntöjä tai ns. parhaita käytäntöjä. Havainnointikyvyn alhainen työmarkkina-arvostus ei siis selity tehtävän itsensä algoritmisella yksinkertaisuudella, vaan sen suurella työmarkkinatarjonnalla. Arkista kaoottisuuden hallintakykyä ei siis arvosteta, koska se on ihmisolennoille evoluution myötä tullut helpoksi ja yleiseksi. Vastaavasti strukturoidun tiedon kykyä arvostetaan, koska vaikka se ei ole teknisesti erityisen haastavaa, se vaatii ihmisiltä suuren informaatiomäärän omaksumista. Tässä mielessä kysynnän ja tarjonnan lait vaikuttaisivat pätevän työmarkkinoilla.

On oikeastaan helpompi kuvitella, että ympäristön havainnointiin liittyvät työnkuvat koneellistetaan ensin edullisempien automaatioratkaisujen avulla, jotka ovat ihmiseen verrattuna tehtävässä jokseenkin kömpelöitä mutta tarpeeksi edullisia, että määrä voi korvata laadun. Yksinkertaisimpana esimerkkinä tällaisesta ratkaisusta vartiointikäytössä voi pitää nykyään edullisia infrapunatekniikkaan perustuvia liiketunnistimia. Strukturoidun tiedon kyvyn digitalisointi on vuorostaan sen verran helppoa, että se todennäköisesti tapahtuu laajasti hyvin pian.

Taulukosta 3 näkee täysin odotetusti myös, että monet korkeaa osaamista vaativat tietotyön kyvyt kuten tietämys (DI: 29,9), tiedon synteesi (DI: 26,5), tavoitteiden määrittely (DI: 26,3) ja arvo- ja tunneäly (DI: 25,2) eivät olisi ensimmäisinä digitalisoitumassa. Toisaalta osa kyvyistä ei välttämättä ole niin turvassa digitalisaatiolta kuin yleisesti ehkä oletetaan, kuten vaikuttaisi olevan ohjelmoijan ohjelmointikyvyn (DI: 14,6) tai lakiasiantuntijan strukturoidun tiedon (DI: 4,7) kohdalla.

6.2 Tulosten pohjalta nähtävissä oleva digitalisaation etenemiskaava

Digitalisaatiopotentiaalia kuvaavan mallin tarkoitus on tarjota väline vertailla keskenään eri palvelu- ja asiantuntijatyönkuvien osien digitalisoinnin järjestystä. Mallin tarkkuus riippuu käytetystä arviointimenetelmästä. Käytetyn arviointimenetelmän tulosten (Taulukko 3) ja toisaalta jo olemassa olevan teknologian tutkimuskirjallisuuden perusteella voidaan esittää joitain yleistason havaintoja.

Kypsät hedelmät (low-hanging fruit)

Ohjelmointipohjaisella sovelluskehityksellä digitalisoitavia triviaaleja ja monimutkaisia työnkuvien osa-alueita löytyy luultavasti kaikista organisaatioista. Esimerkiksi merkittävä osa monen toimistotyöläisen työajasta kuluu edelleen varsin rutiininomaiseen ja analyysittömään tietojen syöttämiseen paperilta tietokoneelle tai yhteen sopimattomasta tietokantasovelluksesta toiseen (Tamminen & Pesälä 2016, 199-200). Tämänlaiset tehtävät vertautuvat Lampelto-kykyihin kuten datan vastaanottaminen (DI: -0,8), tulosten ilmoittaminen (DI: 1,1) ja strukturoitu tieto (DI: 4,7), jotka ovat jo digitalisoitumassa.

Kehittämällä esimerkiksi toiminnanohjausratkaisuja, järjestelmien integraatiota ja muita sovelluksia voidaan edelleen saavuttaa digitalisoinnin hyötyjä datan vastaanottamisen, strukturoidun tiedon, tulosten ilmoittamisen sekä yksinkertaisen kielen tuottamisen (DI: 2,5) alueilla. Monen tällaisen sovelluksen toteutettavuutta rajoittaa lähinnä arvokkaan datan saatavuus tai sen kaoottinen ilmenemismuoto.

Datan ilmenemismuodon rajoitteeseen voidaan yksinkertaisimmillaan vastata yhdenmukaistamalla uuden syntyvän datan ilmenemismuotoa esimerkiksi standardoinnilla ja kehittämällä tietovarantojen välille avoimia rajapintoja. Esimerkiksi koneluettavat digitaaliset kuitit ovat yleistymään päin Suomenkin kuluttajamarkkinoilla, vaikka niitä ei yleensä voikaan suoraan välittää kirjanpitosovelluksille. Myös terveydenhuollon tietojärjestelmien kehittämisessä tehdään edelleen uusia aluevaltauksia ohjelmointipohjaisella kehittämisellä esimerkiksi asiantuntijajärjestelmien parissa, joihin liittyy keskeisesti strukturoidun tiedon Lampelto-kyky.

Toisaalta ohjelmointipohjaisilla sovelluksilla digitalisointia saattaa jo rajoittaa prosessikehittämisen laskeva rajahyöty. Tällöin suoraviivaisesti digitalisoitavien kykyjen vielä laajemmasta automaatiosta saatavat lisähyödyt arvioidaan jo pienemmiksi kuin sovelluskehittämisen kustannukset. Tässä mielessä esimerkiksi Helsingin seudun kuntien asiakas- ja po-

tilastietojärjestelmä Apotin käyttöönottohanke toimii eräänlaisena benchmarkina ohjelmointipohjaisen sovelluskehittämisen vielä hyödyntämättömän tuottavuuskehityspotentiaalin määrälle terveydenhuollossa, josta saataneen joitain arvioita muutaman vuoden kuluessa. Useita vastaavia suuria terveydenhuollon toiminnanohjaushankkeita on käynnissä myös ulkomailla.

Pian kypsät hedelmät

Monet tutkimuksissa jo demonstroidut oppimispohjaiset sovellukset yritysten ja julkisten tahojen tarpeisiin saattavat astua markkinoille tuotteistettuina sovelluksina lähivuosina. Niiden käyttöönotto riippunee pian enää organisaatioiden kyvystä ja resursseista ottaa käyttöön uusia sovelluksia. Tällaisia tuotteiksi kypsyviä sovelluksia ovat erityisesti kompleksisuuteen ja hallittavan kaoottisuuden ongelmien ratkaisuun liittyvät kyvyt kuten visuaalisen informaation tulkintaan (DI: 9,0) ja kuvioiden datasta tunnistamiseen (DI: 12,1) liittyvät ratkaisut. Näiden käytännön sovelluksia voisivat olla laboratoriokoetulosten ja röntgenkuvien seulonta sekä valokuvien kategorisointi niiden sisältämien esineiden ja muotojen perusteella. Osittain tai kokonaan ohjelmoitujenkin asiantuntijasovellusten parissa tapahtuu nopeaa kehitystä. Semanttisen analyysin (DI: 20,4) esiasteista on jo alkeellisia demonstraatioita sekä priorisoinnista (DI: 18,6). Näistä esimerkkinä toimii vuonna 2016 julkistettu ROSS Intelligence -tutkimussovellus, joka tukee lakiasiantuntijoita tutkimustyössään (Lohr 2017). Myös uudet insinöörityön CAD-suunnittelua tukevat optimointisovellukset mahdollistavat merkittäviä parannuksia materiaalitehokkuuteen ja kestävyyteen (Galjaard ym. 2015, 13-15). Näillä alueilla kuviteltavissa oleva tuottavuuslisäyspotentiaali on suuri.

Istutettu sato

Uudet nähtävissä olevat ja aktiivisen tutkimustyön ja alustavan tuotekehitystyön kohteena olevat teknologiat ovat pääosin oppimispohjaiseen sovelluskehitykseen tukeutuvat asiantuntijajärjestelmät. Nämä ovat erityisesti monimutkaisuusongelmien ratkaisemiseen liittyvät sovellukset, kuten ohjelmistosuunnittelijan työtä tukevat ohjelmoijasovellukset. Uusia mahdollisuuksia avautuu myös strukturoidun tiedon sovelluksille. Luonnollisen kielen tuottamisen (DI: 10,1) ja elehdinnän (DI: 6,7) kykyjen digitalisaatio mahdollistaa esimerkiksi koneellisen puheviestinnän ihmisille. Sen sijaan luonnollisen kielen tulkinnan (DI: 19,6) digitalisoinnin haastavuus rajoittaa asiakaspalvelutyön digitalisointia, mahdollistaen lähinnä alkeellisen tekstiin tai puheeseen pohjaavan reagoinnin; kielen semanttinen ymmärtäminen on kokonaan eri haastavuusluokkaa.

