Tiina Heikkinen

DEVOPS JA SEN YHTEENSOVITTAMINEN IT-PALVELUTUOTANTOON

PRO GRADU

JYVÄSKYLÄN YLIOPISTO
INFORMAATIOTEKNOLOGIAN TIEDEKUNTA
2018

TIIVISTELMÄ

Heikkinen, Tiina
DevOps ja sen yhteensovittaminen IT-palvelutuotantoon
Jyväskylä: Jyväskylän yliopisto, 2018, 72 s.
Tietojärjestelmätiede, pro gradu -tutkielma
Ohjaaja: Pulkkinen, Mirja

Asiakkaiden muuttuvat tarpeet ja odotukset sekä kehittynyt teknologia ovat käynnistäneet maailmanlaajuisen digitaalisen murroksen. Digitalisaatio muuttaa yritysten kilpailuasetelmaa ja toimintaympäristöjä. Se edellyttää yrityksiltä innovatiivista toimintaa, uudenlaista osaamista ja omaksumista. Digitalisaatiossa korostuu asiakaslähtöisyyden merkitys. Yritysten on kehitettävä ja tarjottava entistä nopeammin parempia palveluita. Teknologian kehittymisen seurauksena tapahtunut kilpailuympäristön muutos on vaikuttanut myös rahoitusalalla asiakkaiden käytökseen.

Tutkimuksessa käsiteltiin tapausorganisaatiossa pilotoitua DevOps-toimintamallia, jota tutkimalla pyrittiin löytämään haasteita, taustamekanismeja sekä yksittäisiä kehityskohteita IT-palvelutuotannon yhdistämiseen. Tutkimus pohjautui tilanteeseen, jossa toimintamallia pyritään edelleen kehittämään pilotin aikana kehityskohteiden ja palautteiden avulla. Ketterän toimintamallin on tarkoitus laajentua organisaatiossa kattamaan kehittämisen ja tuotannon toimintoja sekä saada organisaation jäsenet käyttämään uutta toimintamallia ja sitoutumaan siihen.

Tässä tutkimuksessa IT-palvelutuotannon ja DevOpsin yhdistämistä lähestyttiin palveluntuotannon näkökulmasta rahoitusalalla. DevOpsin moniulotteisuudesta johtuen ketterän toimintamallin sisältö jää usein epäselväksi. Organisaatioiden on vaikea käyttöönottaa ketterä toimintamalli, koska sen toteuttaminen voidaan tehdä eri tavalla. Tämän tutkimuksen tavoitteena oli muodostaa teoriaan ja empiiriseen osioon perustuen suosituksia eli suunnitteluperiaatteita, jotka tarjoavat lähtökohtia palvelutuotannon huomioimiselle. Palvelutuotannossa halutaan ensisijaisesti varmistaa, että tuotantoympäristö on vakaa ja saatavilla. Tutkimusprosessi toteutettiin konstruktiivisella suunnittelutieteen (Design Science) -menetelmällä, jossa tutkimuksen kohteeksi valittua ongelmaa kehitetään ja jalostetaan iteratiivisesti. Tämän tutkimuksen ratkaisuna toimii tutkimuksen tuloksena syntyneet suunnitteluperiaatteet, jotka keräävät yhteen tutkimuksesta nousseet tärkeimmät havainnot. Suunnitteluperiaatteita voidaan käyttää tukemaan ja kehittämään DevOps-toimintamallia, jossa eri tekijät huomioimalla organisaatio voi saada IT-palvelutuotannon ja DevOpsin yhteensopivaksi.

Asiasanat: IT-palvelunhallinta, ITSM, IT-palvelutuotanto, ITIL, perinteinen ja ketterä IT, bimodaalinen IT, kahden nopeuden IT, DevOps

ABSTRACT

Heikkinen, Tiina
DevOps and its integration to IT Service Operation
Jyväskylä: University of Jyväskylä, 2018, 72 pp.
Information Systems, Master's Thesis
Supervisor: Pulkkinen, Mirja

The changing needs and expectations of customers as well as developing technology have triggered a worldwide digital transformation. Digitalization changes the competitive position and operating environments of companies by requiring innovative activities, new skills and user acceptance. Digitalization emphasizes the importance of a customer-driven approach. Service providers are face with a challenge of developing and offering better services at an increasing speed. This change in the competitive position has also affected customer behavior in the financial sector.

This study processed the DevOps model piloted in the case organization. The model was studied to find out the challenges, background mechanisms and also the individual development targets in combining IT service production. The research was based on a situation in which the model is being further developed during the pilot based on recommendations and feedback. DevOps, the agile model, is expected to cover both development and production operations and to engage members of the organization in using and committing to a new approach.

In this study, the integration of IT service production and a DevOps mindset was approached from a perspective of service production in the financial sector. Due to the multidimensional nature of DevOps, the content of the agile model often remains unclear and is at the same time difficult to implement in organizations because it can be done in many ways. The objective of this study was to provide recommendations based on theory and an empirical section providing the basis for considering the IT service production. The purpose of IT service production is to ensure the production environment's stability, availability and reliability.

The research process was carried out with the constructive design science (DS) method, where problem and solution were studied and processed iteratively. The results of this study include the design principles bringing together the most important observations related to recommendations. The recommendations can be used to support and develop the DevOps model. By considering the various factors of the model allows the organization to make its IT service operation and DevOps compatible.

Keywords: IT Service Management, ITSM, IT Service Operation, ITIL, traditional and agile IT, Bimodal IT, Two-Speed IT, Double Speed IT, DevOps

KUVIOT

TAULUKOT

SISÄLLYS

1 JOHDANTO

Viimeisen vuosikymmenen aikana tietotekniikka (IT, informaatioteknologia) on laajentunut kaikkialle ja monien organisaatioiden IT-toiminnot ovat saavuttaneet tärkeän ja merkittävän roolin. Kehityksestä huolimatta IT-toimintojen alkuperäiset vaatimukset, jotka keskittyvät tehokkaiden ja luotettavien sekä skaalattavissa olevien ja turvallisten IT-palvelujen tuottamiseen, eivät ole muuttuneet. IT-hallinnan on erityisesti viime vuosina ollut yhä vaikeampaa saavuttaa nämä tavoitteet, koska tietotekniikan laaja-alaisuus on ulottunut organisaation rajojen yli ihmisten päivittäiseen arkeen. Haffken (2017a) mukaan tietotekniikan yleistyminen on vaikeuttanut luotettavien järjestelmien toimittamista, jotka koskettavat käyttäjiä, laitteita, liiketoimintaprosesseja ja organisaation toimintoja. Lisäksi globaalit tietoturvauhat heikentävät entisestään järjestelmien luotettavuutta. Vaikka IT-toiminto suoriutuu hyvin tavoitteidensa suhteen, yritysjärjestelmien toimintahäiriötä esiintyy säännöllisesti ja nämä voivat aiheuttaa suurta mainevahinkoa organisaatioille. Yritysten järjestelmien käyttökatkot tai arkaluonteisten tietojen vuotaminen huomataan kaikilla organisaation tasoilla, myös julkisesti. (Haffke, 2017a.)

Teknologian kehittymisen seurauksena tapahtunut kilpailuympäristön muutos on vaikuttanut rahoitusalalla asiakkaiden käytökseen. Suomen Pankin (2016) ja Finanssivalvonnan (2017) mukaan pankkien verkkopalveluiden kehittyminen ja uudet asiointitavat ovat johtaneet muutoksiin konttoriverkostossa. Useammat palvelut ovat saatavilla internetissä ja konttorissa asioiminen on vähentynyt. Digitalisaatiossa keskeistä ovat myös uudet, nuoret asiakkaat, jotka ovat tottuneet hoitamaan asioita internetin ja mobiililaitteiden kautta. Tulevaisuudessa helppous ja vaivattomuus ovat palveluilta vaadittuja ominaisuuksia. (Suomen Pankki 2016; Finanssivalvonta 2017.) Rahoitustoimialalla muuttuvat kilpailuasetelmat, markkinatarpeet sekä asiakkaiden odotukset luovat painetta IT:n hallintaan ja tehokkaiden käytäntöjen etsimiselle. Yritykset automatisoivat tietotekniikkaa, mutta toisaalta on varmistettava, että ydinjärjestelmät toimivat vakaasti ja virheettömästi.

Teknologian muutos ja innovointi sekä kuluttajien nopea digitaalisten tuotteiden omaksuminen ja käyttöönotto ovat lisänneet digitaalisten tuotteiden

ja palveluiden kysyntää. Karunakaran (2013) ja Virmani (2015) toteavat, että ohjelmistokehittämisen alueella toimintatapojen on oltava ketteriä, laadukkaita, tehokkaita ja nopeasyklisiä, jotta kyetään vastaamaan nopeasti asiakkaiden tarpeisiin ja luomaan liiketoiminnalle strategista lisäarvoa. DevOps, ketterä toimintamalli, on noussut yhdeksi uudeksi ilmiöksi organisaatioiden IT:ssä (Virmani, 2015). DevOps korostaa ketterän ohjelmistokehityksen filosofian mukaisia nopeita kehitys- ja julkaisujaksoja, ja sen avulla parannetaan työn sujuvuutta sekä tehokkuutta automatisoinnilla. Ebert, Gallardo, Hernantes ja Serrano (2016) toteavat, että ketterä lähestymistapa soveltuu ylläpidon (tuotannon) ja kehityksen yhteistyöhön koko järjestelmän tai palvelun elinkaaren ajaksi. DevOpsin avulla pyritään parantamaan yhteistyötä ja keskinäistä vuorovaikutusta kehityksen ja tuotantotoimintojen välillä. Tämä asettaa myös uudenlaisia haasteita ja vaatimuksia IT-palvelutuotannolle. Ebertin ym. (2016) mukaan DevOpsilla on käytännössä vaikutusta organisaation toimintakulttuuriin, prosesseihin, teknologioihin, organisaatiorakenteisiin ja palveluihin, joita käytetään kehityksessä ja käyttöönottoprosessissa. Ohjelmistoteollisuudessa DevOps syntyi painottamaan kehittämisen ja käyttöönoton yhteistyön tärkeyttä. DevOps-käsitteen moniulotteisuudesta johtuen sen sisältö jää usein epäselväksi ja samalla vaikeaksi käyttöönotettavaksi organisaatioissa, koska toimintamallin toteuttaminen on mahdollista tehdä monella eri tavalla. (Ebert ym., 2016.)

DevOpsia voidaan soveltaa hyvin erilaisiin toimitusmalleihin, mutta mallit on räätälöitävä ympäristöön ja tuotearkkitehtuuriin sopiviksi. Kaikki tuotteet tai palvelut eivät tee DevOpsin kaltaista jatkuvaa toimitusta helpoksi tai edes edistä jatkuvaa toimitusta. Ebertin (2016) mukaan pitkään kehitettyjen monoliittisten järjestelmien siirtäminen jatkuvan julkaisun piiriin voi olla vaikeaa. Ketterän DevOpsin soveltaminen turvallisuuskriittisten järjestelmien osalta voi olla mahdotonta tai jatkuvan julkaisun toteuttaminen on haasteellista toteuttaa. Erittäin turvallisen toimituksen lisäksi tämänkaltaiset toimitusmallit tarvitsevat erityisiä arkkitehtuureja ja laitteistoja. (Ebert ym., 2016.) Keskeisintä DevOpsin omaksumisessa ja soveltamisessa käytäntöön on, että ohjelmistojen kehittämistä ja käyttöönottoa ei tarkastella enää erillään, vaan ne toimivat keskenään yhteistyössä.

Tutkimuksen tavoitteena on esitellä DevOps-toimintamallia IT-palvelutuotannon näkökulmasta ja kuvata yhdistämisen hyödyt ja haasteet. Tutkimus esittelee, mitä DevOps-toimintamallin käyttöönotto edellyttää palvelutuotannolta ja mitkä ovat DevOps-toimintamallin käyttöönoton vaikutukset IT-palvelutuotantoon. Tämän tutkimuksen tarkoituksena on selvittää DevOps-toimintamallin yhdistäminen IT-palvelutuotantoon ja sen noudattamiin käytänteisiin. Tämän tutkimiseksi tutkimusongelma voidaan esittää tutkimuskysymyksillä seuraavasti:

1. Mitkä ovat DevOpsin ja IT-palvelutuotannon yhtäläisyydet ja erot sekä yhdistämisen haasteet?
2. Minkälaisia valmiita ratkaisumalleja löytyy kirjallisuudesta DevOpsin ja IT-palvelutuotannon yhdistämisestä?

3. Minkälaisia suunnitteluperiaatteita voidaan soveltaa ja hyödyntää ketterän DevOpsin ja IT-palvelutuotannon yhdistämisessä?

Tutkimuskysymyksiin vastataan perehtymällä ensin erikseen IT-palvelutuotannon määritelmään ja DevOpsiin sekä perinteiseen ja ketterään IT:hen. Näiden pohjalta pohditaan ja analysoidaan, mitä palvelutuotanto vaatii DevOpsilta. Tutkimuksessa pyritään tuomaan esille tekijöitä, jotka vaikuttavat IT-palvelutuotannon ja DevOpsin yhdistämiseen, sekä luomaan suunnitteluperiaatteita vastaavan ratkaisun tuottamiseksi organisaatioissa.

Teoreettisina lähtökohtina voidaan pitää yleisiä tutkimuskirjallisuuteen perustuvia käsitteitä. Tässä tutkielmassa käsitellään perinteistä ja ketterää IT:tä, jotta tunnistetaan erilaisia tapoja käyttää ja omaksua ketteriä toimintatapoja osana päivittäistä toimintaa perinteisiä tietojärjestelmiä unohtamatta. Tehokkaan tuen saamiseksi, monet yritykset ovat enenevissä määrin alkaneet käyttää bimodaalisuutta, jota kutsutaan myös bimodaaliseksi IT:ksi (engl. Bimodal-IT) tai kahden nopeuden IT:n toimintamallia (engl. two-speed IT). Lähestymistapana nämä mallit ratkaisevat IT:n ristiriitaisia tavoitteita. Näiden käsitteiden lisäksi avataan IT-palvelunhallinnan ja DevOpsin käsitteitä, jotka liittyvät käsiteltävään tutkimusalueeseen.

Horlachin, Drewsin ja Schirmerin (2017a) tutkimusten mukaan digitaalinen muutos on joissakin tapauksissa johtanut perinteisissä organisaatioissa kahteen erilaiseen IT:n nopeuteen. Digitaalisten innovaatioiden toteuttamiseksi on luotu nopea asiakaslähtöinen ja liiketoimintalähtöinen IT-organisaatio, joka reagoi nopeasti asiakkaiden muuttuviin tarpeisiin. Lisäksi organisaatiot harjoittavat perinteistä IT:tä vakiintuneessa infrastruktuurissa ja organisaatiossa. Tämä osa IT-organisaatiota työskentelee pidemmissä sykleissä ja toimii hitaammin kuin ketterä IT, koska sen on hoidettava suuria taustajärjestelmiä (esimerkiksi pankkien ja verottajan ydinjärjestelmät), joita ei voi muuttaa tai muuttaa helposti ketteräksi. Horlach ym. (2017a) toteavat, että eri nopeuksien lisäksi yritykset voivat toimia erilaisilla organisaatiorakenteilla ja menetelmillä. Siksi monet yritykset perustavat bimodaalisen IT-organisaation, jossa on erilaisia hallinnointimekanismeja, prosesseja ja organisaatiorakenteita. Tässä tutkimuksessa käytetään bimodaalisen IT:n käsitettä kahden nopeuden IT:n sijaan, koska bimodaalisessa IT:ssä ei viitata pelkästään nopeuteen vaan myös eri arkkitehtuureihin, prosesseihin ja organisaatioon. (Horlach ym., 2017a.) Bimodal IT on Gartnerin kehittämä konsepti, jonka avulla IT-organisaatiot voivat tukea sekä perinteisiä että ketteriä IT-ratkaisujen toimituksia ja toimintaa. Perinteisen IT:n siirtäminen ketteräksi edellyttää vaiheittaista muutosta, joka on tehtävä riskejä lieventäen ja johtaen kohti ketterää ja DevOps-käytäntöä ilman suuria askeleita. Agile- ja DevOps-käytäntöihin siirtyminen on tapahduttava vähitellen häiritsemättä järjestelmiä tai palveluita. Tällä pyritään välttämään tuotannon seisokkeja, mutta samalla tuottamaan tuottoja.