Näköpiiri

Selvästi näköpiirissä ovat sovellukset, jotka ratkaisevat erityisesti kompleksisuuteen eli muuttujien valtavaan määrään mutta sinänsä tunnettuun ratkaisuympäristöön liittyviä ongelmia, jotka suppeudestaan huolimatta ylittävät ihmisen käsityskyvyn. Tässä keskeisiä kykyjä ovat kuvioiden tunnistaminen datasta (DI: 12,1), päättely (DI: 12,3), itseluottamuksen arviointi (DI: 16,6), priorisointi (DI: 18,6) ja semanttinen analyysi (DI: 20,4). Kompleksisten ongelmien perusteellisessa ymmärtämisessä koneellisella laskennalla on huomattava etulyöntiasema ihmiseen verrattuna, mutta tosielämän kompleksisten ongelmien ratkaisemisen haasteena on ollut syöteinformaation kaoottisuus ja ongelmien ratkaisun muotoilu ohjelmoinnillisesti. Sovellusten kehittämisen merkittävä haaste liittyy ihmisten aiemmin tuottaman lähdeaineiston kaoottisuuden hallintaan, jossa lähdeaineiston tulkintaan tarvittavan semanttisen analyysin kyvyn toteuttamisen vaativuus saattaa muodostaa kehityksen pullonkaulan. Oppimispohjainen sovelluskehittäminen lienee edellytys monien tällaisten ongelmien ratkaisussa.

Vasta semanttisen analyysin kyvyn kehittyessä sovelluksissa uskottavaksi alkaa ihmisen tavoin tietokoneiden käyttäminen (DI: 20,8), ympäristön havainnointi (DI: 21,3) ja koneiden ja laitteiden käyttäminen (DI: 23,4) tulla sovellusten toimesta mahdolliseksi. Myös entuudestaan tuntemattomien asioiden tavoitteellinen itseohjattu opettelu (DI: 21,8) edellyttää semanttista analyysikykyä.

Oppimispohjaiset sovellukset voivat erityisesti olla hyödyksi ekosysteemitason ymmärtämisessä, esimerkiksi taloustieteen tai ilmastotutkimuksen teorioiden kehittämisessä sellaisen valtavan empiirisen datan pohjalta, jonka merkityksen tulkinta koettelee ihmiskyvyn rajoja tiedonlouhinta- ja havainnollistustyökalujen tuesta huolimatta. Tiedon synteesin (DI: 26,5) kyvyn kehittäminen mahdollistaisi tutkimustyön osien digitalisoinnin ja voisi tuoda tutkimustyöhön huomattavaa tuottavuuskasvua.

Horisontin takana

Vielä toistaiseksi täysin näköpiirin ulkopuolella ovat tekoälytutkimuksen alkuvuosikymmeninä tavoiteltu ihmisen tavoin ajatteleva yleistekoäly, jonka saavuttamisen yksi keskeinen kriteeri on todellisuuden käsittäminen ihmisen kaltaisten merkitysten kautta eli tietämyksen kyvyllä (DI: 29,9). Ihmiselle ilmiselvät asioiden merkitykset liittyvät opittuihin hyvin laajoihin mielleyhtymäriippuvaisuuksiin, jotka yhdessä muodostavat kullekin asialle merkityksiä. Tässä mielessä ihmisellä on toistaiseksi ollut etulyöntiasema, koska ihmisen koko

elämä on historia valtavaa määrää biologisten neuroverkkojen opetusdataa. Laajaan kulttuurin ja ihmiskäyttäytymisen mallien tuntemukseen liittyy myös arvo- ja tunneäly (DI: 25,2), johon myös tavoitteiden määrittely (DI: 26,3) tukeutuu. Näiden kykyjen toteuttamiseen vaadittavan ymmärryksen tason aikaansaaminen oppimispohjaisiin sovelluksiin on haastava tehtävä, joka todennäköisesti edellyttää merkittävää oppimispohjaisen sovelluskehityksen teknisten keinojen jatkokehittämistä ja tutkimustyötä.

6.3 Digitalisaation etenemiskaavan työmarkkinavaikutukset

Vaikutukset palvelualoihin

Tuottavuuskehityksen laajoja yhteiskunnallisia vaikutuksia ensimmäisten joukossa pohti Karl Marx. Yhtenä hänen keskeisenä ennustuksenaan on pidetty sitä, että tuotannon ns. kapitalistinen organisointi johtaisi työntekijöiden osaamisvaatimusten laskuun. Tätä oletusta on kritisoitu kirjallisuudessa laajasti siksi, että 1900-luvulla työntekijöiden osaamisvaatimukset eivät ole jatkuvasti laskeneet vaan päinvastoin nousseet, mikä heijastuu koulutetun väestön osuuden kasvussa kaikissa kehittyneissä ja kehittyvissä talouksissa.

Tuottavuuskasvua tavoitteleva prosessikehittäminen on herättänyt vastustusta työntekijöiden keskuudessa alusta saakka. Jo 1910-luvulta tunnetaan teollisuustyön taylorismin eli tieteellisen liikkeenjohdon kritiikkiä. Vaikka tieteellinen liikkeenjohto kasvattaa tuottavuutta kasvattamalla tuotantovolyymiä, työntekijät kokevat sen raskaaksi ja epäinhimilliseksi. Yhtenä keskeisenä tieteellisen liikkeenjohdon virheenä on pidetty sen oletusta, että työntekijöitä motivoisi ainoastaan palkka, kun työntekijöille on työmotivaation kannalta yhtä lailla tärkeää hyvinvointi, etenemismahdollisuudet uralla ja päätösvalta omasta työstään. (Rahman 2015, 3-6).

Mcdonaldisaatio on alun perin sosiologi George Ritzerin esittämä termi kuvaamaan prosessikehittämisen soveltamista palvelualoilla. Nimi on peräisin sen hyvästä esimerkistä, eli pikaruokaravintolakonsepti McDonald’sista, jossa palvelutyö organisoitiin uudelleen tayloristisiksi prosesseiksi.

Ritzerin mukaan mcdonaldisaatiota kuten tieteellistä liikkeenjohtoakin määrittää rationalisoinnin paradigma, joka painottaa tehokkuutta, kvantifioitavuutta, työntekijöiden taitovaatimusten korvaamista teknologialla ja epävarmuuksien kontrollointia. Ritzerin tulokulma on kulttuurintutkimuksellinen ja kriittinen. Ritzer esimerkiksi toteaa, että rationalisointi on ihmisyksilön kannalta irrationaalista, eli kohtuutonta ja epäinhimillistävää. Paluu rationalisoinnista on kuitenkin Ritzerin mukaan mahdotonta eikä se ole edes toivottavaa; huomio

tulisi keskittää keinoihin saattaa rationalisointi paremmin ihmisyyttä huomioivaksi. (Ritzer 1983, 372, 378-379).

Marxin ennustama osaamisvaatimusten lasku on yleisestä trendistä poiketen toteutunut sellaisten työtehtävien kohdalla, joissa automaatiolla on onnistuttu poistamaan itse tuotantoprosessiin liittyviä työntekijöiden taitotarpeita, kuten työkalujen käyttötaitoja teollisuudessa tai ymmärrystä ruoanlaitosta pikaruokaravintoloissa. Samaan aikaan tuottavuuskasvu tällaisessa tuotannossa on yhdistettynä hyödykkeiden kysynnän rajallisuuteen johtanut työvoiman vapautumiseen työmarkkinoiden käyttöön muissa tehtävissä, jotka tyypillisesti ovat edellyttäneet korkeampaa osaamista.

Opinnäytetyön arviointimenetelmän tuottamien tulosten (Taulukko 3) valossa, digitalisaation voidaan odottaa etenevän työnkuvien murrosmaisen täysautomaation sijaan enemmänkin asteittaisena työnkuvien murenemisena erikoistuneiden digitalisoivien sovellusten toteutettaviksi. Useimmissa ammateissa ei keskitytä yksinomaan toteuttamaan mitään tiettyä Lampelto-kykyä, vaan työnkuvat ja organisaatioiden tuotanto koostuvat monimutkaisesta yhdistelmästä erilaisia Lampelto-kykyjä. Sovelluskehittämisen soveltamisalan kasvutrendi näyttäytyy populaaristi murroksen tapaisena lähinnä siksi, että se laajenee koskemaan muitakin sektoreita kuin jalostusteollisuutta, jossa teknologinen tuottavuuskehittäminen on aiemmin vähentänyt työntekijöiden tarvetta voimakkaasti. Teknologiavetoisen tuottavuuskehittämisen etenemiskaava on kuitenkin sama kuin ennenkin historiassa. Tuotannon työnkuvat järjestetään edelleen todennäköisesti niin, että ihmiset suorittavat sellaisia tehtäviä, joita ei vielä voida tehokkaasti automatisoida (digitalisoida). Tämän uuden vastuunjaon mahdollistamiseksi nykyisiä työnkuvakokonaisuuksia on luontevaa muuttaa toisenlaisiksi (Tamminen & Pesälä 2016, 155). Tätä asteittaista digitalisoivien sovellusten etenemiskaavaa palvelu- ja asiantuntija-aloilla voisi kutsua *digitaaliseksi mcdonaldisaatioksi*.

Kun sovelluskehittämisen uusien aluevaltausten seurauksena voidaan alkaa automaatiolla korvata palvelu- ja asiantuntija-alojen tiettyjä osaamistarpeita, helposti koneellistettavia palvelualoja aiemmin lähinnä koskettanut mcdonaldisaatio alkaa koskettaa yhä uusia aloja ja taitoja. Tämän kehityksen johdonmukainen seuraus olisi se, että työn taitovaatimusten laskiessa näiden alojen työntekijöiden neuvotteluvalta heikkenisi, koska yhä useampi voisi periaatteessa suoriutua tehtävistä, jotka edellyttävät ihmisille lähestulkoon itsestään selviä mutta toistaiseksi vielä vaikeasti digitalisoitavia kykyjä. Tällaisia ovat esimerkiksi taulukossa 3 tunnistetut ympäristön havainnointi (DI: 21,3), luonnollisen kielen tulkinta (DI: 19,6) ja arvo- ja tunneäly (DI: 25,2), jotka eivät edellytä korkeaa koulutusta mutta joita on haastavaa uskottavasti digitalisoida Turingin testiä vastaavalla tavalla.

Mikäli tämän trendin seurauksena Marxin kritiikin pohjana ollut ilmiö uusien ammattien syntymisestä korkeampaa osaamista vaativille aloille jatkuisi, palvelu- ja asiantuntija-alojen tuottavuuskehityksen ja tuotteiden rajallisen kysynnän looginen seuraus olisi osaamisvaatimusten jatkuva kasvu eli päinvastainen kuin Marxin ennustuksissa. Tämä kehitys ei ole työllisyyskehityksen kannalta täysin ongelmaton.

Osaamisvaatimusten rajaton kasvu johtaisi todennäköisesti tilanteeseen, jossa yhä harvempi kykenisi tai ehtisi enää opiskelemaan ja omaksumaan jatkuvasti haastavampia taitoja uusilta osaamisaloilta. Kehitys voisi saavuttaa (tai on jo alkanut saavuttaa) pisteen, jossa iäkkäät ja matalasti koulutetut eivät enää kykenisi sopeutumaan nopeaan muutokseen. Tällöin työmarkkinoille olisi alkanut syntyä rakenteellista työttömyyttä, joka johtuisi ihmisaivojen muuntautumiskyvyn rajoista eikä koulutuksen itsensä puutteesta. Toteutuessaan laajasti, ilmiö näyttäytyisi samanaikaisesti korkeana työttömyytenä ja asiantuntevan työvoiman pulana. (Tamminen & Pesälä 2016, 172-173).

Sekä ohjelmointi- että oppimispohjaisen sovelluskehittämisen aloilla tämä työmarkkinailmiö vaikuttaisi olevan valloillaan, koska osaavasta työvoimasta on pulaa, kuten Ghosh (2017) ja Hollingworth & Harvey-Price (2013, 8) toteavat.

Kuten Ritzer (1983, 379) toteaa, tässä prosessikehittämisessä olisi syytä huomioida inhimillinen tekijä, eli tuotannon rationalisoinnin vaikutus työn ergonomiaan kuten työhyvinvointiin. Saman haasteen voisi olettaa pätevän myös digitaaliseen mcdonaldisaatioon.

Vaikutukset asiantuntijatyön työnkuviin

Taulukon 3 mukaan seuraavaksi digitalisaatio etenisi ennen kaikkea strukturoidun tiedon (DI: 4,7), visuaalisen informaation tulkinnan (DI: 9), elehdinnän (DI: 9,5) ja luonnollisen kielen tuottamisen (DI: 10,1) alueille.

Saatamme olla lähellä tilannetta, jossa rutiininomainen lakiasiantuntijan tai lääkärin työ tulee mcdonaldisoiduksi, ja ihmisen rooli teollisen prosessin legitimoijana kasvaa (Tamminen & Pesälä 2016, 154). Palvelu- ja asiantuntijatyön työnkuvista poistuisi asteittain rutiininomaisia tehtäviä, mikä työnkuvan sisällöstä riippuen voi näyttäytyä joko rutiininomaisen asiantuntijatyön asteittaisena mcdonaldisaationa tai tietotyövoittoisen asiantuntijatyön muuttumisena kohti koneellisten assistenttien johtamista.

Tietotyön käsitteen käyttökelpoisuuden ylläpitämiseksi tietotyön määritelmästä on hyödyllistä rajata pois sellaiset tehtävät, jotka ovat triviaalisti digitalisoitavissa (Tamminen & Pesälä 2016, 199). Näin tietotyön käsite ei viittaisi datan prosessointiin vaan inhimillisten tietosisältöjen käsittelyyn ja tuotantoon.

Palvelualan digitaalisen mcdonaldisaation aikaansaaman tuottavuuskasvun seurauksena voidaan olettaa, että työvoimaa vapautuu tietotyötehtävien työmarkkinoiden käyttöön. Toisaalta osaamisvaatimusten kasvu saattaa tästä huolimatta muodostaa rekrytointihaasteita, jos se etenee nopeammin kuin työvoimareservi kykenee omaksumaan uusia taitoja.

Ajan saatossa taulukon 3 mukaan tietotyön työnkuvista jäisivät jäljelle lähinnä organisaatioiden tavoitteiden määrittely ja muut strategisen johdon ydintehtävät, joiden suorittaminen ihmistyönä merkittävissä määrin liittyy siihen, että organisaatioiden tehtävä loppuen lopuksi on palvella ihmisten niille asettamia rooleja. Osana samaa kehitystä työn määritelmä todennäköisesti muuttuu.

Turvatut ammatit

Monet digitalisaatiolta välittömässä lähitulevaisuudessa turvatun oloiset ammatit ovat ironisesti kyllä matalapalkkaisia ja fyysisiä, koska niiden digitalisointia rajoittaa kaoottisuuden ongelmat. Ihminen on parempi intuitiivisesti vastaamaan kaoottisuuteen omassa toimintaympäristössään kuin mikään toistaiseksi kehitetty sovellus. Tällaisia kaoottisia ammatteja löytyy esimerkiksi rakennus-, siivous-, logistiikka- ja hoitoaloilta. Ellei oppimispohjaisessa sovelluskehityksessä löydetä kaoottisuuden hallintaan keinoja kuten valmiita kaupallisia sovellusmoduuleja, kaoottisten työnkuvien digitalisaatio saattaa tapahtua vasta, kun ympäristöjen kaoottisuus poistetaan muokkaamalla ympäristöt helposti koneellistettaviksi tai työnkuvien tarve poistetaan muilla innovaatioilla.

Hyvä esimerkki työtehtävien digitalisoinnin kaoottisuushaasteesta on rakennustyömaan työntekijän korvaamisessa robotilla. Vaikka monet rakennustyömaan työntekijän osatehtävät ovat kohtalaisen yksinkertaisia robotinkin ratkaista erikseen, rakennustyömaalla liikkuminen on kaoottinen tehtävä, joka edellyttää motorisia taitoja (DI: 20,4), ympäristön havainnointia (DI: 21,3) ja koneiden ja laitteiden käyttöä (DI: 23,4). Jokainen rakennus on hieman erilainen ja rakentamisen etenemisen myötä rakennukset muuttuvat ja työkalut vaihtuvat. Rakennustyömaalla toimiva ihminen on siis oppinut hallitsemaan tätä epävarmuutta luomalla kognitiivisia malleja lukuisista erilaisista ympäristöjen stereotyypeistä, ja vaikka virheitäkin saattaa sattua, pääosin ihminen suoriutuu tehtävistään työmaalla varsin

hyvin. Lukemattomien muuttujien ongelmaa on ratkaistu siirtämällä kompleksisuuden ongelmien ratkaisu työnjohdolle, projektijohdolle ja projektisuunnittelijoille.