IT-palvelunhallinta ja ITIL ovat edelleen yleisimpiä ja käytetyimpiä liiketoimintaprosessien käytäntöjä, jotka tukevat IT-toimintoja. Käytännössä IT-palvelunhallinta ja ITIL sisältävät monia organisatorisia kykyjä, joita IT-toiminnot tarvitsevat myös DevOps-tyyppisen työvirran tukemiseksi. ITIL on

kehitetty lähes täydelliseksi lähestymistavaksi IT-toimintojen hallinnointiin lukuun ottamatta projektimenetelmiä tai järjestelmäarkkitehtuuria. ITIL ei suoranaisesti vastusta ketteryyttä tai DevOpsia. Palvelusuunnittelu tukee iteratiivista ja inkrementaalista suunnittelua. Tällä tarkoitetaan, että suunnittelua ja toteutusta tehdään pienissä osissa prosessia toistaen. Palvelustrategia korostaa jatkuvan palautteen tarvetta IT-palvelun elinkaaren vaiheissa (strategia, suunnittelu, transitio, tuotanto ja jatkuva parantaminen). Vaikka IT-palvelunhallinnan, ITIL:in ja DevOpsin välillä on eroavaisuuksia, molempia kuitenkin tarvitaan ja näiden ei tarvitse olla toisiaan poissulkevia tai kilpailevia vaihtoehtoja. Molemmat tulee nähdä toisiaan täydentävinä. Tänä päivänä IT-toimintojen on kyettävä toimimaan älykkäämmin ja nopeammin, mutta tarvitsemme edelleen prosesseja ja valvontaa. DevOpsiin ja IT-palvelunhallintaan on usein liitetty väärinkäsityksiä esimerkiksi, että DevOps korvaa IT-palvelunhallinnan, koska ei ole tarvetta palveluille ja operoinnille. Palvelunhallinnan keskeisiä osa-alueita kuitenkin tarvitaan IT-palveluiden hallintaan ja hallinnoimiseksi. Kyseiset osa-alueet ovat tärkeitä liiketoiminnalle, ja niitä ei voida sivuuttaa.

Tämän tutkimuksen luvuissa kaksi, kolme ja neljä luodaan käsitteellis-teoreettinen perusta ja esitellään tutkimuksessa käytettävät teoreettiset käsitteet IT-palvelunhallinta, kahden nopeuden IT ja DevOps. Luvussa viisi esitellään teoriaosuuteen liittyviä tuloksia, jotka käsittelevät DevOpsin ja IT-palvelunhallinnan yhdistämistä sekä kehityksen ja tuotannon välistä yhteistyötä. Luvussa kuusi kuvataan tarkemmin tutkimusmenetelmät ja niiden käyttäminen tutkimuksessa. Luvussa seitsemän esitellään ratkaisun arviointi ja tulokset sekä syntyneet suunnitteluperiaatteet. Toiseksi viimeisessä luvussa esitellään pohdintaa ja viimeisessä luvussa kuvataan johtopäätökset, joissa tuodaan esille empiirisessä tutkimuksessa tehdyt selvitykset ja tulosten merkitys.

2 IT-palvelunhallinta

Tietotekniikan (IT, informaatioteknologia) hallinta sisältää tänä päivänä paljon muutakin kuin IT-teknologiaa. IT-palvelunhallinta on erilainen näkökulma IT:n hallintaan. IT-teknologian johtamisen sijaan IT-palvelunhallinta sisältää tietotekniikan organisoinnin osana palveluja, jotka ovat linjassa liiketoiminnan tarpeiden kanssa. Orandin (2013) mukaan tämän päivän IT-organisaatio on kriittinen tekijä lähes kaikilla liiketoiminnan osa-alueilla ja IT:llä on myös kehittyvä rooli. Teknologian kehittyminen ja käytön lisääntyminen liiketoiminnassa on johtanut siihen, että IT kamppailee uusien vaatimusten kanssa. Liiketoiminta haluaa palveluita, jonka teknologia mahdollistaa ja jättää teknologiaosaamisen IT:lle saavuttaakseen liiketoimintaansa haluttuja tuloksia. (Orand, 2013, 31-32.)

Jotta voidaan ymmärtää, mitä palvelunhallinta on ja miksi se on yrityksille tärkeä, on ymmärrettävä ensin, mitä palvelut tarkoittavat ja miten palvelunhallinta voi auttaa palveluntarjoajia toimittamaan ja johtamaan palveluja. Palvelu tarkoittaa arvon tuottamista asiakkaille. ITIL:in määritelmän mukaan tämä toteutetaan auttamalla asiakasta saavuttamaan tuloksia ilman erityisiä kustannuksia ja riskejä. (Griffiths, Lawes, Brewster & Sandbury, 2016.) Palvelut eroavat muista toimituksista tai tuotoksista, koska palvelut vaikuttavat eri tavoin. Palveluilla on monenlaisia muotoja, mutta niillä kaikilla on samat ominaispiirteet. Ne ovat osaksi aineettomia, liittyvät asiakkaiden tuloksiin ja niitä kulutetaan samanaikaisesti, kun niitä tuotetaan. Jos palvelua ei käytetä, ne menettävät arvonsa. (Orand, 2013, 39.)

IT-palvelunhallinta (ITSM, IT Service Management) on kokoelma jaettuja vastuualueita sekä toisiinsa liittyviä tieteenaloja ja prosesseja, joiden avulla organisaatio kykenee mittaamaan, kontrolloimaan ja viime kädessä hallitsemaan IT-infrastruktuuria (Orand, 2013, 40). Lisäksi IT-palvelunhallinta toimittaa korkealaatuisia ja kustannustehokkaita palveluja lyhyen ja pitkän aikavälin liiketoiminnan vaatimusten täyttämiseksi. Palvelujen hallinta on määritelty joukoksi erikoistuneita organisationaalisia kyvyyksiä, joilla tarjotaan asiakkaille arvoa palveluiden muodossa (Macintyre, Parry & Angelis, 2011, 92). Palvelunhallinta on seurausta organisaation kykyjen ja voimavarojen kohdentamisesta palvelu-

jen arvon tuottamiseksi. Tämän halutun tuloksen pitäisi vastata liiketoiminnan tarpeisiin.

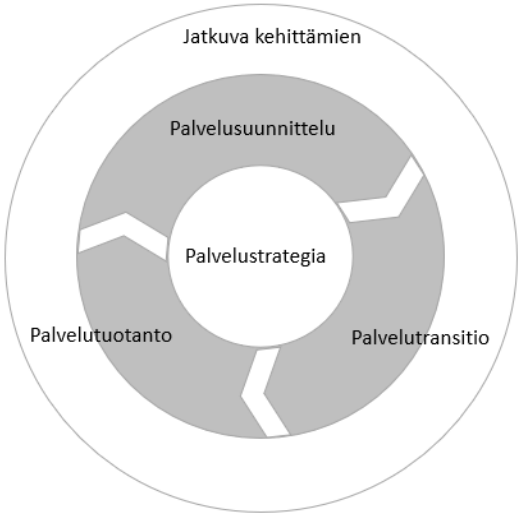
IT-palvelunhallinnan tarkoituksena on tuottaa liiketoiminnan vaatimuksiin erilaisia ratkaisuja, kehittää IT-toimintojen laatua ja parantaa palvelujen kustannustehokkuutta. IT-palvelunhallinnan taustalla on palvelukeskeinen ajattelutapa, joka perustuu IT-palveluiden liiketoimintakriittisyyteen ja kasvaviin laatu- ja tehokkuusvaatimuksiin. (Agutter, 2017, 95.) IT-palvelunhallinnassa yhdistetään liiketoiminnan tarvitsemia resursseja, prosesseja ja teknologiaa oikeassa suhteessa vähentäen palveluiden kustannuksia, mutta samalla ylläpitäen palvelun tasoa. (Iden & Eikebrokk, 2013.) IT-palvelunhallintaa kuvaillaan prosessiorientoituneeksi ja se keskittyy prosessien parantamiseen kuten TQM-laatujohtamisen malli (engl. Total Quality Management), Six Sigma-laatujohtamisen työväline, BPM-liiketoimintaprosessien hallinta (engl. Business Process Management) ja CMMI-tuotekehityksen kypsyysmalli (engl. Capability Maturity Model Integration). (Marrone, Gacenga, Cater-Steel & Kolbe, 2014.) IT-palvelunhallinnassa voidaan hyödyntää erilaisia standardeja ja viitekehyksiä kuten COBIT tai IT4IT, mutta yleisimmin ITSM rakentuu ITIL-viitekehyksen varaan. Organisaatiot voivat hyödyntää ITIL:iä parhaiden käytäntöjen soveltamisessa. Tässä tutkimuksessa keskitytään ensisijaisesti ITIL-malliin ja esitellään sen sisältämät elinkaaren vaiheet ja prosessit.

2.1 ITIL-malli

ITIL (Information Technology Infrastructure Library) on laajasti hyväksytty lähestymistapa IT-palveluiden hallintaan (Agutter, 2017, 96; Orand, 2013, 42). ITIL on Britannian OGC:n (Office of Government Commerce) rekisteröimä tavaramerkki ja malli kehitettiin 1980-luvun puolivälissä alun perin yhteistyössä alan asiantuntijoiden, konsulttien ja kouluttajien kanssa auttamaan organisaatioita IT:n käytössä ja hallinnassa sekä kehittämään ja yhdenmukaistamaan käytäntöjä (Daniels & Davis, 2016; Orand, 2013, 43). ITIL-mallin parhaita käytäntöjä selvittämällä tutkitaan, miten ne edistävät IT-palvelunhallintaa. Parhaat käytännöt ovat testattuja malleja tai prosesseja, joita on käytetty onnistuneesti useissa organisaatioissa ja ne ovat erityisesti tarkoitettu sovellettavaksi IT-palvelunhallinnassa. (Agutter, 2012; BS, 2014; Davies, 2016; Iden & Eikebrokk, 2013; Orand, 2013, 43.) ITIL voi auttaa yksilöitä ja organisaatioita käyttämään tietotekniikkaa muun muassa liiketoiminnan muutoksissa ja kasvun toteuttamiseen (Agutter, 2017, 96). ITIL-viitekehyksen tavoitteena on antaa ohjeita sovellettavaksi kaikentyyppisille organisaatioille, jotka tarjoavat tietotekniikkapalveluja yrityksille riippumatta niiden koosta, monimutkaisuudesta, kaupallisista palveluntarjoajista tai liiketoiminnan sisäisistä liiketoiminta-alueista (Griffiths ym., 2016; Marrone ym., 2014). Griffithsin ym. (2016) mukaan viitekehyksen ei pitäisi olla byrokraattinen tai hankala, jos sitä käytetään järkevästi ja tunnistetaan yrityksen erityiset tarpeet. ITIL:in geneerinen luonne on sen sekä vahvuus että heikkous. Koska ITIL on geneerinen, sitä voidaan soveltaa eri organisaa-

tiokoolle ja eri markkinasektoriin riippumatta onko palveluntarjoaja liiketoiminnan tai kaupallisen yrityksen sisällä. Organisaatioiden on kuitenkin hyväksyttävä ja mukautettava ohjeitaan ITIL:in erityisvaatimuksiin, jotka joissakin tapauksissa edellyttävät huomattavia ponnisteluja ja sitoumuksia. (Griffiths ym., 2016.)

IT-palvelun elinkaari voidaan kuvata ITIL-elinkaarinmallissa kuvattujen vaiheiden ja prosessien kautta. Nämä vaiheet ovat luonteeltaan dynaamisia ja niitä voidaan soveltaa päätöksentekoon. Dynaamisuudella tarkoitetaan sitä, kun keskitytään työtehtävän elinkaaren tiettyyn vaiheeseen, niin voidaan joutua tekemään toiseen vaiheeseen liittyviä päätöksiä. Esimerkiksi julkaisu- ja käyttöönottoprosessin palvelutransitiossa työskentelevän kehittäjän on tehtävä päätöksiä suunnitteluun ennen julkaisun kokoamista. (Ayat, Sharifi, Sahibudin & Ibrahim, 2009; Cruz-Hinojosa & Gutiérrez-de-Mesa, 2016.) ITIL:in sisältämät viisi elinkaaren vaihetta ja keskeistä prosessia ovat palvelustrategia, palvelusuunnittelu, palvelutransitio, palvelutuotanto ja palvelun jatkuva parantaminen (kuvio 1). ITIL suosittelee, että IT-palvelut ovat linjassa yrityksen tarpeiden kanssa ja tukevat sen ydinprosesseja. (Bernard, 2011, 30; Orand, 2013, 26-28.) ITIL tarjoaa organisaatioille ja yksilöille mahdollisuuden käyttää tietotekniikkaa työvälineenä helpottaakseen liiketoiminnan muutosta ja kasvua (Agutter, 2017, 96).



KUVIO 1 Pelkistetty ITIL v3 Elinkaarimallin rakenne

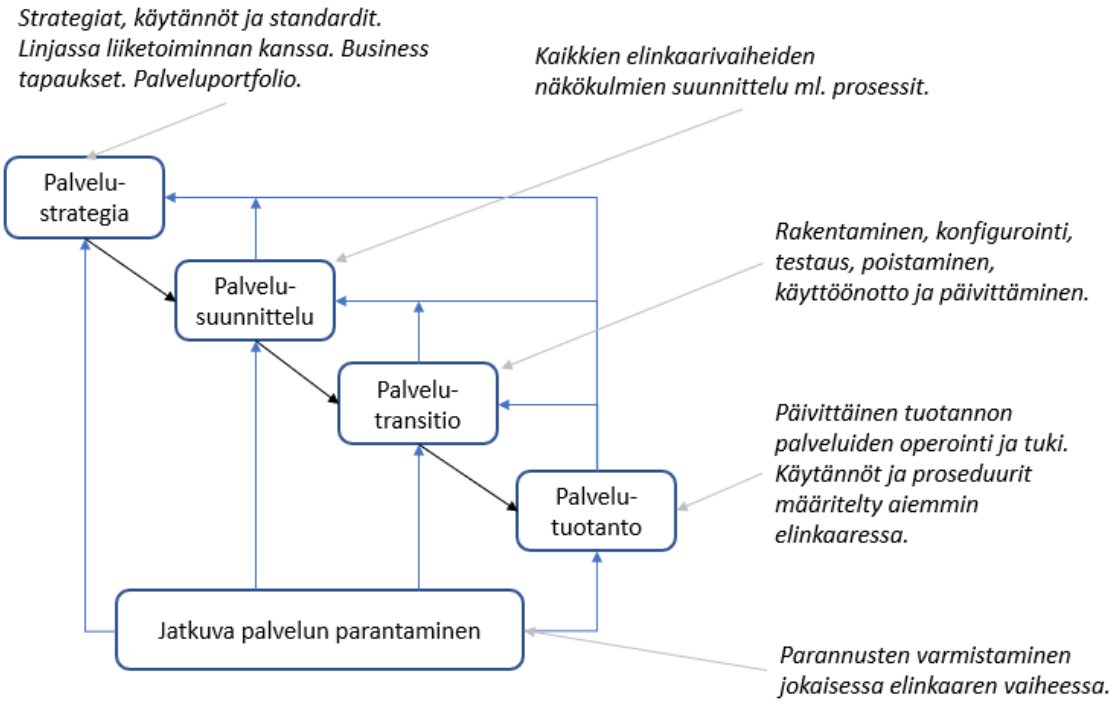
Palvelustrategia toimii keskeisenä osana palvelun elinkaarta. Se ohjaa muita vaiheita ja antaa strategisia ohjeita palvelunhallintaan. Palvelustrategian tavoitteena on löytää tapoja asiakkaiden palvelemiseen. Palvelustrategia kattaa myös taloudellisen näkökulman sekä riskienhallinnan. (Griffiths ym., 2016; Himi, Bahsani, & Semma, 2011; Nabiollahi & bin Sahibuddin, 2008.)