Perinteisesti automaatiossa tällaisia ongelmia on ratkaistu muotoilemalla koko prosessi koneellistettavuuden ehtojen ympärille. Tästä hyvä esimerkki on liukuhihna. Rakennustyömaalla tämä voisi tarkoittaa esimerkiksi nostureista käsin piirustusten pohjalta tehtävää alhaalta-ylös -rakentamista, joka muistuttaisi additiivista valmistusta. Toinen mahdollisuus olisi valmistaa talojen elementit 3D-tulostamista muistuttavalla tekniikalla viimeistä pintakäsittelyä vaille valmiiksi ja vain latoa elementit paikoilleen. Tällaisessa esivalmistettujen elementtien automaattisessa valmistuksessa keskeinen tavoiteltava kilpailutekijä olisi tuotannon joustavuus eli Kutayn (1989, 7-9) mainitsema laajuustaloudellinen etu (engl. *economy of scope*).

Itse sovelluskehittämiselle ei välittömässä lähitulevaisuudessa ole nähtävissä loppua, mutta alan työpaikkojen keskimääräiset osaamisvaatimukset todennäköisesti nousevat edelleen oppimispohjaisen sovelluskehittämisen yleistyessä menetelmänä. Oppimispohjainen sovelluskehitys monimutkaisten ongelmien ratkaisussa ei toistaiseksi ole kilpailukykyistä ohjelmointipohjaisen sovelluskehityksen kanssa, mutta triviaalien ongelmien ratkaisu ihmisen ohjeistamana saattaa onnistua piankin. Tämä avaa uusia mahdollisuuksia älykkäiden agenttien hyödyntämiselle.

Oppimispohjaisen sovelluskehittämisen etu on kuitenkin sen sovellettavuus sellaisiin ongelmiin, jotka ovat käytännössä olleet ohjelmointipohjaisen sovelluskehityksen ulottumattomissa. Sovelluskehityksessä kannattaa siis keskittyä valitsemaan ne työkalut, jotka ovat ratkaistavien ongelmien kannalta tehokkaimmat.

Lisäksi oppimispohjainen sovelluskehittäminen saattaa keskipitkällä aikavälillä alkaa korvata ohjelmointipohjaista kehittämistä, kun ohjelmointityön digitalisoivat sovellukset kypsyvät tuotteiksi.

7 Johtopäätökset

7.1 Yhteenveto

Opinnäytetyössä pyrittiin digitaalisen tuottavuuskehityksen luonteen, keinovalikoiman, automatisoitavuuden arvioiden sekä automaation taloudellisten kannustinten analyysin avulla esittämään strategiatyössä ja päätöksenteossa hyödynnettävä malli, jonka avulla voitaisiin arvioida digitalisoinnin mahdollisuuksia ja toisaalta arvioida markkinatoimijoiden välistä kilpailutilannetta.

Lisäksi pyrittiin mallin yksinkertaisen arviointimenetelmän avulla esittää arvioita löydösten mukaisella tavalla etenevän digitalisaation työmarkkinavaikutuksista.

Tutkimuskysymykseen liittyvästä aiemmasta kirjallisuudesta tunnistettiin Thórissonin ym. (2016) laskennallisen tehtäväteorian kriteeristö. Tämän kriteeristön vaatimuksiin tyydyttämään tunnistettiin kirjallisuudesta Lampelton (2013) haastattelututkimukseen perustuvat työtehtävien komponenttiset kyvyt (Lampelto-kyvyt) sekä ihmisen toimintaympäristön ongelma-alueiden kategorisointiin kehitetty Cynefin-viitekehys, jota päätettiin hyödyntää algoritmisen ratkaisun löytämisen haastavuuden kategorisointiin, kuten Graves (2010) oli tehnyt. Lisäksi kirjallisuudesta tunnistettiin Kutayn (1989) suositukset automaation taloudellisen potentiaalin arvioimisessa. Analyysin pohjalta syntyi abstrakti mutta erilaisin arviointimenetelmin kvantifioitavaksi tarkoitettu malli.

Mallin arviointia varten tehtävien automaation keskinäisen taloudellisuuden arviointiin päätettiin käyttää yksinkertaista keskipalkkatietoihin ja tehtävien yleisyyteen tukeutuvaa arvonlisäysarviota, joka tuotti pistemäärän skaalalla 0-10. Kirjallisuus ei kuitenkaan tarjonnut keinoja kvantifioida työtehtävien automaation teknologista haastavuutta. Tästä syystä arviointimenetelmän teknologisen vaativuuden mittaristoksi kehitettiin NASA-TLX -kyselyn tapainen pisteytysjärjestelmä, jollaa opinnäytetyön tekijä pisteytti omaan arvioonsa pohjaten Lampelto-kykyjen ongelmienratkaisun haastavuuden eri Cynefin-ongelma-alueilla siten, että kaoottisuus sai kaksinkertaisen painoarvon ja haastavuuspisteet ynnättiin yhteen. Tällöin vaativuusarvion suurimmaksi mahdolliseksi vaihteluväliksi muodostui 0-40 ja mallin tuottaman digitalisaatioindeksin vaihteluväliksi -10 – 40.

Arviointimenetelmän käytön tuloksena syntyi taulukko (Taulukko 3) Lampelto-kykyjen keskinäisestä digitalisaatiopotentiaalista, josta voitiin todeta, että jo sovelluskehityksellä paljolti digitalisoidut Lampelto-kyvyt kuten datan vastaanottaminen ja tulosten ilmoittaminen saivat hyvin korkean digitalisaatiopotentiaalin eli alhaisen digitalisaatioindeksin, kun taas

korkeita digitalisaatioindeksejä saivat selvästi tietotyön ydinkyvyiksi tunnistettavat Lampelto-kyvyt kuten tietämys, tiedon synteesi, tavoitteiden määrittely sekä arvo- ja tunneäly. Mielenkiintoiset havainnot liittyivät näiden ääripäiden väliseen alueeseen, joissa esimerkiksi ohjelmointi vaikuttaisi todennäköisesti tulevan aiemmin digitalisoiduksi kuin luonnollisen kielen ymmärtäminen.

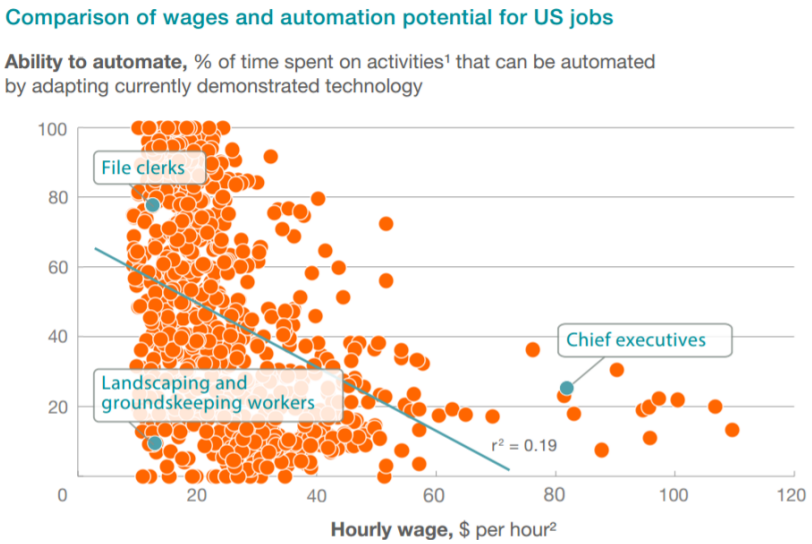
Arviointimenetelmän tulosten ja toisaalta oppimispohjaisen sovelluskehittämisen esimerkkien pohjalta tehtiin oletus, että ainakaan useimmilla palvelu- ja asiantuntija-aloilla täysautomaatioon vaadittu teknologia ei ollut vielä lähitulevaisuudessa saavutettavissa, vaan että digitalisointi etenisi asteittain teknologian kehittyessä. Tässä valossa esitettiin arvioita asteittaisesta digitalisaatiosta seuraavista työmarkkinailmiöistä. Työmarkkinailmiöiden arviot voitaneen verifioida vasta tulevaisuudessa.

Arvioinnin tulosten valossa voidaan todeta, että malli epätarkankin arviointimenetelmän avulla kykeni tarjoamaan pääpiirteittäin uskottavalta tuntuvan tilannekuvan eri työtehtävien taustalla olevien kognitiivisten kykyjen automaatiojärjestyksestä näiden teknologisen haastavuuden ja toisaalta automaation taloudellisen potentiaalin kautta. Tämä tukisi oletusta, että mallia voisi kehittyneemmän arviointimenetelmän kanssa hyödyntää organisaation strategiatyössä ja päätöksenteossa.

7.2 Tulosten vertailu kirjallisuuteen

Arviointimenetelmän tuloksia on vaikea verrata kirjallisuuteen, koska esiteltyä mallia muistuttavia malleja löytyi vain yksi. McKinsey & Company -konsulttiyhtiön tutkijat Chui, Manyika ja Miremadi olivat vuosina 2015 ja 2017 julkaisseet tutkimustyönsä tuloksia.

Chui, Manyika ja Miremadi (2015) arvioivat digitalisoinnin taloudellista potentiaalia etsimällä Yhdysvaltain työministeriön O*Net Online -tietokannasta sellaiset työnkuvissa toistuvat tehtävät, jotka olivat jokseenkin digitalisoitavissa vuoden 2015 teknologialla, ja kertoivat näiden työaikaosuudet eri ammateissa näistä ammateista maksetuilla keskipalkoilla. Ainoastaan kuvana artikkelissa ilmoitetut tulokset (Kuvio 7) vaikuttaisivat olevan jokseenkin linjassa opinnäytetyön mallin arviointimenetelmän tulosten kanssa. Chui, Manyika ja Miremadi (2015) toteavatkin, että työstä saatava tuntipalkka ei ole hyvä ennustaja työnkuvien automatisoitavuudelle, vaikka nämä korreloivatkin keskenään.

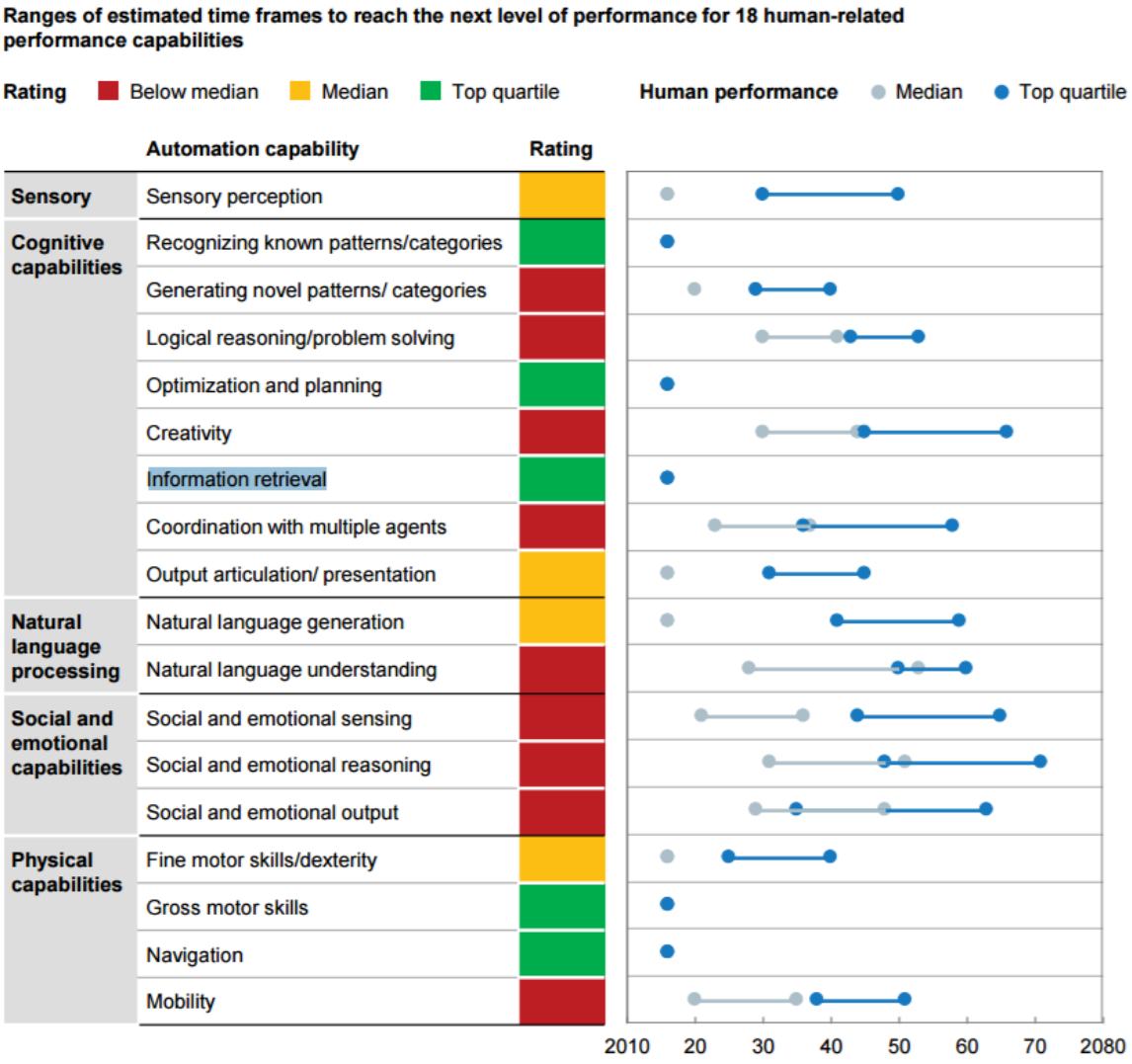


Kuvio 7. Ammattien työtehtäväkomponenttien automatisoitavuus ja ammattien tuntipalkat (Chui, Manyika & Miremadi, 2015)

Vuonna 2017 julkaistussa laajemmassa tutkimusraportissa, jossa pyrittiin arvioimaan eri liiketoiminnan alojen automatisoitavuutta, Chui, Manyika & Miremadi (2017, 36) kuvasivat tunnistamiensa 18 kyvyn teknologisen haastavuuden arviointimenetelmäänsä vain toteamalla, että tutkijat arvioivat kykyjen työelämässä hyödyllisen tason vaativuutta suhteessa tyypillisen ihmisen kykeneväisyyteen kolmiportaisella asteikolla ja vertailivat tätä olemassa olevan teknologian kykyyn suoriutua näiden soveltamisesta (Chui, Manyika & Miremadi 2017, 36).

Tämä antaisi ymmärtää, että tässä opinnäytetyössä käytetty teknologisen haastavuuden Cynefin-viitekehykseen tukeutuva kykyjen vaativuuden arviointimenetelmä olisi moniuloitteisempi kuin aiemmin kirjallisuudessa käytetyt. Chui, Miremadi & Manyika (2017) eivät yksityiskohtaisesti kuvanneet käyttämiään menetelmiä tulevaisuusennusteidensa (Kuviot 4 ja 8) taustalla.

Kuvio 8. Ennustuksia eri kykyjen digitalisoinnin mahdollistavan teknologian kehittämisvuosista tyypillisen ihmisen (harmaalla) ja huipputasoisen ihmisen (sinisellä) tasoilla (Chui, Manyika & Miremadi 2017, 72)



Taulukossa 4 on vertailtu vertailuun soveltuville Lampelto-kyvyille arviointimenetelmän tuottamia digitalisaatioindeksejä ja C-muuttujaa (Taulukko 3) kuvion 8 arvioihin. Taulukko 4 on muodostettu silmämääräisesti päättelemällä kuvion 8 sinisten ja harmaiden vuosivaihteluvälien keskipisteet, laskemalla näistä vuosiluvuista keskiarvo ja vähentämällä keskiarvosta luku 2017. Taulukossa on yhdistetty Lampelto-kyvyt CMM-kykyihin vertailemalla niiden nimiä.

Taulukko 4. Lampelto- ja CMM-kykyjen digitalisaatioennusteiden vertailu (suom. Lilja Tamminen)

Lampelto-kyky	**CMM-kyky**	**C**	**DI**	**Ennuste vuosina**
Strukturoitu tieto	Tiedon muistelu	9	4,7	0
Luonnollisen kielen tuottaminen	Luonnollisen kielen tuottaminen	17	10,1	15,5
Kuvioiden tunnistaminen datasta	Uusien kuvioiden ja kategorioiden tunnistaminen	22	12,1	10
Luonnollisen kielen tulkinta	Luonnollisen kielen ymmärtäminen	23	19,6	30,5
Motoriset taidot	Liikkumiskyky	25	20,4	18,5
Ympäristön havainnointi	Aistinvarainen ympäristön havainnointi	26	21,3	13,5

Taulukosta 4 käy ilmi, että arviointimenetelmän mukaisesti syntyneet arviot eri kykyjen teknologisesta haastavuudesta ja taloudellisesta potentiaalista eivät ole täysin yhteneväiset Chui, Manyika ja Miremadin (2017, 72) arvioiden kanssa. Merkittävin poikkeama liittyy luonnollisen kielen tulkinnan (Lampelto 2013, 80) ja luonnollisen kielen ymmärtämisen (Chui, Manyika & Miremadi 2017, 35) arvioihin.