Arvon luominen ja liiketoiminnan yhdenmukaistaminen ovat palvelusuunnittelun keskeisiä tavoitteita. Se käsittää periaatteet ja menetelmät, joiden avulla liiketoiminnan tavoitteita voidaan muuntaa pitkän aikavälin suunnitelmiin. (Himi ym., 2011.) Palvelusuunnittelun vaihe vastaa uusien palvelujen suunnittelusta tai muutoksista nykyiseen palvelukatalogiin ja palveluiden siir-

tymisestä tuotantoympäristöön. Suunnitteluvaiheessa määritetyt tuotteet ja palvelut toimitetaan tuotantoon ja ne ovat asiakkaiden ja käyttäjien saatavilla. (Griffiths ym., 2016; Nabiollahi & bin Sahibuddin, 2008.)

Palvelustransition vaiheessa keskitytään muutoksiin eli transitioon. Sen tarkoituksena on huolehtia, että palvelut otetaan tuotantokäyttöön hallitusti ja valvotusti. Palvelutuotanto on elinkaaren kriittisin vaihe ja sisältää palvelun tuottamisen. Tässä vaiheessa tuotetaan varsinaista arvoa asiakkaalle ja liiketoiminnalle. Palvelutuotanto vastaa palveluiden tuottamisesta tehokkaasti, sovituilla palvelutasoilla ja kustannuksilla. (Himi ym., 2011; Nabiollahi & bin Sahibuddin, 2008.) Palvelutuotanto käsittää tapahtuma-, häiriö-, ongelma- ja palvelupyyntöprosessit sekä pääsynhallinnan prosessit (Griffiths ym., 2016).

Jatkuvan palvelun kehittäminen on jatkuva prosessi, joka tarjoaa ohjeita asiakkaiden arvon luomiseen ja ylläpitämiseen paremman palveluiden suunnittelun ja tuotannon avulla (Himi ym., 2011). Se yhdistää laadunhallinnan, muutoshallinnan ja kyvykkyyden parantamisen periaatteet, käytännöt ja menetelmät. (Griffiths ym., 2016; Nabiollahi & bin Sahibuddin, 2008.) Kuviossa 2 on havainnollistettu Daviensin (2016) mukaan ITIL-elinkaarimallin prosessit. Seuraavissa alaluvuissa esitellään ITIL-elinkaarimallin prosessit tarkemmin.



KUVIO 2 ITIL v3 Elinkaarimallin prosessit (Daviesin, 2016 mukaan)

2.1.1 Palvelustrategia

Palvelustrategian tavoitteena on tarjota parempia palveluja kuin kilpailijat. Se sisältää myös palvelunhallinnan suunnittelun, kehittämisen ja toteuttamisen hyvän hallintotavan perustana sekä osana organisaation strategista omaisuuspohjaa. (Griffiths ym., 2016.) Palvelustrategia tarjoaa ohjeistuksen kokonaisstra-

tegian kehittämiseen IT-palvelunhallintaa varten. Orandin (2013) mukaan tämä tarkoittaa ymmärrystä markkinoista, asiakkaista, kyvykkyyksistä ja resursseista sekä taloudellisista rajoitteista, joiden mukaan palvelut on määriteltävä, toimitettava ja tuettava. Palvelustrategia sisältää myös meneillään olevien IT-palveluiden palveluportfolion hallinnan, taloushallinnan, kysynnän hallinnan ja liiketoimintasuhteiden hallinnan prosessit. Palvelustrategian perimmäinen tarkoitus on kehittää strategia määritellyille markkinoille. (Orand, 2013, 47.)

2.1.2 Palvelusuunnittelu

IT-strategian määrittelyn jälkeen organisaatio käyttää palvelusuunnittelun elinkaarivaihetta uusien palveluiden luomiseen. Tämän jälkeen palvelutransitio siirtää palvelut tuotantoympäristöön. Griffiths ym. (2016) esittää, että palvelumallin tarkoituksena on toteuttaa tarvittavat toimenpiteet varmistaakseen, että uusi palvelu toimii suunnitellulla tavalla ja tuottaa liiketoiminnalle tarvittavat toiminnot ja hyödyt. Tämä periaate on ITIL-lähestymistavan ytimessä ja siksi suurin osa palvelusuunnittelun prosesseista keskittyy toiminnan valvontaan. (Griffiths ym., 2016.) Palvelusuunnittelu tarjoaa ohjeistuksen tasapainottavan suunnittelun periaatteille monenlaisia rajoitteita vastaan. Orandin (2013) mukaan palvelusuunnittelussa käsitellään myös sitä, miten suunnitella palvelu, joka vastaa liiketoiminnan tarpeisiin ja on taloudellisesti perusteltua. Tässä vaiheessa vaatimukset sisällytetään suunnitteludokumentteihin, joiden avulla palvelua tai palvelun muuttamista voidaan kehittää. Palvelusuunnittelu sisältää palvelutasonhallinnan, palveluluettelonhallinnan, saatavuudenhallinnan, tietoturvan hallinnan, toimittajahallinnan, kapasiteetinhallinnan, IT-palveluiden jatkuvuudenhallinnan ja suunnittelun koordinoinnin prosessit. (Orand, 2013, 48-50.)

2.1.3 Palvelutransitio

IT:n kehityksen ja tuotannon välillä on usein kuilu, mikä on johtanut monien uusien tai muuttuneiden palvelujen epäonnistuneisiin toteutuksiin. Griffithsin ym. (2016) mukaan palvelujen siirtymävaiheessa otetaan huomioon toiminnalliset vaatimukset ennen kuin mitään siirretään tuotantoympäristöön, mukaan lukien dokumentointi ja koulutus käyttäjille sekä ylläpitäjille. Palvelutransitio on myös vastuussa sellaisten palvelujen käytöstä poistamisesta, joita ei enää tarvita ja palvelun siirtämisestä yhdeltä palveluntarjoajalta toiselle. (Griffiths ym., 2016.) Palvelutransitio tarjoaa ohjeistuksen palvelun siirtymisestä toimintaan. Palvelutransitio käsittelee kaikkia palvelun edellyttämiä elementtejä. Nämä elementit käsittävät kaikki tekniset ja ei-tekniset palvelut. Kokonaisvaltainen näkymä palvelusta auttaa varmistamaan, että palvelu siirretään tavalla, jota voidaan tukea jatkuvasti mahdollisimman tehokkaasti. Orand (2013) toteaa, että palvelutransitio on kriittinen vaihe palvelun elinkaaressa. Tässä vaiheessa varmistetaan, että liiketoiminnan vaatimukset täyttyvät käyttöönotettujen palveluiden avulla samalla kun minimoidaan liiketoiminnan riski. Palvelutransitio

käsittää transition suunnittelun ja tuen, muutoshallinnan, palveluomaisuuden- ja konfiguraationhallinnan, julkaisun- ja käyttöönotonhallinnan sekä tietämyksenhallinnan ja muutoksen evaluoinnin prosessit. (Orand, 2013, 50.)

2.1.4 Palvelutuotanto

Palvelutuotanto on IT-palvelunhallinnan elinkaaren vaihe, joka vastaa liiketoiminnan tavanomaisesta toiminnasta. Jos palveluita ei hyödynnetä tai niitä ei toimiteta tehokkaasti, ne eivät anna täyttä arvoa riippumatta siitä, miten hyvin ne suunnitellaan. Palvelutuotanto on vastuussa prosessien hyödyntämisestä palveluiden tuottamiseksi käyttäjille ja asiakkaille. Merkittävien mittaustulosten tuottaminen palvelutuotannon avulla muodostaa perustan ja lähtökohdan palvelun parannustoiminnalle. (Griffiths ym., 2016.) Palvelutuotanto tarjoaa ohjeistuksen tehokkaaseen palvelun operointiin. Palvelutuotanto on tärkeää jatkuvan parantamisen kannalta, koska palvelutuotantovaiheessa monitoroidaan ja parannettavia kohteita tunnistetaan palvelun suorituskykyraporttien avulla. Palvelutuotanto käsittää häiriönhallinnan, ongelmanhallinnan, herätteidenhallinnan, palvelupyyntöjen hallinnan ja pääsynhallinnan prosessit. (Orand, 2013, 52.)

2.1.5 Jatkuva kehittäminen

Kun palvelut tai järjestelmät on toimitettu tuotantoympäristöön, on olennaista jatkaa palveluiden parantamista, koska kaikki ympäristön näkökohdat muuttuvat jatkuvasti. Griffiths ym. (2016) korostavat, että jatkuva palvelun parantaminen vastaa siitä, että nämä parannukset tunnistetaan ja toteutetaan. IT-palveluntarjoajan suorituskykyä mitataan jatkuvasti ja parannetaan prosesseja, IT-palveluita ja IT-infrastruktuuria tehokkuuden ja kustannustehokkuuden lisäämiseksi. (Griffiths ym., 2016.) Jatkuva palvelun kehittäminen tarjoaa ohjeistuksen, joka varmistaa, että palvelun parantaminen toteutetaan huolellisesti ja vastaa asiakkaiden tarpeita. Orandin (2013) mukaan jatkuva palvelun kehittäminen määritellään siten, että se on integroitava kaikkiin muihin elinkaaren vaiheisiin ja se kuvaa parannusta jatkuvana aktiviteettinä. Palvelutuotannon suorituskykyraportteihin pohjautuen jatkuva palvelun kehittäminen pyrkii tunnistamaan parannuksia ja dokumentoi suositeltavat parannukset palvelun parantamissuunnitelmaan. Seitsemän askeleen kehittämisprosessia käytetään tarkentamaan ja hallitsemaan parannuskohteita. (Orand, 2013, 52-53.) Parannusprosessiin sisältyvät seuraavat vaiheet: tunnista strategia parantamista varten, määritä mitä mittaat, kerää tietoa, käsittele tietoa, analysoi tietoa, esitä ja käytä tietoa sekä toteuta parannuksia (Himi ym., 2011).

2.2 Yhteenveto

Tässä luvussa kuvattiin IT-palvelunhallinnan (ITSM) käsitettä. IT-palvelunhallinta pohjautuu vahvasti ITIL-malliin, joka käsittää kokoelman parhaita käytäntöjä ja periaatteita IT-palveluiden hallintaan. ITIL muodostuu palvelun viidestä elinkaaren vaiheesta, joita ovat palvelustrategia, palvelusuunnittelu, palvelutransitio, palvelutuotanto ja jatkuva palvelun kehittäminen. ITSM tarjoaa erinomaiset puitteet IT-toiminnan aktiviteetteihin ja eri tahojen väliseen vuorovaikutukseen.

Toimialat voivat soveltaa IT-palvelunhallinnan parhaita käytäntöjä optimoidakseen IT-palveluja. ITSM keskittyy tarjoamaan prosesseja, mittareita ja ohjeistusta IT-palveluprosessien arvioinnin, suunnittelun ja toteuttamisen mahdollistamiseksi ja hallinnoimiseksi sekä taktisten ja strategisten IT-voimavarojen käytön optimoimiseksi (Galup, Dattero, Quan & Conger, 2009). Tässä tutkimuksessa painotetaan IT-palvelutuotannon näkökulmaa. IT-toiminnoilla on keskeinen rooli liiketoimintojen tukemisessa ja liiketoiminnan vaatimusten täyttämisessä. Palvelutuotannossa korostuvat palveluiden ja järjestelmien häiriöttömät ja laadukkaat käyttöönotot sekä tuotantoympäristön toimintavarmuuden varmistaminen. Kehittäminen vastaa laadukkaista toimituksista ja lopputuotoksista vaarantamatta tuotannon toimintavarmuutta. Laadukas kehittäminen, tuotantokäytön aloittaminen ja palvelutuotannon ylläpitäminen eivät tapahdu yksittäisissä tuotantoon luovutuksissa vaan kehittäminen ja tuotanto yhdessä varmistavat tavoitteeseen pääsemisen koko kehittämisen aikana. Tiedonsiirron jakaminen varmistetaan tarkoituksenmukaisella yhdessä sovitulla mallilla. Tuotantoon siirtymisen jälkeen pyritään varmistamaan, että palvelutuotannolla on hyvät mahdollisuudet toimia häiriöttä.

Palvelutuotannolla on tärkeä rooli myös jatkuvan parantamisen kannalta. Palvelutasonhallinnalla varmistetaan, että palvelutasot vastaavat liiketoiminnan tarpeita ja tuotettu palveluiden laatu on linjassa palvelutasosopimusten kanssa. Mittaaminen on tärkeää, jotta palveluiden suorituskykyä voidaan seurata suhteessa liiketoiminnan tarpeisiin ja sopimusvelvoitteisiin. Palvelutasonhallinnassa käynnistetään tarvittaessa palveluiden parannustoimenpiteitä ja muutostarpeita.

Seuraavassa luvussa tarkastellaan bimodaalista IT:tä sekä avataan perinteistä ja ketterää IT:tä. IT-palvelunhallinnan tulee reagoida liiketoiminnan luonteen mukaiseen toimintaan ja tarpeisiin. Samaan aikaan IT-palvelunhallinnan tulee turvata myös perinteisen IT:n jatkuvuus ja saatavuus, mutta myös vastata nopeasti uusiin liiketoiminnan vaatimuksiin nopeiden toimitusten, tehostamisen ja tehokkuuden osalta.

3 Bimodaalinen IT

Digitaalinen muutos on haastanut perinteisen IT-toimintojen odotukset, koska organisaatiot vaativat enemmän IT-toimitusten nopeutta, ketteryyttä ja näiden hyödyntämistä digitaalisessa liiketoimintaympäristössä. Tämän saavuttamiseksi, yritysten täytyy muuttaa nykyisiä IT-organisaatiorakenteitaan ja prosessejaan sekä luoda erillisiä toimintatapoja liiketoimintalähtöiselle ja perinteiselle IT-toimitukselle. Optimaalisen tasapainon saavuttaminen ja IT:n hyödyntäminen koetaan usein haasteellisena, koska samanaikaisesti tuotetaan ketteryyttä ja korkeaa luotettavuutta. Tämä on lisännyt uudelleenorganisoitumista ja muutoshallintaa. Haffke (2017a) ja Horlach sekä Drews, Schirmer ja Böhmann (2017b) toteavat, että yhä suositummaksi lähestymistavaksi ristiriitaisille IT:n innovaatiotavoitteille luotettavuuden ja vakauden osalta on tullut bimodaalinen suunnittelu. Bimodaalinen IT mahdollistaa IT-toiminnon toimivan kahdessa rinnakkaisessa toimintatavassa - perinteisessä ja ketterässä. (Haffke, 2017a; Horlach, Drews, Schirmer & Böhmann, 2017b.) Seuraavassa luvussa esitellään kahden lähestymistavan eli perinteisen ja ketterän IT:n eroja ja ominaispiirteitä.

3.1 Perinteinen ja ketterä IT

Bimodaalinen IT on melko uusi konsepti, jonka tutkimustalo Gartner nimesi vuonna 2014 (Jöhnk, Röglinger, Thimmel & Urbach, 2017; Horlach ym., 2017a). Bimodaalinen IT jakaa IT-toiminnot perinteiseen ja ketterään IT:hen. Jälkimmäistä kutsutaan myös nimellä "digitaalinen IT" (Haffke, Kalgovas & Benlian, 2017b). Perinteinen IT korostaa vakautta (stabiliteettia) ja turvallisuutta sekä jatkuvuutta ja saatavuutta, kun taas ketterä IT keskittyy nopeuteen, innovaatioihin ja uusiin teknologioihin (Haffke ym., 2017b; Horlach ym., 2017b).

Perinteinen malli (kutsutaan kirjallisuuslähteissä moodiksi 1) korostaa peräkkäisiä vaiheita, tarkkuutta ja sisältää pitkän aikavälin suunnitelmia, tavoitteita ja kehitystä vesiputousmenetelmää soveltamalla. Tähän malliin kuuluvat tietojärjestelmät ovat tyypillisesti liiketoiminnan kannalta kriittisiä järjestelmiä

ja liiketoiminnan osallistuminen sovelluksen elinkaareen on yleensä rajoitettua. Näissä ympäristöissä muutosten toteuttaminen voi olla monimutkaista, hidasta ja usein liian kallista (Tate, Eicher, Jagannathan, Lad & Burns, 2016). Lisäksi kehityksen, testauksen ja operoinnin siiloutuminen on yleistä. Moodi 1 vastaa hyvin määritellyillä mittareilla vakauden, tehokkuuden, turvallisuuden ja tarkkuuden varmistamisen operatiivisten riskien minimoimiseksi palveluliiketoiminnassa. (Horlach ym., 2017a.)

Ketterä malli (kutsutaan kirjallisuuslähteissä moodiksi 2) sen sijaan keskittyy ketterään ja nopeaan IT-toimitukseen (Horlach ym., 2017a). Se pyrkii nopeuden ja reagointikyvyn kautta myötävaikuttamaan markkinoiden nopeasti muuttuviin vaatimuksiin käyttämällä uudentyyppisiä menetelmiä ja tekniikoita kuten pilvipohjaisia ympäristöjä ja mikropalveluja (engl. microservices). Asiakastyytyväisyys varmistetaan yksinkertaistamisella, integroinnilla, automatisoinnilla, läpinäkyvyydellä ja nopeammalla reagoinnilla kuitenkaan unohtamatta turvallisuus- ja regulaatiovaatimuksia. (Haffke ym., 2017b; Horlach ym., 2017b.)

Moodissa 1 korostuvat laatu ja virheettömyys. Tässä moodissa on ajatuksena käyttää olemassa olevia menetelmiä ja prosesseja, mutta etsitään koko ajan parempia tapoja toimia. Moodissa 2 haetaan kevyempiä, helpompia ja nopeampia prosesseja, joilla saadaan kustannustehokkuutta ja innovointia sekä sallitaan korkeampi riskinottokyky. (Horlach ym., 2017a.) Näitä ovat esimerkiksi kevyemmät prosessit ja vaatimukset palveluiden transitioon tai tuotantoon valmistautumiseen tai hyödynnetään nopeampia prosesseja palveluoperoinnin osalta esimerkiksi muutoksen hallinnassa. Ketterä malli mahdollistaa nopean kehityksen, testauksen ja operoinnin sekä reagoinnin markkinoiden palautteeseen. Taulukossa 1 on kuvattu tyypillisimmät ominaispiirteet perinteiselle ja ketterälle IT:lle. (Haffke ym., 2017b; Horlach ym., 2017b.)

TAULUKKO 1 Perinteisen ja digitaalisen IT:n piirteet (Horlachin ym., 2017a mukaan)

Perinteinen IT (moodi 1, ydin IT)		**Digitaalinen IT (moodi 2, agile IT)**
Vakaus	Tavoite	Ketteryys & nopeus
IT-lähtöinen	Kulttuuri	Liiketoimintalähtöinen
Etäällä asiakkaasta	Asiakasläheisyys	Lähellä asiakasta
Suorituskyvyn ja turvallisuuden parantaminen	Laukaisija	Lyhyen ajan markkinatrendit
Palveluiden suorituskyky	Arvo	Liiketoiminta, asiakasbrändäys
Turvallisuus & luotettavuus	Palvelun fokus	Innovaatio
Vesiputousmenetelmän filosofia	Lähestymistapa	Iteratiivinen, ketterä kehittäminen
Taustajärjestelmät (systems of record)	Järjestelmät	Asiointiliittymät (systems of engagement)
Hidas	Palvelun toimituksen nopeus	Nopea

Tyypillisesti IT:n järjestelmänä voi olla esimerkiksi pieni pilvipalvelu, jota muutetaan ja kehitetään jatkuvasti ketterästi (Lean-Agile-tavalla) ja tuotanto toimii DevOpsin toimintatapoja noudattaen. Tämäntyyppisten järjestelmien tai palve-

lujen nopeutta ei voida tukea hitailla tai byrokraattisilla prosesseilla, jotka varmistavat jatkuvuuden, mutta tekevät muutosten toteuttamisen hitaaksi. Tämän takia tarvitaan kahden IT-palvelunhallinnan tapaa ja tulevaisuudessa on kyettävä tukemaan DevOpsin tukemia ketteriä palveluita. Bimodaalisuus pyrkii kaventamaan IT-toimitusten ja liiketoiminnan tarpeiden välistä kuilua ja yksi IT-organisaatio voi toimia kahdella nopeudella (Horlach ym., 2017b). Seuraavaksi esitellään bimodaalisen IT:n lähestymistapoja.

3.2 Näkökulmia bimodaaliseen IT:hen

Haffke ym. (2017b) ovat tunnistaneet bimodaaliselle IT-mallille neljä arkkityyppiä eli malliesimerkkiä, joille on ominaista erilaiset struktuurilliset tasot perinteisessä ja ketterässä mallissa. Arkkityypit kuvaavat työnjakoa organisaation tiimien tai osastojen kesken sekä osastojen välistä vuorovaikutusta. Eri tyypeillä on omat heikkoutensa ja vahvuutensa. Seuraavaksi avataan näiden neljän arkkityypin sisältöä ja ominaispiirteitä.

Ensimmäinen arkkityyppi A on projektikohtaisesti toimiva malli (engl. Project-by-Project Bimodal IT). Uuden projektin alkaessa IT-toiminnon on päätettävä, käytetäänkö perinteistä vai ketterää toimintatapaa. (Haffke ym., 2017b.) IT-kehittäjien mukaan ottaminen Agile-vaihtoehdossa voi olla haasteellista, koska esimerkiksi iteratiivinen työskentely voi olla tuntematon käsite etenkin erittäin säännellyillä toimialoilla, joissa on tiukat prosessit ja hallintotapa IT-toteutuksille. Arkkityyppi ”Project-by-Project Bimodal IT” voi olla hyvä valinta yrityksille, jotka ovat haluttomia tekemään suuria muutoksia. Tämä arkkityyppi mahdollistaa vaihtoehtoisen mallin käyttöönottamisen vähitellen perinteisen rinnalle. (Haffke ym., 2017b; Haffke ym., 2017c.) Haffke ym. (2017c) nostavat esille esimerkiksi pankkisektorin, joka historiallisesti on erittäin konservatiivinen, mutta ala on alkanut ottaa kasvavaa riskienottohalukkuutta saavuttaakseen digitalisaation innovaation mahdollisuudet. Projektikohtaisesti toimiva malli on käytetyimpiä lähestymistapoja bimodaalisen IT:n toteuttamiseksi. Haffke (2017a) toteaa, että tietyissä tapauksissa IT-toiminnot ovat kehittäneet ”nopean-polun” lähestymistavaksi, joka noudattaa kevyttä hallinnointimallia ja jonka avulla hankkeet tai projektit voivat ohittaa tiettyjä prosessin vaiheita nopeuden ja ketteryyden saavuttamiseksi. Tätä toimintatapaa ei voida kuitenkaan noudattaa kaikilla hankkeilla sääntelyvaatimuksista tai palvelutason määräyksistä johtuen. (Haffke, 2017a.)

Toinen arkkityyppi B ”Subdivisional Bimodal IT” jakaa IT-toiminnot kahteen erilliseen sisäiseen osa-alueeseen. Näistä toinen toimii perinteisellä ja toinen ketterällä tavalla. (Haffke ym., 2017b.) Usein yritykset erottavat perinteisten IT-palvelujen, -toimintojen ja -tuen toimittamisen tietotekniikka-alan IT-innovaatioihin ja kokeiluihin. Jälkimmäinen edellyttää työntekijöiltä erilaista taitoa ja osaamista toisin kuin perinteisessä IT-yksiköissä. Liiketoiminta hyötyy IT-toiminnon sisäisestä osa-alueesta, koska liiketoiminta voi usein panostaa vähemmän perinteiseen tietotekniikkaan sekä vaatia vastaavasti myös tehokasta

tukea organisaation digitaalisille liiketoiminta-aloitteille. IT-toimintojen erottaminen kahdesta toimintatavasta voi kuitenkin aiheuttaa syvän kulttuurisen jakautumisen ja jännitteet eri tiimien välillä. (Haffke ym., 2017b; Haffke ym., 2017c.)

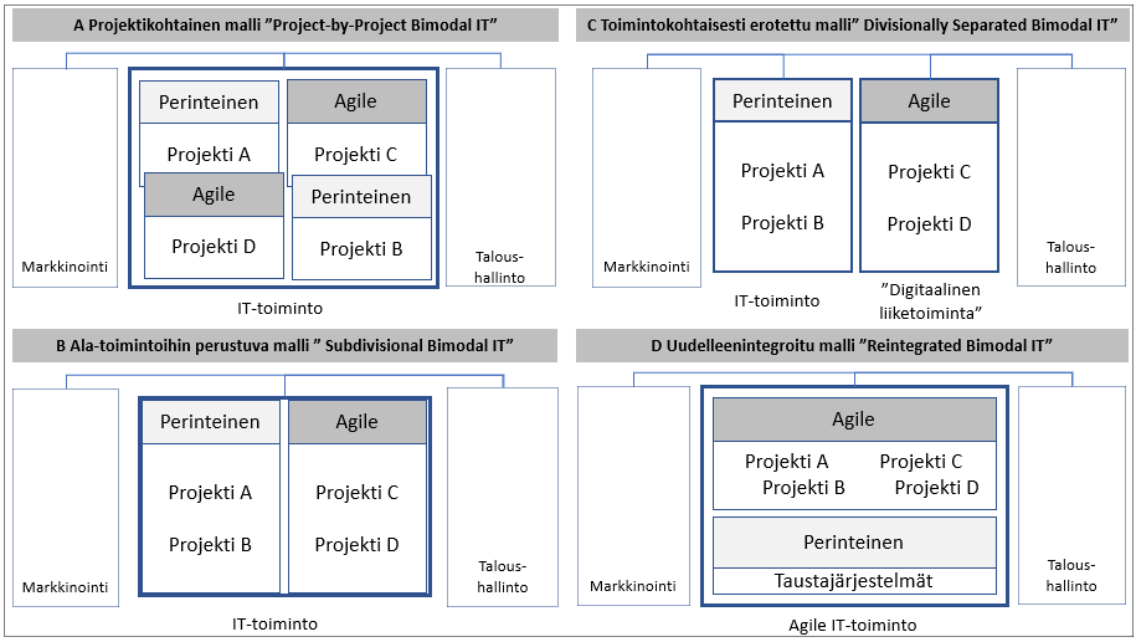
Kolmas arkkityyppi C kuvaa toimintokohtaisesti erillään olevaa funktiota (engl. Divisionally Separated Bimodal IT). Tämä arkkityyppi muodostaa ketterän mallin täysin perinteisen IT-toiminnon ulkopuolella. (Haffke ym., 2017b.) C-arkkityypin hyväksyneissä organisaatioissa ketterää divisioonaa johdetaan usein digitaalisen liiketoiminnan johtajan toimesta (CDO, Chief Digital Officer) ja sitä kutsutaan usein digitaaliseksi liiketoiminnaksi. Tämä bimodaalinen IT-arkkityyppi usein vähentää toimitusjohtajan ja IT-toiminnon tehoa ja vaikutusta. (Haffke ym., 2017b; Haffke ym., 2017c.) Yritykset voivat saavuttaa toimintokohtaisesti erillään olevan bimodaalisen IT:n myös strategisten yritysostojen kautta (Haffke, 2017a).

Neljäs arkkityyppi D edustaa uudelleenintegroitua mallia (engl. Reintegrated Bimodal IT). Tämä bimodaalinen IT-arkkityyppi sallii yrityksen täysin keskittyä digitaaliseen liiketoiminnan muutosprosessiin ja siirtää perinteisiä taustajärjestelmiä (engl. backend, systems of record) ulkoistuskumppaneille tai taustalla toimivaan pienempään osa-alueeseen. (Haffke ym., 2017b.) Tämän arkkityyppisen toiminnon avulla IT-toiminto säilyttää taustalla perinteisen toimintamallin siirtämättä ulkoisia sidosryhmiä yhtenäiseen IT-funktioon. Toisin kuin ”Subdivisional Bimodal IT” (arkkityyppi B), bimodaalisen IT-toiminnon perinteisellä mallilla ei tavallisesti ole suoria suhteita yrityksen liiketoimintaan tai muihin toiminnallisiin yksiköihin. Perinteistä tapaa käytetään pelkästään sisäisiin IT-tukipalveluihin. (Haffke ym., 2017b.)

Neljän IT-arkkityypin vertailu osoittaa, että bimodaalinen IT toimii katalysaattorina muunnettaessa IT-toimintoja perinteiseltä painopisteeltä digitaaliseen innovaatioon keskittyvälle ketterälle ja tutkivalle organisaatiolle (kuvio 3). Jokaisella IT-arkkityypillä on potentiaalisia etuja ja haittoja ja näiden yhteensovittamisessa on huomioitava yrityksen erityiset olosuhteet ja tarpeet. Haffke ym. (2017b) korostavat, että on tärkeää ymmärtää erot neljän eri arkkityypin välillä, jotka liittyvät ensisijaisesti IT:n jakamisen aiheuttamiin sisäisiin häiriöihin, niiden synnyttämän kulttuurisen jakamisen tasoon, IT-resurssien hallintaan ja liiketoimintamallien välisiin yhdenmukaistamismekanismeihin IT-toiminnon sekä kahden IT-toimintatavan välillä. (Haffke ym., 2017b.)

Perinteisesti organisoituneen IT-toiminnon siirtyminen bimodaaliseen malliin on muutosprosessi, joka yleensä sisältää jonkin verran sisäisiä häiriöitä. Haffke ym. (2017b) toteavat, että keskeisimmät häiriöiden syyt liittyvät tyypillisesti johtajuuden muutokseen, uusien hallintamallien ja menetelmien käyttöönottoon sekä IT-henkilöstön tehtävien siirtymiseen. Lisäksi he toteavat, että jotkut organisaatiot tarkoituksellisesti luovat sisäisiä häiriöitä korostaakseen digitaalisen muutoksen tavoitteita, mutta toiset voivat havaita huomattavan häiriön olevan haitallinen. Organisaation sietokyky eli toleranssi sisäisten häiriöiden osalta vaikuttaa bimodaaliseen IT-arkkityyppiin, jota organisaation tulisi käyttää. Erityisesti organisaatiot, joilla on suurempi toleranssi sisäisten häiriöiden

suhteen, pyrkivät valitsemaan toimintokohtaisesti erotetun bimodaalisen IT:n (arkkityyppi C), kun taas vähemmän suvaitsevaiset organisaatiot yleensä hyväksyvät projektikohtaisen lähestymistavan (arkkityyppi A). (Haffke ym., 2017b.)



KUVIO 3 Bimodaalisen IT:n arkkityypit (Haffken ym., 2017b, s. 103 mukaan)

Perinteisten ja ketterien mallien erottaminen voi aiheuttaa merkittäviä kulttuurillisia jännitteitä eri toimintamalleissa työskentelevien välillä. Ketteryys luo kulttuuria, joka kannustaa kokeilemista, etsimistä sekä oikeuttaa epäonnistumisen riskin. Sitä vastoin perinteinen malli ylläpitää kulttuuria, joka korostaa vakauden ylläpitämiä IT-hallinnan tekniikoita ja käyttää perinteisiä muutoshallintaprosesseja. Nämä keskittyvät esimerkiksi kurinalaiseen ja tiukkaan testaukseen ennen tuotantoon julkaisua sen sijaan, että hyväksytään epäonnistumisen mahdollisuus. Haffke ym. (2017b) toteavat, että projektikohtaisessa lähestymistavassa (arkkityyppi A) kulttuurillinen jakautuminen on rajallista, koska mallissa hallinnoidaan ja käytetään samoja resursseja. IT-henkilöstö voi noudattaa erilaisia prosesseja ja työskentelytapoja riippuen projektin käyttöönotosta. Sitä vastoin alatoimintojen tai toimintokohtaisesti erotettujen (arkkityypit B ja C) mallit ovat toisenlaiset, koska molemmille malleille kehittyy erilaiset kulttuuriympäristöt sekä johtamismuodot. Jännitteet voivat näissä malleissa tapahtua kunkin toimintamallin välissä. Haffke ym. (2017b) esittävät tutkimuksessaan, että projektikohtainen lähestyminen (arkkityyppi A) on suositeltava malli yrityksille, jotka haluavat minimoida bimodaalisen IT:n vaikutuksia. Se on suositeltava arkkityyppi myös yrityksille, jotka haluavat minimoida mahdollisen vahingollisen kulttuurillisen jakautumisen ketterässä mallissa toimivan ja perinteisen toimintamallin välillä. (Haffke ym., 2017b.)