Myöskään arviointimenetelmän arviot kykyjen suorittamisen yleisyydestä eivät ole täysin yhteneväiset Chui, Manyika ja Miremadin (2017, 38) työaikaosuusarvioiden kanssa, jotka perustuvat Yhdysvaltain työministeriön tilastotietoihin eri ammattien työsisällöistä Yhdysvalloissa. Taulukossa 5 on vertailtu Lampelto- ja CMM-kykyjen soveltamisen yleisyysarvioita.

Taulukko 5. Vertailu CMM-kykyjen yleisyysarvioiden ja arviointimenetelmässä syntyneiden Lampelto-kykyjen yleisyysarvioiden välillä

Lampelto-kyky	**Vastaava CMM-kyky**	**$Y^{1,5}$**	**Chui, Manyika & Miremadi**
Visuaalisen informaation tulkinta	Tunnettujen kuvioiden tunnistaminen	11/11	67 %
Luonnollisen kielen tuottaminen	Luonnollisen kielen tuottaminen	8/11	46 %
Ympäristön havainnointi	Aistinvarainen ympäristön havainnointi	8/11	41 %
Strukturoitu tieto	Tiedonhaku	5/11	38 %
Luonnollisen kielen tulkinta	Luonnollisen kielen ymmärtäminen	8/11	35 %
Motoriset taidot	Suuripiirteiset motoriset taidot	5/11	17 %
Päättely	Looginen päättely tai ongelmanratkaisu	5/11	13 %

Yhtenä merkittävänä syynä taulukkojen 4 ja 5 eroavaisuuksiin voi olla Lampelto- ja CMM-kykyjen toisistaan poikkeavat ja suppeat määritelmät. Lampelto (2013, 80-85) määrittelee nimettyjen kykyjen sisällöt laajemmin kuin Chui, Manyika ja Miremadi (2017, 35) ja toisaalta useimmat Lampelto-kyvyt ja CMM-kyvyt eivät ole suoraan vertailukelpoisia. Erityisesti CMM-kyky *luonnollisen kielen ymmärtäminen* antaa muotoilunsa puolesta ymmärtää,

että kyse on huomattavasti Lampelto-kykyä *luonnollisen kielen tulkinta* laajemmasta kyvykkyydestä. Kattava laskennallinen tehtäväteoria mahdollistaisi näiden yhdistämisen, jos määritelmät olisivat tarpeeksi tarkat.

Pääpiirteittäin esitellyn arviointimenetelmän tuottamat tulokset ovat kuitenkin linjassa Chui, Manyika & Miremadin (2017, 72) arvioiden kanssa. Suurin yhteneväisyys on kuitenkin tämän opinnäytetyön esittämän mallin taustaoletuksissa. Näitä ovat esimerkiksi se, että automaation taloudelliset hyödyt eivät rajoitu henkilöstökustannussäästöihin (Chui, Manyika & Miremadi 2017, 68) ja että kokonaistaloudelliset hyödyt voivat ohjata tutkimus- ja tuotekehitysinvestointeja (Chui, Manyika & Miremadi 20167, 95).

7.3 Tutkimuksen rajoitteet ja jatkotutkimus

Arviointimenetelmän keskeisin puute liittyy esitellyn abstraktin digitalisaatiopotentiaalimallin sisäänrakennettuun epäsuhtaan vastaamisessa: vaikka teknologinen haastavuus käyttäytyy mallissa tutkimus- ja kehityskustannuksen tavoin, sille on erittäin vaikea arvioida mitään rahamääräistä arvoa, joka mahdollistaisi sen suoran vertailun digitalisoivan sovelluksen markkinapotentiaaliin. Tämä johtuu osittain siitä, että tutkimus- ja kehitysinvestointien kannattavuus erityisesti perustutkimuksessa vaihtelee äärimmäisen paljon.

Opinnäytetyön analyysiä rajoittaa erittäin merkittävästi myös se, että tekoälytutkimuksen laskennallinen tehtäväteoria ja laskennallinen neurotiede ovat mallin tarpeiden kannalta edelleen alkutekijöissään. Tarkka työtehtävien digitalisaation etenemisjärjestyksen mallintaminen edellyttäisi sekä kehittynyttä laskennallista tehtäväteoriaa että huomattavaa tutkimustyötä tehtäväteorian tunnistamien komponenttien algoritmisen vaativuuden määrittämiseksi ja kustannusvertailtavuuden aikaansaamiseksi. Mikäli sellainen tietomäärän piste saavutetaan, mahdollistaisi se hyvinkin tarkan digitalisaatiopotentiaalin arvion esittämisen. Toisaalta, mikäli sellainen piste joskus saavutetaan, olemme silloin todennäköisesti jo hyvin lähellä uusia läpimurtoja oppimispohjaisten sovelluskehittämisen soveltamisalueiden valtauksessa, koska tehtävän ongelman haastavuuden määrittelykyky on tueksi ongelman ratkaisun löytämisessä.

Koska arviointimenetelmän taloudellisen potentiaalin arviointitapa tukeutuu vertailtavien digitalisoitavien tehtävien tai kykyjen keskinäiseen vertailuun, arviointimenetelmä toimii vain keinona vertailla valittuja kykyjä tai tehtäviä keskenään. Arviointimenetelmää ei siis voi käyttää toisistaan poikkeavien syötteiden välillä syntyneiden tulosten vertailuun. Tätä voidaan tasapainottaa käyttämällä arvioinnissa laajaa määrää erilaisia vaihtelevan haasta-

vuuden komponenttisia kykyjä, joista jotkut ovat ihmisillekin erittäin haastavia ja toiset hyvin yksinkertaisia, tai yksinkertaisesti kehittämällä toinen tapa yhteensovittaa eli kvantifioida digitalisaatiopotentiaalimallin muuttujat C ja B.

Koska digitalisaatiopotentiaalimallin esitelty arviointimenetelmä ja todennäköisesti muutkin mallin arviointiin mahdollisesti kehitettävät arviointimenetelmät tukeutuvat tietyn ajankohdan dataan automaation taloudellisesta potentiaalista, ne ovat vääjäämättä sidottuja arvioinnissa käytetyn datan ajankohtaan. Malli ei siis ole ongelmattomasti sovellettavissa jo automatisoitujen tehtävien järjestyksen jälkikäteiseen ennustamiseen ilman historiallista dataa. Tässä mielessä malli ja sen arviointimenetelmät soveltuvat lähtökohtaisesti vain vielä toteuttamattoman digitalisoinnin arviointiin. Mallin käyttötarkoituksen eli organisaatioiden strategiatyön ja päätöksenteon kannalta se ei kuitenkaan muodosta ongelmaa. Historiallinen katsaus mahdollistaisi kuitenkin mallin ja arviointimenetelmän kriittisemmän tarkastelun ja edelleen kehittämisen ennustamisen työkaluna.

Kuten Thórisson ym. (2016, 5) toteaa, on tarve kattavalle laskennalliselle tehtäväteorialle, joka tarjoaisi keinot pilkkoa ihmistyötehtävien luonne Lampelto- ja CMM-kykyjä atomisemmiksi komponenteiksi, joita voisi yhdistellä korkeamman tason tehtäviksi ja joiden laskennallinen vaativuus yksin ja yhdessä olisi tunnettu. Toisaalta, mikäli koskaan pääsemme aidosti atomiselle tasolle laskennallisen neurotieteen ymmärryksessä, olemme tuolloin jo ratkaisseet ainakin suurimman osan ihmisaivojen ja kognition arvoituksista, koska meillä olisi silloin keino mallintaa ainakin suurin osa ihmisaivojen kognitiivisten kykyjen matalan tason toiminnasta. Tässä mielessä voi olla, että oppimispohjainen sovelluskehitys ehtii algoritmisesti ratkaisemaan ainakin huomattavan osan liiketoiminnallisista ongelmista ennen kuin olemme kehittäneet keinot ihmisinä ymmärtää miten ne on ratkaistu.