IT-resurssien hallinta haastaa perinteisen IT-toiminnon, joka usein joutuu kohtaamaan kilpailevia ja ristiriitaisia prioriteetteja. Bimodaalinen IT tarjoaa myös resursseja, joita on hallittava. Haffken ym. (2017b) mukaan erityisesti pro-

jektikohtaisessa lähestymistavassa (arkkityyppi A) huomiota vaatii IT-resurssien kohdistaminen ja hallinta, koska perinteiset ja ketterät tiimit toimivat rinnakkain ja jäsenten on mahdollisesti vaihdettava eri toimintamalliin ja tiimiin projektista riippumatta. Tämä voi johtaa resurssien osalta ristiriitoihin, joissa henkilöstön jäsenet kokevat vaikeuksia vuorottelevien tehtävien ja tekniikoiden välillä. Resursseihin liittyvät haasteet on kuitenkin helpompi hallita alatoiminnoittain tai toimintokohtaisesti erotettujen mallien avulla (arkkityypit B ja C), koska tiimit ovat pysyvästi sijoitettuna omiin ryhmiin tai ala-toimintoihin. (Haffke ym., 2017b.)

Bimodaalisen IT:n omaksumisessa ja käyttöönotossa yhdenmukaisuus on välttämätöntä liiketoiminnan ja IT-toiminnon välillä sekä perinteisen ja ketterän lähestymistavan välillä jännitteiden hallitsemiseksi. Perinteisen ja ketterän toimintatavan välillä on useita rajapintoja, jotka vaativat strategista ja operationaalista yhdenmukaistamista. Esimerkiksi ketterässä tiimissä olevat voivat joutua työstämään asioita perinteisen tiimiin omistamiin järjestelmiin ja tällöin muutosprosessi kehittämisen näkökulmasta voi olla aikaa vievää. Haffken ym. (2017b) mukaan organisaatiot, jotka ovat haluttomia käsittelemään yhdenmukaistamiseen liittyviä asioita, haluavat mieluummin toteuttaa projektikohtaista (arkkityyppi A) toimintatapaa. Toimintokohtaisesti erotettu lähestymistapa (arkkityyppi C) toisaalta vaatii usein merkittäviä linjaamistoimenpiteitä. Se vaatii asianmukaisia yhdenmukaistamismekanismeja, koska perinteisen ja ketterän tiimien organisaatioyksiköt voivat keskinäisesti nähdä toisensa kilpailevina kokonaisuuksina ja toimintoina. (Haffke ym., 2017b.)

Haffke ym. (2017b) ja Horlach ym. (2017b) ovat yhteisessä tutkimuksessaan määritelleet myös hieman eri näkökulmista viisi tyyppiä, jotka ovat tyypillisiä ketterän IT:n toimituksessa. Nämä tyypit on luokiteltu seuraavasti: ”perinteinen IT ja bimodaalinen kehitysprosessi”, ”perinteinen IT ja ketterä IT-ulkoistaminen”, ”bimodaalinen IT-hankinta”, ”bimodaalinen IT” sekä viimeisenä ”ketterä IT”. Ensimmäiselle bimodaaliselle IT-tyypille on ominaista perinteinen IT, jossa bimodaalisuus rajoittuu kehitystyöhön. Se käyttää sekä ketterää että perinteistä prosessipohjaista vesiputousmenetelmää. Muut vaiheet kuten suunnittelu, testaus ja operointi noudattavat perinteistä vesiputoustapaa, jossa jokaisessa vaiheessa on korkea kontrollin taso. (Horlach ym., 2017b.) Bimodaalista kehittämisen lähestymistapaa sovelletaan olemassa olevien taustajärjestelmien (engl. systems of records) uusien muutosten kehittämiseen. Toinen bimodaalisen IT:n malli keskittyy IT-organisaation perinteisiin kyvykkyyksiin. Ketterä IT saavutetaan kolmannen osapuolen toimittajien tai tytäryhtiöiden kautta. Tämä johtaa osittain ulkoistettuun IT-organisaatioon, jossa on perinteisesti organisoitu ("hidas") sisäinen IT ja ketterä ("nopea") ulkoinen IT. (Horlach ym., 2017b.) Kolmas tyyppi käsittää bimodaalisen lähestymistavan IT-hallintaan. Tämä tarkoittaa, että yhden IT-toimitusmallin ulkoistaminen ja muiden mallien sisäinen ylläpito ei ole ainoa näkyvä lähestymistapa ketteryyden mahdollistamiseksi perinteisessä IT:ssä. Molempien ulkoistaminen on myös suosittua. (Haffke ym., 2017b.) Esimerkiksi ulkoisten kumppaneiden taitojen joustava integroituminen on eräs peruste ulkoisten palvelujen käyttämiselle sekä perintei-

sessä että ketterässä IT:ssä. Ketterän toimintatavan ulkoistaminen on kriittinen tilanteissa, joissa yritykset eivät kehitä tarpeeksi omaa osaamistaan. Kun ulkoistetaan molemmat toimintatavat, voidaan erottaa kaksi erilaista IT-organisaatiota, jotka muokkaavat sisäisen tietotekniikan roolia: (1) asiakas- ja toimittajayhteistyö yritysten IT:n ja ulkoistamiskumppanin välillä ja (2) sisäinen IT-projektiorganisaatio, jossa IT toimii projektipäällikön asemassa ja projektiryhmänä toimii ulkoistamiskumppani. Neljännessä tyypissä bimodaalinen IT toteutetaan sisäisesti antamatta ulkoisille palveluntarjoajille merkittävää roolia. Tämäntyyppiselle bimodaaliselle IT:lle on ominaista kahden IT-toimitustavan erottaminen struktuureista ja prosesseista. Tämä voi tarkoittaa johtamisen erottelua siten, että esimerkiksi digitaalisen liiketoiminnan johtaja (CDO) on vastuussa ketterästä IT:stä ja yrityksen toimitusjohtaja vastaa perinteisestä IT-organisaatiosta. Viimeiselle eli viidennelle bimodaaliselle IT-tyypille on ominaista sisäinen ja ketterä IT-organisaatio, joka pyrkii edistämään nopeasti palveluiden markkinoille saattamista nopeasti reagoivan IT-organisaation avulla. (Haffke ym., 2017b; Horlach ym., 2017b.)

3.3 Yhteenveto

Tässä luvussa kuvattiin perinteisen ja ketterän IT:n käsitteitä. Ensimmäiseksi tarkasteltiin kahden nopeuden IT:tä ja sen kyvykkyyttä vastata tehokkaasti ja joustavasti liiketoimintaympäristöjen muutoksiin. Tässä yhteydessä tarkasteltiin ketterän ja perinteisen IT:n lähestymistavan välisiä eroja. Tämän lisäksi esitettiin myös erilaisia näkökulmia organisaatioiden IT-toiminnoille, koska siirtyminen perinteisestä toimintamallista ketterän toimintamallin käyttöönottoon on osoittautunut vaikeaksi. Ketterän toimintamallin käyttöönoton onnistumisen varmistamiseksi on kehitetty erilaisia strategioita ja lähestymistapoja. Yhteenvetona voidaan todeta, että ketterä toimintamalli edellyttää muutoksen hyväksymistä ja omaksumista koko organisaatiotasolla, koska ketteryys vaikuttaa kaikkeen tekemiseen.

Bimodaalisuus ei kuitenkaan sovi kaikkeen tekemiseen. Perinteisen toiminnan ohessa voi olla nopea IT, jonka onnistumiselle on erilaiset mittarit. Organisaation toimiminen kahdessa eri kulttuurissa voi olla haasteellista. Myös yrityksen toimiala tai toimintaympäristö on otettava huomioon, koska valmiudet ja toimintakulttuurit voivat olla hyvin erilaisia. Järjestelmiä ja palveluita voidaan tehdä ketterästi, mutta on suunniteltava hyvin tarkkaan, mitä kannattaa tehdä perinteisellä tai ketterällä tavalla. Organisaatiolla voi on hyvin ketterä toimintamalli esimerkiksi digitaalisessa liiketoimintaympäristössä sekä aktiivista kehittämistä vaativissa palveluissa sekä perinteisempi malli muissa IT-toimituksissa ja IT-palveluissa. Tällöin organisaatiolla on kaksi nopeudeltaan ja joustavuudeltaan erilaista toimintamallia, joilla voi olla yhteiset tukitoiminnot.

Haffke ym. (2017b) esittävät tutkimuksessaan, että yrityskulttuurin tulee sallia kokeilut ja epäonnistumiset, jotta organisaation jäsenet voivat oppia uutta. Epäonnistuminen itsessään ei ole suoraan oikeutettua ja tätä tulee arvioida

kriittisesti. Tiukasti säädellyissä ja kriittisissä toimintaympäristöissä epäonnistuminen on harvoin hyväksyttävää, koska järjestelmien on oltava turvallisia ja luotettavia. Virheistä palautuminen järjestelmien aikaisempaan tilaan tulee tapahtua hallitusti ja nopeasti. Mahdolliset epäonnistumiset sen sijaan tulee jalostaa oppimiseksi, jotta organisaatio tulee vahvemmaksi. Vaikka uusien muutosten käyttöönottoon liittyy aina riskejä, DevOpsin nopeus ja ketteryys auttavat virheiden korjaamisessa.

Seuraavaksi tarkastellaan DevOpsin toimintamallia, joka liittyy läheisesti bimodaalisen IT:n kontekstiin. DevOps on ketterä toimintamalli, joka soveltuu tuotannon ja kehityksen väliseen yhteistyöhön koko palveluiden elinkaaren ajaksi. Ketterä kehittäminen tukee nopeaa ja vaiheittaista kehittämistä ja ohjelmistoversioiden julkaisua.

4 DevOps

DevOps muodostuu sanoista kehittäminen tai kehitys (engl. development) ja operointi tai tuotanto (engl. operations). Nämä yhdessä muodostavat DevOps-käsitteen, jolla pyritään edistämään kehitys- ja tuotantotoimintojen välistä yhteistyötä ja kyvykkyyttä tehdä muutoksia tuotantoon. DevOps pohjautuu ketterään kehitykseen, mutta korostaa ja parantaa samalla yhteistyötä eri liiketoimintaryhmien välillä. DevOpsia kuvataan työskentelytavaksi, jossa IT- ja kehitystyöryhmien eli kehitys- ja tuotantotiimien toiminta on tiivistä yhteistyötä päämäärien tavoittamiseksi. (Gruver & Mouser, 2015, 81, 103; Hüttermann, 2012, 4; Verone, 2018.) Agutterin (2017, 110) mukaan DevOps edustaa IT-kulttuurin muutosta, keskittyy nopeaan IT-palveluiden toimittamiseen ja omaksuu ketterät Lean-käytännöt järjestelmäsuuntautuneessa lähestymistavassa. DevOps-toteutukset hyödyntävät teknologiaa ja erityisesti automaation työkaluja, jotka voivat hyödyntää yhä enemmän ohjelmoitavaa ja dynaamista infrastruktuuria elinkaaren näkökulman huomioiden. (Agutter, 2017, 110.) Ohjelmiston harjoittajien ja tutkijoiden keskuudessa ei ole yleistä tai yksiselitteistä määritelmää DevOpsin sisällöstä. Kirjallisuudessa on esitetty useita määritelmiä, joissa suurimmassa osin DevOps-käsitettä käytetään korostamaan kehittämisen ja tuotannon toiminnan keskinäistä yhteistyötä. (Fitzgerald & Stol, 2017; Lwakatare, Kuvaja & Oivo, 2015; Naryan, 2015.) DevOps voidaan määritellä myös ajattelutavaksi (engl. mindset), kulttuuriksi ja joukoksi teknisiä käytäntöjä (Daniels & Davis, 2016; Leffingwell, 2018).

Babb, Nørbjerg, Yates ja Yates (2017) toteavat, että DevOps tarjoaa uuden konseptoinnin Agile-kehitykselle, joka on yhdenmukainen automaatiologiikan kanssa. DevOpsin käytännöt täydentävät Agilen käytäntöjä korostamalla operointia ohjelmisto- ja järjestelmäkehityksessä, Agile-käytännöt vastaavasti korostavat asiakkaiden panosta kehitykseen. DevOps syntyi Agile-ohjelmistokehityksen periaatteista, jotka liittyvät tiheisiin julkaisuihin ja työkalujen korkeaan automatisointiin. Nämä laajentavat jatkuvaa suunnittelua, kehitystä, testausta ja integraatiota toimitukseen, käyttöönottoon ja seurantaan asti. Babb ym. (2017) väittävät, että Agile-kehitys tukee organisaation muutoksen eettisyyttä yhteistyöllä ja oppimisella. Sen sijaan DevOps korostaa entistä

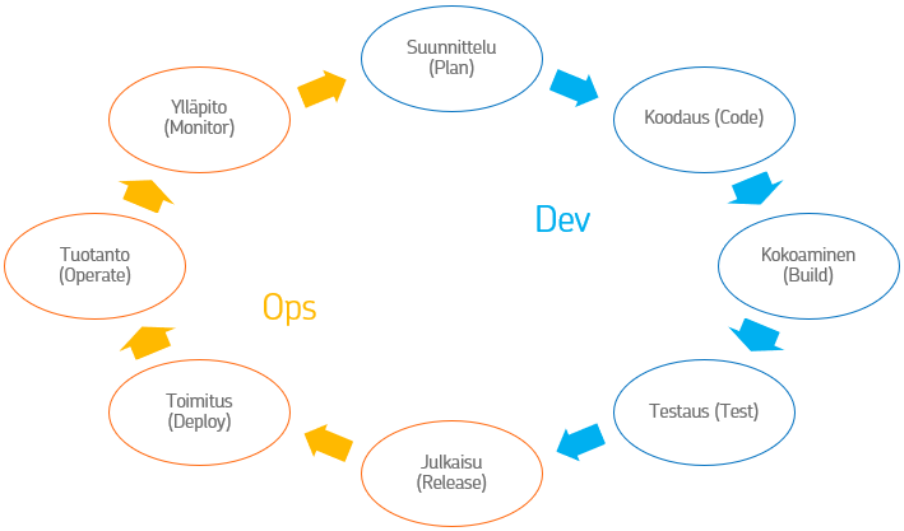
enemmän organisatorisen muutoksen toteuttamista liiketoiminnan tavoitteiden saavuttamiseksi sekä standardointiprosesseja, automaatiota sekä datapohjaisia parannuksia prosesseihin ja tuotteisiin.

DevOps on osa SAFe -viitekehystä (Scaled Agile Framework), joka on eräänlainen viitekehys ketterien ja skaalautuvien menetelmien mallista. (Scaled Agile Framework, 2018.) SAFe perustuu Lean- ja ketterään periaatteeseen, joka antaa kaikille SAFen rooleille ja käytännöille pohjan. DevOps kuuluu SAFe-viitekehyksen hanke- ja arvovirtaustasoon. Toimintamalli esiintyy SAFessa omana osakokonaisuutena, mutta sillä on tarkoituksenmukainen rooli viitekehyksessä. DevOps liittyy Lean-Agile-hankkeisiin (engl. program), jossa yksi arvovirtaus (engl. value stream) voi sisältää useamman kuin yhden julkaisuketjun. Ketjujen välillä on keskinäisiä riippuvuussuhteita. (Scaled Agile Framework, 2018.) SAFe-viitekehyksen DevOps-tiimissä jäsenet toimivat reaaliaikaisesti ja toimituskeskeisesti tietyssä arvovirtauksessa. He esimerkiksi vastaavat palvelun tai tuotteen työjonosta (engl. backlog), joka liittyy palvelun asteittaiseen kehittämiseen ja parantamiseen. Jotta loppukäyttäjät saavat lisäarvoa, kehityksen ja julkaisun eri toiminnot on yhdistetty tiiviisti yhteen kokonaisuuteen. SAFen näkökulmasta tämä tarkoittaa sitä, että tuotannosta vastaava taho ja kehityksestä vastaava Agile-tiimi integroidaan keskenään, jolloin jäsenistä tulee osa ketterää toimitusjunaa (ART, Agile Release Train). (Scaled Agile Framework, 2018.)

Leffingwell (2018) toteaa, että ilman DevOps-lähestymistapaa, uusia ominaisuuksia toteuttavien ja tuotantoympäristön vakautta ylläpitävien välillä on usein merkittäviä jännitteitä. Kehityksen panosta mitataan loppukäyttäjille toimitetun liiketoiminta-arvon mukaan, kun IT-palvelunhallintaa mitataan tuotannon ympäristön ja vakauden mukaan. Vastakkaiset liiketoimintatavoitteet, toimituksen tehottomuus ja organisatorinen kitka aiheuttavat ristiriitoja. Leffingwell (2018) korostaa, että toimintamallina DevOps pyrkii estämään siiloutumista kehittämällä ja julkaisemalla pieniä versiopaketteja loppuasiakkaalle virtausprosessissa, jota kutsutaan jatkuvaksi toimitusputkeksi. DevOps on olennainen osa jokaista arvovirtausta ja kiinteä osa SAFea. Monet SAFe-konseptit ja periaatteet kuten järjestelmäajattelu, pienet versiopaketit, lyhyet iteraatiot, nopea palaute tukevat suoraan DevOps-periaatteita. Lisäksi SAFe-käytännöt kuten jatkuva tutkiminen, jatkuva integrointi, jatkuva käyttöönotto ja julkaisu tukevat suoraan liiketoiminnan tarvetta. (Leffingwell, 2018.)

DevOpsissa kehittäjillä viitataan järjestelmäkehittäjiin, testaajiin, laadunvalvojiin sekä ylläpitäjiin, kun taas operoinnilla (tuotannolla) tarkoitetaan resursseja, jotka vastaavat järjestelmän, tuotteen tai palvelun tuotannosta kuten asiantuntijat tai järjestelmävastaavat, tietokantavastaavat ja verkkoteknikot, jotka vievät ohjelmistoja tuotantoon ja hoitavat tuotantoinfrastruktuuria. (Hüttermann, 2012, 4.) DevOps kuvaa virtaviivaisuutta ohjelmiston toimitusprosessissa. DevOps ei pelkästään valtuuta, että ohjelmiston käyttöönotto on nopeampaa, vaan se auttaa luomaan korkealaatuisia ohjelmistoja. Ketterä DevOps korostaa nopeutta ja laadun parantamista prosessin aikana. Farrohan ja Farrohan (2014) mukaan DevOps on tavallaan päätty-

mätön silmukka, joka pohjautuu jatkuvan palautteen antamiseen ja parantamiseen (kuvio 4). Humble ja Molesky (2011) tunnistavat neljä DevOpsin periaatetta, joita ovat kulttuuri, automaatio, mittaaminen ja jakaminen. Seuraavissa alaluvuissa käsitellään tarkemmin DevOpsin periaatteita.



KUVIO 4 DevOps-työnkulun malli (Farrohan & Farrohan, 2014, mukaan)

4.1 Kulttuuri

DevOps edellyttää kulttuurimuutoksen hyväksymistä ja yhteistä vastuuta korkealaatuisten ohjelmistojen toimittamiseksi loppukäyttäjille. Käytännössä tämä tarkoittaa, että DevOpsissa kehittäjät ja operoijat keskittyvät julkaisemisen hallintaprosessiin eli onnistuneeseen toimitukseen aina koodista käyttöönottoon asti. DevOpsissa tuetaan työskentelykulttuuria, jossa vuorovaikutus on sujuvaa ja roolien välillä ei esiinny vastakkaisasettelua. Toimivan työskentelykulttuurin saavuttamiseksi tiimin jäseniltä edellytetään yhteistyötä, avointa vuorovaikutusta, palautteen antamista ja yhteisiä tavoitteita (Gruver & Mouser, 2015, 51). Puppetin ja DevOps Research & Assessmentin julkaisema raportti ”2017 State of DevOps Report” korostaa DevOpsin käytäntöjen myös parantavan organisaation kulttuuria ja vahvistavan työntekijöiden sitoutumista. Agutter (2017, 110) toteaa, että ketterä DevOps omaksuu täyden ohjelmistokehityksen ja tuotannon elinkaaren. Sitä kutsutaan joustavaksi filosofiaksi ja lähestymistavaksi (Agutter, 2017; Rangama, Svendsen, Scholman, Buchanan & Meyler, 2017). DevOps-ajattelumalli keskittyy näkökohtiin, joita ovat omistajuus ja vastuuvelvollisuus, järjestelmäajattelu, jatkuva kokeilu ja oppiminen, yhteistyökulttuuri ja jakaminen, automaatio, turhien välivaiheiden poistaminen, Lean-periaatteet sekä mittaaminen. (Agutter, 2017, 110.)

4.2 Automaatio

DevOpsin tavoitteena on automatisoida työvaiheita, jolloin kehitystyöstä tulee nopeampaa ja laadukkaampaa (Gruver & Mouser, 2015, 61). Tämä mahdollistaa asiakkaiden palvelemisen nopeammin. DevOpsissa voidaan automatisoida ohjelmiston kääntäminen, testaus ja julkaisu sekä monitorointi. Automaation avulla voidaan varmistaa, että ohjelmisto on toteutettu samalla tavalla joka kerta. Hüttermann (2012, 30) korostaa, että kun ohjelmisto testataan ja tarkastetaan samalla tavalla joka päivä, niin virheitä ei pääse läpi.

Automaatio on välttämätöntä lyhyen toimitusajan ja nopean palautteen saavuttamiseksi. Palautteella saadaan kehitystä ohjattua nopeammin toivottuun suuntaan. Humblen ja Moleskyn (2011) mukaan kehityksen ja testauksen automatisointi on edellytys alhaisten läpimenoaikojen saavuttamiseksi ja nopean palautteen saamiseksi. DevOps edistää läpinäkyvyyttä. Sovellukset ja palvelut testataan perusteellisesti jo kehityksen aikana, mutta testaaminen ei pääty siihen, kun sovellukset on viety tuotantoon, vaan ratkaisuja tulee seurata koko prosessin ajan. Agutter (2017, 111) korostaa, että automaation aktiviteetit kuten testaus ja käyttöönotto ovat tärkeitä ominaisuuksia DevOpsissa. Automaatiolla voidaan nopeuttaa toimitusta ja vähentää riskejä. Automaatio täytyy integroida muutoshallinnan hallinnoinnin vaatimuksiin. (Agutter, 2017, 111.)

Babbin ym. (2017) mukaan DevOps-lähestymistavan tarkoituksena on lisätä IT-organisaatioiden kykyä reagoida nopeasti ja julkaista uusia ohjelmistoversioita usein poistamalla organisaattoriset ja käytännön esteet kehityksen ja tuotannon väliltä. Käytännössä tämä toteutetaan automatisoimalla operatiiviset tehtävät. Näitä ovat laitteiston kokoonpano ja asennus, järjestelmien integrointi, käyttöönotto ja valvonta. (Babb ym., 2017.)

DevOpsissa jatkuva integrointi ja käyttöönotto pyrkivät jatkuvaan ja erittäin automatisoituun ohjelmoinnin, testauksen, toimituksen ja käyttöönoton virtaukseen. Babbin ym. (2017) mukaan se on riippuvainen standardoiduista arkkitehtuureista ja kehittyneistä työkaluista, jotka automatisoivat rakentamisen, testauksen, käyttöönoton ja seurannan prosessit. Ohjelmistokehitysprosessin ja käytössä olevan palvelun, järjestelmän tai tuotteen mittaukset syötetään takaisin kehittämiseen, jotta voidaan tehdä muutoksia tai lisäyksiä sekä prosessin parannuksia. (Babb ym., 2017.)

4.3 Mittaaminen

Toimituskyvyn yhteisymmärryksen saavuttaminen ja tavoitteiden parantaminen voidaan saada aikaiseksi vain mittaamisen avulla. Hüttermannin (2012, 31) mukaan DevOpsin hyödyt on kyettävä näyttämään myös organisaation johdolle esimerkiksi parempana kassavirtana tai nopeampana markkinoille tuloina. Mittaustapojen tulee kuvastaa liiketoiminnan arvoja ja

esimerkiksi toimituksen syklin kesto on kriittinen mittari kaikille sidosryhmille. DevOpsissa korostuu erityisesti reaaliaikainen mittaaminen. (Hüttermann, 2012, 31).

4.4 Jakaminen

DevOps luo kulttuurin, jossa jaetaan osaamista, ideoita, onnistumisia, prosesseja ja työkaluja. Hüttermannin (2012, 27) mukaan DevOpsissa työskennellään työryhmissä, joissa on kehittäjiä, testaajia, laaduntarkkailijoita ja ylläpitäjiä. Kun ryhmä jakaa tietämyksensä, kokemuksensa, onnistumisensa ja muita oppejaan toistensa kanssa, se luo yhteisöllisyyttä ja lisää perustietämystä muidenkin alueelta. Agutterin (2017, 110-111) mukaan kulttuuriin ja jakamiseen keskittyminen kannustaa yhteistyöhön ja kommunikointiin koko palvelun elinkaaren ajan. Tämä saadaan aikaiseksi käyttämällä rinnakkain sijoitettuja monitaitoisia tiimejä, jotka kaikki jakavat tavoitteen, joita asiakkaat haluavat. Esimerkiksi kaikki tiimin jäsenet ovat vastuussa muutoksen onnistumisesta toimintaympäristössä. He kantavat ja hyväksyvät kollektiivisen vastuun. Vastaavasti tyypillisessä IT-palvelunhallinnan lähestymistavassa asiantuntija voi olla yksittäinen henkilö, joka on vastuussa. (Agutter, 2017, 110-111.)

4.5 Yhteenveto

Operoinnin tai tuotannon näkökulmasta DevOps vaikuttaa merkittävästi kulttuuriin ja kurinalaisuuteen. Tuotantotiimien täytyy olla jatkuvassa yhteydessä muihin toimintoihin menettämättä kontrollia. Tuotannon ja kehityksen täytyy tiiviisti tehdä yhteistyötä saavuttaakseen jatkuvan prosessin. DevOpsissa monitorointi ja sovelluksen suorituskyvyn hallinta on tärkeämpää ja keskinäisen kommunikaation täytyy toimia moitteettomasti ja sujuvasti (Ebert ym., 2016). DevOps luotiin alun perin rikkomaan tiimien siiloutuminen, jotta tiimit voisivat toimia ja tehdä paremmin keskenään yhteistyötä (Rangama ym., 2017). Perinteisesti kehityksestä ja tuotannosta on vastannut vähintään kaksi erillistä tiimiä. Kehittäjät ovat vastanneet kehittämisestä ja tuotanto on vastannut järjestelmistä. Tuloksena on ollut jäykkä ja hidas toiminta. Kimin, Behrin ja Spaffordin (2011, 347) mukaan ristiriita kehityksen ja tuotannon välillä määrää ennalta epäonnistumisen koko IT-organisaatiossa. Ristiriidat viivyttävät palveluiden tai tuotteiden markkinoille menoa, tuotteen julkaisut kärsivät pidemmistä toimitusajoista ja ongelmallisesta kehityksestä. DevOpsia ei saa ajatella muotitrendinä vaan se on ymmärrettävä kokonaisuutena käytäntöjä ja kulttuurisia toimintamalleja. DevOpsia hyödyntävät ihmiset eivät ainoastaan paranna päivittäistä työskentelyä vaan he parantavat organisaationsa suorituskykyä, liikevaihtoa, kannattavuutta ja muita mitattavia tavoitteita (2017 State of DevOps Report).

Kuten todettu, toimintatapana DevOps sekoittaa kehittäjien ja IT-järjestelmän ylläpitäjien roolit, koska toimintamallilla pyritään pienentämään mahdolliset kommunikaatioon liittyvät ongelmat ja aikataululliset viiveet kehitystiimin ja tuotannon välillä. Kehittäjät joutuvat tekemään ylläpidon tehtäviä ja ylläpitäjät joutuvat mahdollisesti lukemaan ohjelmakoodia. Näin roolit menevät DevOps-tiimissä sekaisin. Useimmiten kehitys omaksuu varhaisessa vaiheessa ketterät toimintatavat, mutta muu organisaatio toimii perinteisemmin. Tämä johtaa siihen, että ketteryydestä saadut hyödyt jäävät tällöin rajalliseksi. Jotta ketteryydestä saadaan maksimaalista arvoa, tulee ajattelumalli omaksua koko organisaation tasolla. Keskeisintä DevOpsissa on avoimuuden ja läpinäkyvyyden kulttuuri, joka saavutetaan tuotannon ja kehittämisen välisellä yhteistyöllä. Eri organisaatioiden välinen tiivis yhteistyö luo pohjan menestymiselle. DevOps toimintatapa vaatii toimiakseen sujuvan automaatioputken. Sillä mahdollistetaan laatu, toistettavuus, nopeat palautteet sekä jatkuva oppiminen ja parantaminen.

Tässä luvussa kuvattiin DevOps-ajattelumallia palvelunhallinnan näkökulmasta. Yhteenvetona tarkastelusta voidaan todeta, että ketterässä toimintamallissa keskeistä on toimiva automaatio ja kattava monitorointi sekä tiivis yhteistyö eri osapuolten välillä, jotta IT-palvelutuotanto voi edelleen varmistaa häiriöttömän ja vakaan tuotantoympäristön. Kysymyksessä ei ole enää pelkästään yksittäisten käytäntöjen käyttöönotosta vaan myös kulttuurimuutoksesta. Tämä edellyttää muutoksen hyväksymistä ja omaksumista koko organisaatiotasolla, koska ketteryys vaikuttaa kaikkeen tekemiseen. Tapausorganisaatiossa automatisointia toteutettiin osittain kokoonpanojen, testausten ja monitoroinnin osalta. Automatisoinnilla vähennetään virheiden määrää, manuaalista ja toistuvaa työtä. Tämän avulla myös tehostetaan ja nopeutetaan uusien ohjelmistoversioiden julkaisua ja tuotantoon vientiä. Mittaamisen suunnitteleminen käynnistettiin pilotin aikana ja kehittämistyö jatkuu pilotin päättymisen jälkeen. Yhdessä tekeminen, osaamisen ja tiedon jakaminen korostuvat erityisesti DevOps-tiimissä. Jatkuva oppiminen ja osaamisen jakaminen konkretisoituivat ja toteutuivat enenevissä määrin.

Kokonaisuutena DevOps-toimintamalli soveltuu selkeimmin modernin ohjelmistokehityksen taustalle, joita ovat esimerkiksi pilvipohjaisilla teknologioilla toteutetut palvelut. DevOps sisältää monia käytäntöjä, jotka sopivat erityisesti kehityshankkeisiin tai aktiivisesti kehitettäviin palveluihin. DevOps toimintamallina parantaa perinteisen ja ketterän kehityksen mukanaan tuomia haasteita. Automatisointi vähentää manuaalista ja toistuvaa työtä ja rutiinityön automatisointi vähentää virheiden määrää. DevOps tehostaa ja nopeuttaa uusien ohjelmistoversioiden julkaisua ja tuotantoon vientiä ja pitkällä aikavälillä pienentää kehitys- ja ylläpitokustannuksia. DevOps-toimintamalli voi tuoda kehittämiseen myös riskejä tai haasteita. Esimerkiksi testauksen automatisointi voi olla työmäärältään vaativa ja suuri investointi. Tiheät versiopäivitykset eivät välttämättä sovellu kriittisiin toimialoihin ja järjestelmiin. Uudenlaiset prosessit ja työvälineet vaativat usein

käyttöönottovaiheessa panostusta kaikilta osapuolilta. Palvelujen tai järjestelmien kehittämisen elinkaari tulee myös huomioida. Kun palvelun kehittäminen ei ole enää aktiivista, niin DevOps-malli ei enää välttämättä tuota tarvittavia hyötyjä vaan voi muuttua kalliiksi ylläpitää. DevOpsin ja IT-palvelunhallinnan piirteiden vertailu taulukossa 2 nostaa esille ne haasteet, joihin tämä tutkimus pyrkii vastaamaan.