Yksi tapa lähestyä laskennallisen tehtäväteorian ongelmaa voisi olla hyödyntää algoritmisen ratkaisun löytämisen aikavaativuustietoja ja toteutetuista oppimispohjaisen sovelluskehityksen sovelluksista, joiden toteuttamille kyvyille tai taidoille meillä olisi yhteensopiva joskin rajallinen tehtäväteoria. Näin saisimme tietoa laskennallisesta vaativuudesta ilman, että meidän täytyisi ymmärtää sen rakenteellisia (objektiivisia) taustatekijöitä.

7.4 Opinnäytetyöprosessista

Opinnäytetyön esittämän mallin teknologisen eli laskennallisen vaativuuden kvantifioitavuuden haasteet eivät tulleet yllätyksenä. Tästä huolimatta organisaatioiden strategiatyötä

helpottamaan tarkoitetun mallin arviointimenetelmän kehittäminen erityisesti laskennallisen vaativuuden suurinpiirteiseksi arvioimiseksi ilman välitöntä yhteyttä laskennalliseen kompleksisuusteoriaan oli erittäin opettavainen kokemus.

Opinnäytetyöhön käytetystä ajasta enemmistö kului pelkästään relevanttiin kirjallisuuteen tutustumiseen mm. ohjelmistokehityksen, tietojenkäsittelytieteen, tekoälytutkimuksen, systeemiteorian ja työntutkimuksen alueilla. Vain murto-osa tästä aineistosta päätyi opinnäytetyössä hyödynnettäväksi. Erityisesti Cynefin-viitekehys Gravesin (2010) tulkitsemana oli mielenkiintoinen löytö kirjallisuudesta, sillä opinnäytetyön kirjallisuustutkimustyössä tuli toistuvasti vastaan Cynefin-viitekehyksen jaottelua muistuttavia havaintoja systeemiteorian ja tekoälytutkimuksen kirjallisuudessa hieman eri nimityksin. Opinnäytetyön yhtenä kontribuutiona voikin ehkä pitää Cynefin-viitekehyksen esittelyä erityisesti oppimispohjaisen sovelluskehittämisen kontekstissa keinona hahmottaa liiketoimintaympäristön ongelmien ratkaisun digitalisoitavuuden teknologisia haastavuusasteita.

Lähdeluettelo

Balog, M., Gaunt, A. L., Brockshmidt, M., Nowozin, S. & Tarlow, D. 2017. DeepCoder : Learning to write programs. 5th International Conference on Learning Representations. Luettavissa: https://arxiv.org/abs/1611.01989. Luettu: 6.3.2017.

Banker, R. D., Datar, S. M., Kemerer, C. F. & Zweig, D. 1993. Software complexity and maintenance costs. Communications of the ACM, 36, 11, s. 81-94. Association for Computing Machinery. Luettavissa: http://www.pitt.edu/~ckemerer/CK%20research%20papers/SwComplexityAndMaintenanceCost_BankerDatar93.pdf. Luettu: 10.4.2017.

BBC 2014. AI: 15 key moments in the story of artificial intelligence. British Broadcasting Company. Luettavissa: http://www.bbc.co.uk/timelines/zq376fr. Luettu: 17.4.2017.

Beck, M. W. 2013**.** Visualizing neural networks in R – update. Luettavissa: https://www.r-bloggers.com/visualizing-neural-networks-in-r-update/. Luettu 2.3.2017

Boh, W. F., Slaughter, S. A. & Espinosa, J. A. 2007. Learning from experience in software development: a multilevel analysis. Management science 53, 8, s. 1315-1331. Informs. Luettavissa: https://www.researchgate.net/publication/220535077_Learning_from_Experience_in_Software_Development_A_Multilevel_Analysis. Luettu: 10.4.2017.

Brooks, R. 1990. Elephants don’t play chess. Robotics and Autonomous Systems 6, s. 3-15. Elsevier. Amsterdam. Luettavissa: http://people.csail.mit.edu/brooks/papers/elephants. Luettu: 2.5.2017.

Callahan, S. 2009. A simple explanation of the Cynefin framework. Luettavissa: http://www.anecdote.com/2009/04/a-simple-explanation-cynefin-framework/. Luettu: 6.3.2017.

Cilliers, P. 1998. Complexity and postmodernism: Understanding complex systems. Routledge. Lontoo. ISBN 0-203-01225-9

Chui, M., Manyika, J. & Miremadi, M. 2015. Four fundamentals of workplace automation. McKinsey Quarterly November 2015. McKinsey & Company. Luettavissa: http://www.mckinsey.com/business-functions/digital-mckinsey/our-insights/four-fundamentals-of-workplace-automation. Luettu: 3.3.2017

Chui, M., Manyika, J. & Miremadi, M. 2017. A future that works: automation, employment and productivity. McKinsey Global Institute. McKinsey & Company. Luettavissa: http://www.mckinsey.com/~/media/McKinsey/Global%20Themes/Digital%20Disruption/Harnessing%20automation%20for%20a%20future%20that%20works/MGI-A-future-that-works_Full-report.ashx. Luettu: 25.3.2017.

Coppin, B. 2004. Artificial intelligence illuminated. Jones & Bartlett Publishers. Lontoo/Mississauga. ISBN 0-7637-3230-3.

Dettmer, W. H. 2011. Systems thinking and the Cynefin framework – a strategic approach to managing complex systems. Goal Systems International. Luettavissa: http://www.goalsys.com/books/documents/Systems-Thinking-and-the-Cynefin-Framework-Final.3.pdf. Luettu: 16.3.2017.

Foote, D. F. 2016. A brief history of artificial intelligence. Dataversity. Luettavissa: http://www.dataversity.net/brief-history-artificial-intelligence/. Luettu: 14.4.2017.

Galjaard, S., Hofman, S. & Ren, S. 2015. New opportunities to optimize structural designs in metal by using additive manufacturing. Advances in Architectural Geometry 2014, s. 79-93. Luettavissa: https://www.researchgate.net/publication/278681542_New_Opportunities_to_Optimize_Structural_Designs_in_Metal_by_Using_Additive_Manufacturing. Luettu: 26.3.2017.

Ghosh, P. 2017. 2017 Machine learning trends. Dataversity Education. Luettavissa: http://www.dataversity.net/2017-machine-learning-trends/. Luettu: 10.3.2017.

Graves, T. 2010. More on meta-methodology (‘Beyond-Cynefin’ series). http://weblog.tetradian.com/2010/03/01/more-on-meta-methodology/. Luettu: 10.3.2017.

Harman, M. 2012. The role of artificial intelligence in software engineering. University College London. Lontoo. Luettavissa: http://www0.cs.ucl.ac.uk/staff/mharman/raise12.pdf. Luettu: 6.3.2017

Hart, S. G. 2006. NASA task load index (NASA-TLX); 20 years later. NASA-Ames Research Center. Luettavissa: https://humansystems.arc.nasa.gov/groups/tlx/publications.php. Luettu: 10.3.2017.

Hollingworth, L. & Harvey-Price, A. 2013. Technology and skills in the digital industries. UK Commission for employment and skills. Luettavissa: https://www.gov.uk/government/publications/technology-and-skills-in-the-digital-industries. Luettu: 10.4.2017.

Hong, K. & Kim, Y. 2001. The critical success factors for ERP implementation: an organizational fit perspective. Information & Management, 40, s. 25-40. Elsevier.

Huang, X., Kwiatkowska, M., Wang, S. & Wu, M. 2017. Safety verification of deep neural networks [v3]. Luettavissa: https://arxiv.org/abs/1610.06940. Luettu: 6.5.2017.

Ingo, H. 2002. Kollektiivisesta ajattelusta ja neuroverkoista. Luettavissa: http://avoinelama.fi/hingo/filosofia/kollektiivinenajattelu.html. Luettu: 3.12.2016.

Kutay, A. 1989. The economic impact of automation technology. Robotics Institute, School of Computer Science. Carnegie Mellon University. Luettavissa: http://repository.cmu.edu/robotics/629/. Luettu: 3.3.2017.

Lampelto, P. 2013. Understanding knowledge work and the performance potential of its computerization: Case IBM's Watson. Pro gradu -tutkielma. Hallintotieteiden laitos. Tampereen yliopisto. Luettavissa: http://urn.fi/URN:NBN:fi:uta-201310091470. Luettu: 3.3.2017.

Liu, P. & Li, Z. 2012. Task complexity: a review and conceptualization framework. International Journal of Industrial Ergonomics, 42, s. 553-568. Elsevier. Luettavissa: http://ftp.demec.ufpr.br/disciplinas/EME743/Artigos%20Ergonomia%20geral/Task%20complexity%20%20a%20review.pdf. Luettu: 3.3.2017.

Littmann, E. 2002. Trends in neural network research and an application to computer vision. Bielefeld University. Luettavissa: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.45.5002&rep=rep1&type=pdf. Luettu: 2.5.2017.

Lohr, S. 2017. A.I. is doing legal work. But it won’t replace lawyers, yet. New York Times 20.3.2017, s. B1. Luettavissa: https://www.nytimes.com/2017/03/19/technology/lawyers-artificial-intelligence.html. Luettu: 25.3.2017.

Moon, P. & Wozniak, S. 2007. Three minutes with Steve Wozniak. PC World 19.7.2007. Luettavissa: http://abcnews.go.com/Technology/PCWorld/story?id=3396207. Luettu: 6.3.2017

Moravec, H. 1988. Mind Children: The Future of Robot and Human Intelligence. Harvard University Press. ISBN 0-674-57618-7.

Neittaanmäki, P. & Kinnunen, P. 2016. Työttömyys IT-alalla koko Suomessa ja maakunnissa 5/2011–9/2016. Informaatioteknologian tiedekunta. Jyväskylän yliopisto. Luettavissa: https://www.jyu.fi/it/tutkimus/muistiot/ITtyottomat_1116.pdf. Luettu: 10.4.2017.

Niu, J., Tang, W., Xu, F., Zhou, X. & Song, Y. 2016. Global research on artificial intelligence from 1990–2014: Spatially-explicit bibliometric analysis. ISPRS International Journal on Geo-Information. Luettavissa: http://www.mdpi.com/2220-9964/5/5/66. Luettu: 2.5.2017.

Ogheneovo, E. E. 2014. On the relationship between software complexity and maintenance costs. Journal of Computer and Communications. Scientific Research Publishing (SCIRP). Wuhan. Luettavissa: https://www.scirp.org/journal/PaperInformation.aspx?PaperID=51631. Luettu: 15.4.2017.

Oikotie palkkavertailu 2017. Keskimääräisten kuukausipalkkatietojen tietokanta. Luettavissa: https://tyopaikat.oikotie.fi/palkkavertailu/. Luettu: 2.3.2017.

Pearl, R. 25.9.2015. Toimitusjohtaja, Permanente Medical Group. Consumer vs. patient – A false dichotomy. Seminaariluento Stanford Medicine X -seminaarissa. Stanford, Kalifornia. Katsottavissa: https://www.youtube.com/watch?v=SzE9k3Verrg. Katsottu: 13.3.2017.

Rahman, M. 2015. Criticism on scientific management: literature review. University of Dhaka. Luettavissa: https://www.slideshare.net/mpeash1/criticism-on-scientific-management-literature-review-based-article. Luettu: 26.3.2017.

Ritzer, G. 1983. The McDonaldization of society. Journal of American Culture, 6, 1, s. 100-107. Luettavissa: https://www.researchgate.net/publication/227981832_The_McDonaldization_of_Society. Luettu: 3.3.2017.

Rosenberger, J. 2012. P vs. NP Poll results. Communications of the ACM, 55, 5, s. 10. Luettavissa: http://mags.acm.org/communications/201205?pg=12#pg12. Luettu 10.3.2017.

Russel, S. J., Norvig, P. 1995. Artificial intelligence : A modern approach. Prentice-Hall. New Jersey. ISBN 0-13-103805-2.

Sandberg, J. 2008. ERP systems – fully integrated solution or a transactional platform? Luettavissa: http://www.diva-portal.org/smash/get/diva2:141977/FULLTEXT02. Luettu: 10.3.2016.

Shin, H., Roberts, K., Lu L.; Demner-Fushman D., Yao, J. & Summers R. M. 2016. Learning to read chest X-rays: recurrent neural cascade model for automated image annotation. National Institutes of Health. Bethesda, Maryland. Luettavissa: https://arxiv.org/abs/1603.08486. Luettu: 12.3.2017.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L, van den Driessche, G., Schrittweiser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T. & Hassabis, D. 2016. Mastering the Game of Go with Deep Neural Networks and Tree Search. Nature, 529, s. 484-489. doi:10.1038/nature16961

StackOverflow 2016. Developer survey results 2016. Luettavissa: http://stackoverflow.com/insights/survey/2016. Luettu 10.3.2016.

Tamminen, L. & Pesälä, J. 2016. Olipa kerran työ – ihmisyys, hyvinvointivaltio ja yhteiskuntasopimus vähenevän työn digitaaliyhteiskunnassa. IVA Kustannus. ISBN 978-952-93-8289-7

Thórisson, K. R., Bieger, J., Thorarensen, T., Sigurðardóttir, B. & Steunebrink, R. 2016. Why artificial intelligence needs a task theory – and what it might look like. Luettavissa: https://arxiv.org/abs/1604.04660v2. Luettu: 5.3.2017.

Liitteet

Liite 1. Taulukko: mallin arviointiin käytetyt luvut ja tulokset

Komponenttinen kyky	DI	C	B	Työnimike	Keskipalkka	Yleisyys	Monimutkaisuus	Kompleksisuus	Kaoottisuus	Markkinapotentiaali
Datan vastaanottaminen	−0,8	2	2,78	Tallentaja	1 805 €	4	2	0	0	19 400,18
Tulosten ilmoittaminen	1,1	3	1,85	Toimistoharjoittelija	1 102 €	5	3	0	0	12 933,84
Yksinkertaisen kielen tuottaminen	2,5	6	3,47	Kassamyyjä	1 675 €	5	5	1	0	24 236,91
Strukturoitu tieto	4,7	9	4,26	Lakiasiantuntija	3 197 €	3	5	4	0	29 702,69
Visuaalisen informaation tulkinta	9,0	19	10,00	Valvomopäivystäjä	3 390 €	5	9	2	4	69 783,79
Elehdintä	9,5	12	2,47	Asiakaspalvelija	2 224 €	3	7	5	0	17 233,92
Luonnollisen kielen tuottaminen	10,1	17	6,90	Tekninen kirjoittaja	3 310 €	4	10	7	0	48 176,14
Kuvioiden tunnistaminen datasta	12,1	22	9,90	Ketjumyyntipäällikkö	5 611 €	3	9	7	3	69 062,50
Päättely	12,3	15	2,69	Myynti- ja markkinointiassistentti	2 352 €	3	8	3	2	18 742,95
Luokittelu	14,5	19	4,53	Lajittelija*	2 000 €	5	9	4	3	31 622,78
Ohjelmointi	14,6	17	2,39	Ohjelmoija	3 266 €	2	10	7	0	16 694,34
Itseluottamuksen arviointi	16,6	21	4,43	Tutkija (tohtori)	4 923 €	2	8	7	3	30 895,11
Priorisointi	18,6	24	5,39	Projektijohtaja	5 615 €	2	10	8	3	37 633,07
Luonnollisen kielen tulkinta	19,6	23	3,42	Tekstinkäsittelijä (sanelunpurkaja)	2 071 €	4	10	3	5	23 842,96
Motoriset taidot	20,4	25	4,60	Apupoika	3 369 €	3	9	2	7	32 131,68
Semanttinen analyysi	20,4	24	3,56	Hallintoassistentti	2 839 €	3	10	4	5	24 855,93
Tietokoneiden käyttäminen	20,8	24	3,22	Kirjanpitäjä	2 654 €	3	10	4	5	22 466,39
Ympäristön havainnointi	21,3	26	4,67	Laaduntarkkaaja	2 549 €	4	10	2	7	32 557,03
Opettelu	21,8	27	5,16	Tohtorikoulutettava	2 727 €	4	10	7	5	36 026,14
Näytön arviointi	23,0	28	5,04	Kärajätuomari	5 366 €	2	10	6	6	35 157,74
Koneiden ja laitteiden käyttäminen	23,4	28	4,56	Asentaja / tekninen tuki	3 345 €	3	10	4	7	31 788,94
Arvo- ja tunneäly	25,2	29	3,82	HR-päällikkö	4 458 €	2	10	9	5	26 622,88
Tavoitteiden määrittely	26,3	34	7,75	Markkinointijohtaja	7 148 €	2	10	10	7	54 053,25
Tiedon synteesi	26,5	28	1,46	Vanhempi tutkija	4 698 €	1	10	8	5	10 182,85
Tietämys	29,9	32	2,12	Johdon konsultti	6 025 €	1	10	10	6	14 788,89

* https://palkkavertailu.com/palkka/lajittelija