TAULUKKO 2 IT-palvelutuotannon ja DevOpsin piirteet

IT-palvelunhallinta		**DevOps**
Vakaus	Tavoite	Ketteryys & nopeus
Palveluajattelu	Ajattelu	Järjestelmäajattelu
Turvallisuus & luotettavuus	Palvelun fokus	Innovaatio
Perinteiset prosessit	Lähestymistapa	Kevyet prosessit
Siiloutunut	Kulttuuri	Tiivis yhteistyö, avoimuus
Työvaiheet osittain manuaalisia	Työvaiheet	Työvaiheet automatisoitu
Teknologiariippumaton	Teknologia	Teknologiariippuvainen
Harvemmat/sovitut	Julkaisut	Tiheät/nopeat
Perinteiset roolit ja vastuut	Roolit	Uudistetut roolit ja vastuut

5 Teoriaosuuden tulokset

5.1 DevOps ja IT-palvelunhallinta

DevOpsia voidaan tarkastella suhteessa ITIL-viitekehykseen tai IT4IT-viitearkkitehtuuriin, joita käytetään IT-palvelunhallinnassa. DevOps ja ITIL eivät ole täysin toisiaan tukevia, mutta ne eivät ole myöskään vastakohtia. Palvelusuunnittelu painottaa iteratiivista ja inkrementaalia tapaa määritellä ja suunnitella palveluita. Palvelustrategian mukaan jokaisen palvelun elinkaaren vaiheessa korostetaan jatkuvaa palautesykliä ja palvelun jatkuvaa parantamista. Jokainen elinkaaren vaihe on dynaaminen ja tukee muita vaiheita. ITIL keskittyy ihmisten, prosessien, tuotteiden ja yhteistyökumppaneiden hyödyntämiseen tehostamalla ja tukemalla palveluita. Jatkuva palautteen saaminen, parantaminen ja oppiminen ovat ominaista myös DevOpsissa. (Fitzgerald & Stol, 2017; Kim ym., 2014.) Koko palvelun elinkaari on DevOpsin kannalta tärkeä, koska se keskittyy palvelun toimittamiseen ja asiakkaan sekä toimittajan väliseen palvelusuhteeseen. ITIL mahdollistaa DevOpsin mukaisen virheiden havaitsemisen ja toipumiseen liittyvän prosessin.

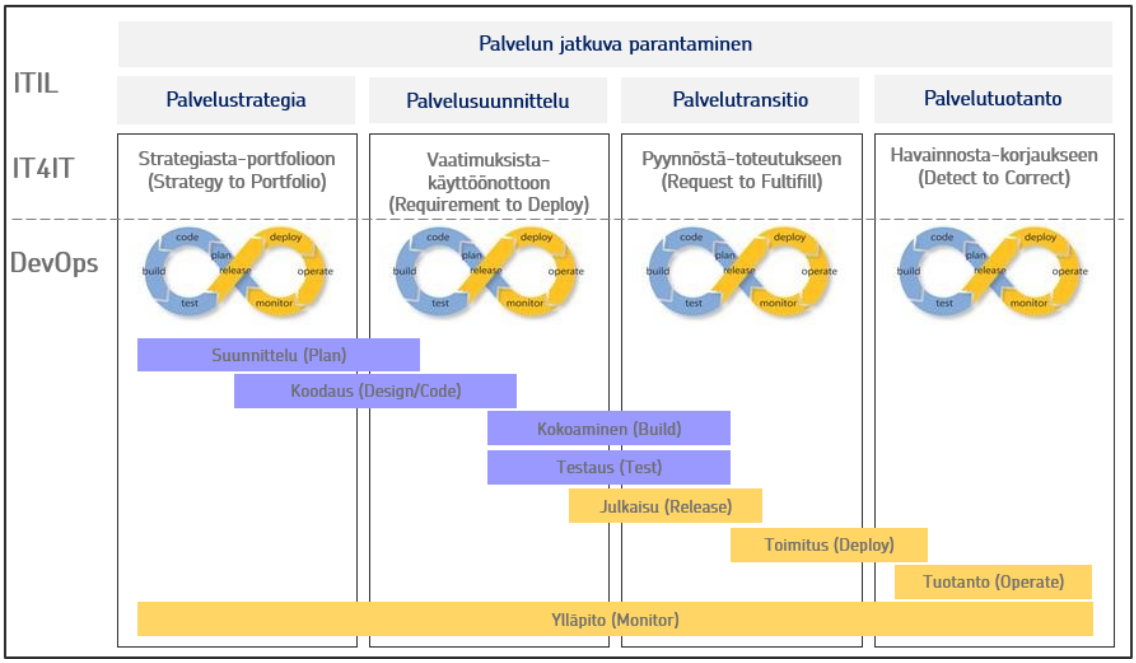
Palvelutuotannon keskeinen periaate on vakauden ja reagointikyvyn hallinta (Cannon & Wheeldon, 2011). Kehitys pyrkii vastaamaan asiakkaiden tarpeisiin. Liiketoiminnan ja IT:n vaatimukset muuttuvat jatkuvasti, mikä vaatii ketteryyttä tuottaa uusia toiminnallisuuksia, mutta myös samalla vakauden ylläpitämistä palveluiden suorituskyvyssä (Nerur, Mahapatra & Mangalaraj, 2005). Rochen (2013, 5) mukaan merkittävänä tekijänä DevOpsissa ja IT-palvelunhallinnassa on toimintavarmuuden ja laadun varmistaminen. Ketterän DevOpsin jatkuvassa integraatiossa ja julkaisussa ohjelmistoa rakennetaan iteratiivisesti ja inkrementaalisesti toteuttamalla pieniä muutoksia nopeassa aikataulussa tuotantoon. Koska inkrementaalisesti tehdyt muutokset koskevat vain tiettyä osaa ohjelmistossa, nähdään ohjelmiston laadun ja toimintavarmuuden myös paranevan. (Roche, 2013, 5.)

ITIL:in palveluiden elinkaari lähestymistapana auttaa organisaatioita suostumaan haluttuihin muutoksiin, hyödyntämään nykyistä infrastruktuuria ja ymmärtämään, mitä se tarvitsee toteuttaakseen arvon realisoitumisen. Palve-

lutuotannon prosessialueet voivat antaa arvokasta panosta DevOpsille. Palvelupyyntöjen, häiriöiden ja ongelmien hallinta sekä keskeiset suorituskykymittarit antavat suuntaa palvelun jatkuvaan parantamiseen DevOpsissa. Palvelutuotannon ja DevOpsin integrointi auttavat parantamaan yleistä asiakastyytyväisyyttä ja palvelun käytettävyyttä. DevOpsissa automatisoinnilla voidaan parantaa palvelutoimituksen suorituskykyä erityisesti tapahtumienhallinnan sekä häiriöhallinnan alueilla. (Ebert ym., 2016.) DevOps-mallin tarkoituksena on muun muassa soveltaa jatkuvan toimituksen periaatteita ja jatkuvaa integraatiota palvelujen kehittämisen tehokkuuden parantamiseksi (Fitzgerald & Stol, 2017). ITIL:in seitsemän vaiheen parannusprosessi voi parhaimmillaan auttaa parantamaan sitä (Himi ym, 2011).

IT-toiminnon organisoinnin tueksi myös IT4IT tarjoaa toimintamallin. Open Groupin IT4IT-viitearkkitehtuuri sisältää arvoketjuihin perustuvan toimintamallin liiketoiminnan IT:n hallintaan. IT4IT:n mukaisesti määritelty toimintamalli palvelee yrityksiä ja tukee todellisia käyttötapoja esimerkiksi joukkoistaminen (engl. cloudsourcing), Agile, DevOps. IT4IT-viitearkkitehtuuri täydentää olemassa olevia viitekehyksiä ja menetelmiä kuten ITIL:iä, hyvää tietohallintotavan viitekehystä eli COBIT:ia (engl. Control Objectives for Information and related Technology), SAFea ja kokonaisarkkitehtuurin viitekehystä eli TOGAF:ia (The Open Group Architecture Framework, 2018). IT4IT luo arvoketjun, joilla palvelut voidaan kuvata alusta loppuun siten, että palvelun elinkaaren jokainen vaihe on seurattavissa ja yhdistettävissä edelliseen vaiheeseen. IT4IT-viitearkkitehtuuri keskittyy IT-palvelujen määrittelyyn, hankintaan, käyttöön ja hallintaan kokonaisvaltaisesti tarkastelemalla koko IT-arvoketjua. (Josey, 2017, 24-25.) Tämän tyyppinen lähestymistapa soveltuu hyvin yhteen Lean-Agile-tyyppisen arvovirta-ajattelun kanssa.

IT4IT-viitearkkitehtuuri on organisoitu IT-arvoketjun (engl. value delivery chain) ympärille, joka käsittää neljä arvovirtaa (engl. value streams). Jokainen arvovirta tukee tehokkuutta ja ketteryyttä. (IT4IT Reference Architecture, 2017.) ”Strategiasta-portfolioon”-arvovirta (engl. strategy to portfolio) huolehtii keskitetysti kaikkien liiketoiminnan tarpeista ja kehittää sekä dokumentoi palvelukonseptin uusista ja parannetuista palveluista. Tässä arvovirrassa valmistellaan liiketoimintalähtöisiä tarpeita ja vaatimuksia toteutusta tai hankintaa varten. ”Vaatimuksista-käyttöönottoon”-arvovirta (engl. requirement to deploy) tuo palvelut palvelukatalogiin asiakkaiden saataville. Tässä arvovirrassa voidaan hallita muutosten kehittämistä ja läpivientiä. ”Pyynnöstä-toteutukseen”-arvovirta (engl. request to fulfill) huolehtii, että palvelupyynnöt käsitellään oikeassa paikassa ja loppukäyttäjä saa tilaamansa palvelut sovitun mallin mukaisesti. Palvelu on käyttöönotettu ja asianmukaisesti kuvattu konfigurointihallinnan tietokannassa eli CMDB:ssä (engl. configuration management database). ”Havainnosta-korjaukseen” -arvovirta (engl. detect to correct) hoitaa ongelmat ja toimii valvontajärjestelmänä. ”Havainnosta-korjaukseen”-arvovirta sisältää myös muutoshallinnan, dokumentoinnin sekä palvelutasonhallinnan tehtäviä. (Josey, 2017, 26; IT4IT Reference Architecture, 2017.) Kuvio 5 esittää ITIL:in, IT4IT:n ja DevOpsin linkitykset toisiinsa.



KUVIO 5 ITIL/IT4IT ja DevOps yhtäläisyydet

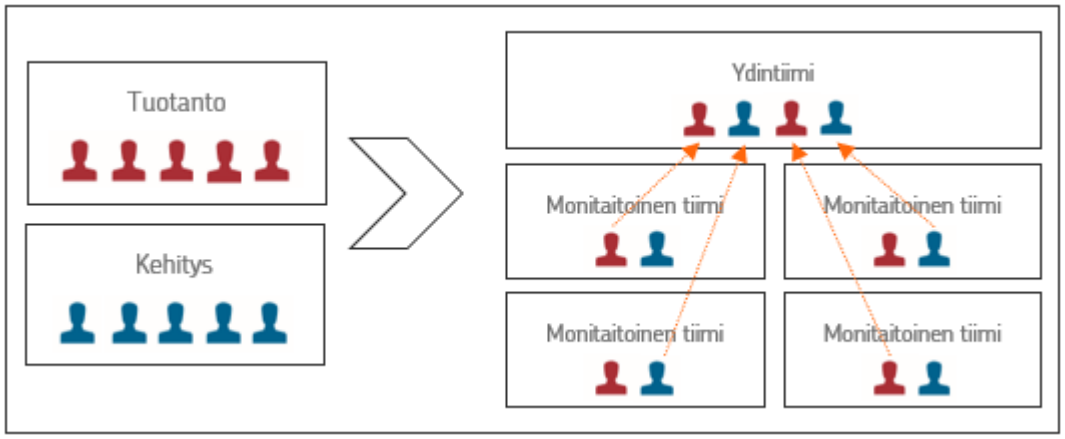
DevOps sopii hyvin yhteen monien viitekehysten kanssa. SAFessa mainitaan erityisesti DevOps, joka on syntynyt Agile-ympäristössä (Verona, Duffy & Swartout, 2016). Verone (2018) nostaa esille toisenlaisen huomion ja toteaa, että ITIL:in osalta tilanne kuitenkin on hieman erilainen. ITIL on laaja viitekehys, joka käsittää useita ohjelmiston elinkaaren osa-alueita. DevOps ja jatkuva toimitus katsovat, että tuotannolle toimitetut muutosten määrät olisivat pieniä ja tapahtuvat usein. Käytännössä ITIL:issä vaikuttaisi toteutuvan vastaavanlainen ilmiö. Vanhat järjestelmät ovat melko usein monoliittisia ja näissä tapauksissa tarvitaan ITIL-prosessia hallitsemaan monimutkaisia muutoksia, jotka usein liitetään monoliittisiin järjestelmiin. Suurissa organisaatioissa monoliittisten järjestelmien todennäköisyys ja osuus on erittäin korkea. On huomattu, että monet ITIL:issä kuvatut käytännöt taipuvat suoraan vastaaviin DevOps-käytäntöihin. ITIL määrittelee konfigurointihallintajärjestelmän ja konfigurointitietokannan. Tämäntyyppiset järjestelmät ovat myös DevOpsin osia. (Verona ym., 2016; Verone, 2018.)

5.2 Kehityksen ja tuotannon välinen yhteistyö

Ketterissä kehitysmenetelmissä otetaan tyypillisesti huomioon ainoastaan ohjelmistokehityksen näkökulma. Palvelun tai järjestelmän julkaisuun, tuotantoon siirtoon ja operointiin liittyvät tehtävät ja toiminnot jäävät tällöin toisarvoisiksi. Hüttermannin (2012, 36) mukaan tämän päivän ohjelmistotuotanto aiheuttaa jakautuneisuutta (siiloutumista) tuotantoprosessiin. Siiloutuminen esiintyy erillisinä toiminnallisina funktioina kuten kehitys-, laadunvarmistus- ja ylläpitotoiminnot (Hüttermann, 2012, 36; Rangama ym., 2017). Toimintojen ongelmat

konkretisoituvat, koska tiimien tavoitteet ovat erilaiset. Kehityksen tavoitteena on toimivan ratkaisun toteuttaminen, kun toisaalta operointi pyrkii varmistamaan vakaan ja toimivan ympäristön. Kyseinen ilmiö korostaa arvomaailman erilaisuutta, koska kehitys haluaa tehdä muutoksia ja ominaisuuksia, mutta operointi suhtautuu varauksella julkaisun aiheuttamaan muutokseen ja kehityksen vastuun siirtämiseen operoinnille. (Humble & Molesky, 2011; Hüttermann, 2012, 36.)

Balalaie, Heydarnoori ja Jamshidi (2016) esittelevät esimerkin DevOpsin yhteistyötä korostavasta toimintamallista (kuvio 6). Tehokas toimintatapa saatiin aikaan hajauttamalla tiimirakenne. Aikaisemmin perinteiset ohjelmistomenetelmät kannustivat projektin jäsenten horisontaalista jakamista toiminnallisesti omiin tiimeihin kuten kehitys- ja operointitiimeihin. Kun tiimit järjesteltiin uudelleen pienimmiksi kokonaisuuksiksi, uudet tiimit koostuivat molempien toimintojen osaajista ja tiimit kykenivät tehokkaammin ottamaan vastuuta kokonaisista palveluista ja koko prosessista. Tiimin jäsenet tekevät yhteistyötä alusta alkaen ja luovat enemmän arvoa palvelun loppukäyttäjille. Lisäarvoa saadaan uusien ominaisuuksien ja muutosten säännöllisistä julkaisuista tuotantoon. Jokaisen palvelun ohjelmistokoodilla on huomattavasti parempi ylläpidettävyys ja ymmärrettävyys, kun tiimit keskittyvät tiettyyn palveluun. Balalaien ym. (2016) tekemän tutkimuksen tapausorganisaatiossa muodostettiin pieniä organisaatiorajat ylittäviä poikkitoiminnallisia, monitaitoisia tiimejä (engl. cross-functional team) jokaiselle palvelulle. Tiimit, joiden jäsenet tulevat organisaation eri funktioista ovat yleistyneet ja niitä hyödynnetään muun muassa erilaisissa kehitystehtävissä ja päätöksentekoprosesseissa. Lisäksi muodostettiin ydintiimi (engl. core team), joka on vastuussa jaetuista kyvykkyyksistä. Tiimi käsittää edustajan jokaisesta tiimistä. Ydintiimillä on kokonaisvaltainen näkemys palvelun vuorovaikutuksista ja se vastaa kriittisistä arkkitehtuuripäätöksistä. Balalaie ym. (2016) mukaan uudelleenorganisoitu tiimirakenne tukee kehitystiimiä organisatorisen vakauden säilyttämisessä.



KUVIO 6 DevOps-tiimin muodostuminen (Balalaien ym., 2016, s. 8 mukaan)

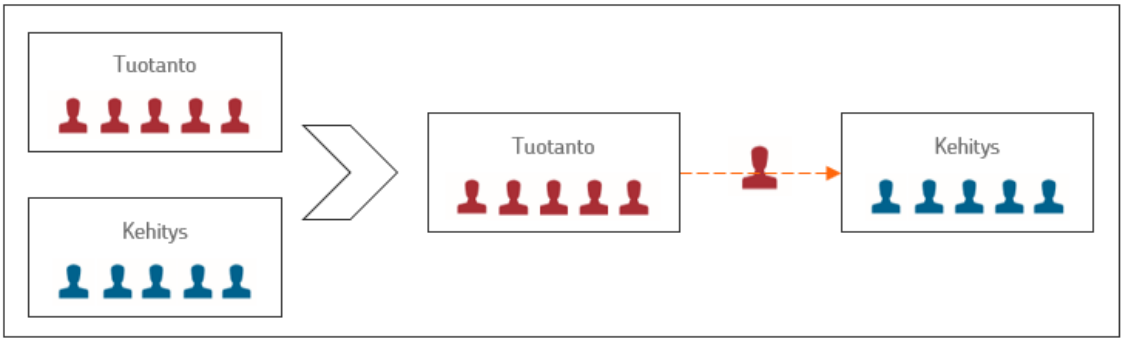
Naryan (2015, 58) kirjoittaa, että poikkitoiminnalliset, monitaitoiset tiimit eivät ole täysin uusi ilmiö. Ainoastaan poikkitoiminnallisuuden laajuutta voidaan

pitää uutena piirteenä. Agile-ohjelmistokehitystiimit ovat aina olleet monitaitoisia tiimejä, koska tiimit koostuvat arkkitehdeistä, analyytikoista, kehittäjistä ja testaajista. DevOpsissa monitaitoiset tiimit laajenevat käyttöönottoon ja IT-tuotantoon. Tässä vaiheessa monitaitoinen tiimi kykenee ketteryyteen toimituksessa. DevOps kannattaa kehityksen ja siihen liittyvän IT-tuotannon yhdistämistä, koska se muodostaa monitaitoisen tiimin kehityksen ja IT-tuotannon välille. Valitettavasti tämä näkökohta jätetään usein huomiotta DevOpsin teknisiin näkökohtiin verrattuna. IT-näkökulmasta katsottuna yleensä on kolme funktiota, joita ovat liiketoiminta, kehitys ja IT-tuotanto. DevOps-malli ei toteudu, jos IT-tuotannon henkilöt eivät ole osa kehittämisen organisaatiota. DevOpsin keskeinen piirre on, että yhdestä tiimistä löytyy sekä kehityksen ja tuotannon osaamista. (Naryan, 2015, 58.)

Monitaitoiset tuotetiimit omistavat tuotteensa - he vastaavat suunnittelusta, toteuttamisesta ja ylläpitämisestä. Monet organisaatiot säilyttävät kuitenkin erilliset ylläpitotiimit (virheiden korjausten ja vähäisten parannusten tekemistä varten) sekä IT-tuotannon. Ylläpito- ja IT-tuotantotiimit aktivoituvat, kun ohjelmistojulkaisu viedään tuotantoon ja kehitys alkaa työstämään seuraavaa julkaisua. Tähän liittyy yleinen mutta puutteellinen käsitys siitä, että ylläpitotyö vaatii vähemmän taitoa kuin täysimittainen kehittäminen. Tämän seurauksena projektin sponsorit, jotka haluavat vähentää tuotannon kustannuksia, valitsevat erillisen ja edullisemman ylläpitotiimin. Naryanin (2015, 66) mukaan tämä on epäedullista toimintaa, koska se vahingoittaa liiketoiminnan tuotosta ja vähentää IT:n ketteryyttä. Kun sama tuotetiimi kehittää ja ylläpitää, niin vältetään tuotantoon luovutus (engl. handoff) julkaisujen aikana. Tämä mahdollistaa helpomman virheenkorjausten yhdistämisen julkaisuun, koska tiimi on perehtynyt käynnissä oleviin muutoksiin. Ylläpitotiimin tärkeys korostuu myös sovelluksen elinkaaren lopun aikaisen tuen aikana (engl. end-of-life). Ylläpitotiimi pitää vanhan sovelluksen tai tuotteen käynnissä, kunnes uusi korvaaja on rakennettu. (Naryan, 2015, 66.)

Lwakatare, Kuvaja & Oivo (2016) ovat kuvanneet vastuita ja toimintatapoja, jotka muodostavat yhteistyökulttuurin hieman eri tavalla (kuvio 7). Heidän tutkimassaan tapausorganisaatiossa kullekin kehittäjätiimille oli operointitiimiin varattu henkilö, joka auttoi kehitystä infrastruktuuriongelmien ilmetessä. Bassin, Weberin ja Zhun (2015) mukaan, DevOpsin operoinnin alueella merkittävässä roolissa toimii Reliability Engineer. Kyseisen roolin omaavalla on useita velvollisuuksia. Reliability Engineer valvoo palvelua kehityksen jälkeisenä ajanjaksona ja toimii kontaktihenkilönä ongelmien ilmaantuessa, joten roolissa työskentely voi edellyttää korkeaa tavoitettavuutta. Ongelmien ilmetessä, Reliability Engineer tekee lyhyen aikavälin analyysin ongelman diagnosoimiseksi, lieventämiseksi ja korjaamiseksi, yleensä automaattisten työvälineiden avulla. Reliability Engineer -osaajalta vaaditaan erinomaista vianetsintä- ja diagnosointitaitoa sekä kattavaa ymmärrystä ja syväosaamista palvelusta, jotta ongelma voidaan korjata kokonaan tai korjata väliaikaisella ratkaisulla. Reliability Engineer havaitsee tai työskentelee tiiminsä kanssa löytääkseen ongelman juurisyyn. (Bass ym., 2015, 39-40, 214.) DevOpsin roolit voivat poiketa merkittävästi

perinteisen IT-palvelutuotannon rooleista tai tehtävistä tai tehtävät ovat jakaantuneet useammalle henkilölle. Esimerkiksi osa palvelutuotantopäällikön tehtävistä sisältää yhtäläisyyksiä Reliability Engineerin kanssa tai vastaavasti Reliability Engineerin tehtävät ovat hyvin samankaltaisia kuin teknisellä sovelluspäälliköllä. DevOpsissa tuotantoa hoitavilta henkilöiltä vaaditaan riittävän hyvää alueen substanssiosaamista sekä teknistä tuntemusta ja tietämystä tuotantoympäristöistä, sovellusten arkkitehtuureista ja tietokantojen tilanteesta.



KUVIO 7 DevOps-tiimin resurssointia (Lwakataren ym., 2016 mukaan)

Ketterän toimintamallin käyttöön siirtymisessä voi ilmetä vaikeuksia. Esimerkiksi organisaatiossa ei nähdä mahdollisuutta yhdistää ketteriä ja perinteisiä käytäntöjä. Työntekijät voivat myös kokea, että he eivät kykene omaksumaan tarvittavaa osaamista ja taitoja tai tulemaan toimeen ketterässä ympäristössä. Ketterässä lähestymistavassa ympäristö hämärtää roolien rajoja ja vaatii monipuolista osaamista (Conboy, Coyle, Wang & Pikkarainen, 2011). Gonzalezin (2017) mukaan DevOpsin kehittäjillä on monipuolinen rooli, koska heidän on tiedettävä, miten sovellus on rakennettu (ja mahdollisesti olla mukana sen kehittämisessä), mutta kehittäjien on keskityttävä myös sovelluksen toimintaan. Turvallisuus, toimintavalmius, infrastruktuuri ja testaus tulisi olla kehittäjien päivittäistä työtä.

6 TUTKIMUSMENETELMÄT

Tässä tutkimuksessa käytetään konstruktiivista, suunnittelutieteellistä tutkimusta (engl. Design Science Research Methodology, DSRM). Luvussa esitellään tutkimuksessa käytettävä tutkimusmenetelmä, luodaan katsaus kyseisen tutkimusmenetelmän viitekehyksiin ja perustellaan sen käyttö tässä tutkielmassa. Tutkimuksessa pyritään noudattamaan pääpiirteittäin Peffersin, Tuunasen, Rothenbergerin ja Chatterjeen (2007), Hevnerin, Marchin, Parkin ja Ramin (2004) sekä Ostrowskin ja Helfertin (2012) kuvattua rakennetta, joka kuvaa suunnittelutieteen prosessimallin vaiheet sekä valitut prosessit viitekehyksen kolmelle päätoimelle. Siinä esitetään myös, kuinka nämä kaikki osa-alueet toimivat yhteistyössä halutun ratkaisun saavuttamiseksi. Tässä tutkimuksessa luodaan suosituksia eli suunnitteluperiaatteita DevOpsin ja IT-palvelutuotannon yhdistämiselle. Luvussa tarkastellaan myös tutkimuksen tiedonkeruumenetelmiä.

6.1 Suunnittelutiede

Vuodesta 2004 lähtien Design Science (DS) -tutkimusmenetelmä on saanut enemmän huomiota tietojärjestelmätutkimuksissa. Suunnittelutieteestä on tullut hyväksytty lähestymistapa ja siihen liittyvä kirjallisuus on lisääntynyt merkittävästi. Suunnittelutiedettä pidetään myös tutkimuslähtöisenä, jossa voidaan käyttää erilaisia tutkimusmenetelmiä. (Ostrowski & Helfert, 2012.) Gregorin ja Jonesin (2017) mukaan yksi suunnittelutieteen tärkeimpiä kysymyksiä on, miten tieto on hankittu, kirjoitettu ja kommunikoitu.

Suunnittelutieteessä on hyödynnetty muilta tieteenaloilta peräisin olevia käsitteitä ja menetelmiä. Hevnerin ym. (2004) mukaan käyttäytymis- ja suunnittelutieteet liittyvät läheisesti tietojärjestelmätieteen tutkimukseen. Käyttäytymistieteet keskittyvät tutkimaan inhimillisiä tekijöitä, jotka liittyvät tietojärjestelmiin tai teknologian käyttöön tai hyväksyntään. Suunnittelutieteiden avulla luodaan organisaatioille uusia mahdollisuuksia tehostaa tai parantaa toimintaansa IT-artefaktien avulla. (Hevner ym., 2004; March & Smith, 1995.) Suunnittelutieteellinen tutkimus keskittyy ihmisten tuotoksiin, organisaatioihin, tieto-

järjestelmiin ja siinä pyritään luomaan tuotoksia, jotka palvelevat inhimillisiä tarkoituksia. Luodakseen tehokkaita ja tuloksia tuottavia artefakteja, suunnittelutieteessä tuotetaan ja sovelletaan tietoa tehtävistä ja tilanteista teoreettisen tiedon tuottamisen sijaan.

Suunnittelutieteen kehittelemät tulokset ovat luonteeltaan konstruktioita (engl. constructs), malleja (engl. models), metodeja eli menetelmiä (engl. methods) ja realisointeja eli toteutuksia (engl. instantiations). (March & Smith, 1995; Ostrowski & Helfert, 2012.) Konstruktiot on määritelty käsitteiksi ja konsepteiksi (March & Smith, 1995) sekä sanastoiksi ja symboleiksi (Hevner ym., 2004). Käsitteellistäminen on äärimmäisen tärkeää suunnittelutieteessä, koska ne määrittelevät termejä, joita käytetään tehtävien kuvaamisessa ja ajattelussa (March & Smith, 1995). Mallit eivät ole konseptin tavoin abstrakteja kokonaisuuksia kuten käsitteet. Mallit käyttävät käsitteitä esittämään reaalimaailman tilannetta (Hevner ym., 2004). Mallit tukevat ongelman ja ratkaisun ymmärtämistä ja edustavat usein ongelman ja ratkaisun komponenttien välistä yhteyttä mahdollistaen suunnittelupäätöksiä ja muutoksia reaalimaailmassa (Hevner ym., 2004). Menetelmä määritellään joukoksi vaiheita (algoritmi tai ohje), joka suorittaa tehtävän (March & Smith,1995). Esimerkkinä tällaisesta toteutuksesta on prototyyppi tai järjestelmä. March ja Smith (1995) toteavat, että artefakti rakennetaan suorittamaan jokin tietty tehtävä ja kun se on rakennettu valmiiksi, niin suunnitteluongelma on ratkennut. Kun artefakti sijoitetaan johonkin toimintaympäristöön, niin tutkijan tulee ymmärtää tämä ympäristö. Myös arvioinnin kriteerit tulee määritellä toiminnallisen ympäristön mukaan.

Tietojärjestelmätieteen tutkimuksessa tyypilliset tutkimusalueet kohdistuvat tietojärjestelmien, mallien tai menetelmien suunnitteluun, kehittämiseen ja toteuttamiseen. Tutkija on usein tutkimusprosessissa vahvasti mukana tai jäsenenä IT-organisaatiossa tutkimuksen aikana. Soveltavan tieteen tuloksia sovelletaan määriteltyyn reaalimaailman ongelmaan ja tutkimuksessa käytetään mahdollisesti useammin laadullisia tiedonkeruu- ja analyysitekniikoita. Suunnittelutieteen tavoitteena on muuttaa todellisuutta, konstruktoida uusia ratkaisuja ratkaistakseen ongelmia, luoda innovaatioita tai parantaa nykyistä tilannetta (March & Smith, 1995). Suunnittelutiede pyrkii myös kehittämään tietoa vaihtoehtoisten ratkaisujen eduista ja haitoista.

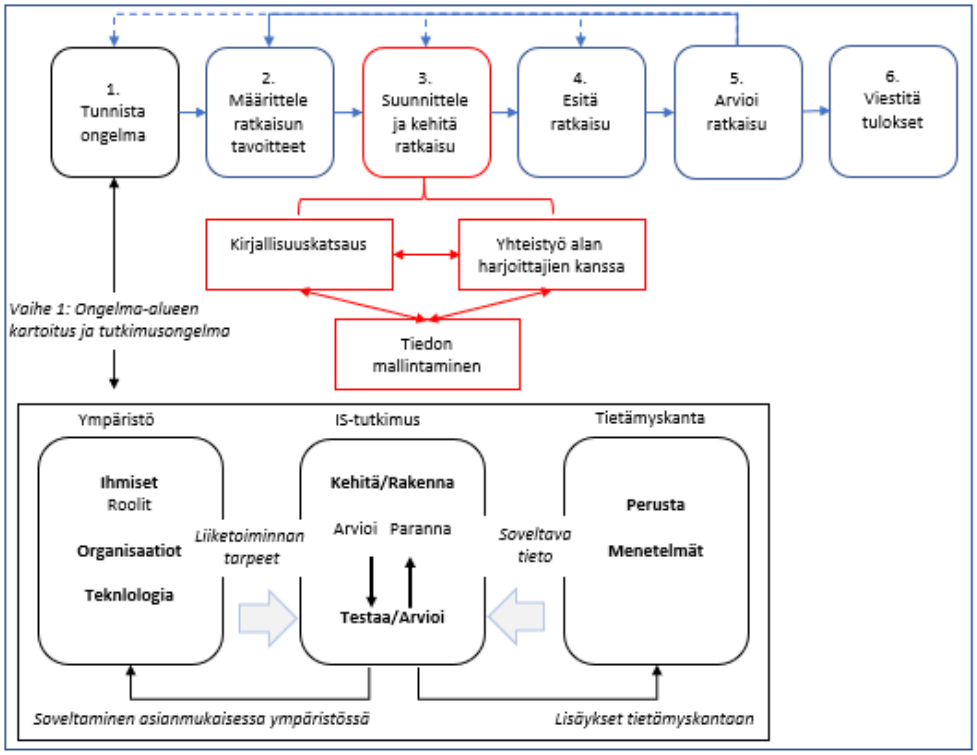
Peffers ym. (2007) ovat kehittäneet suunnittelutieteen tutkimusmetodologian, jonka avulla voidaan kehittää ja tuottaa tietojärjestelmien tutkimusta suunnittelutieteen avulla. Suunnittelutieteen menetelmän prosessi sisältää kuusi vaihetta, joita ovat ongelman tunnistaminen ja motivaatio, ratkaisun tavoitteiden määrittely, suunnittelu ja kehittäminen, esittely, arviointi ja viestintä. Suunnittelutieteellisessä menetelmässä käytettävän prosessin avulla luodaan tietotekninen artefakti, jonka avulla pyritään ratkaisemaan asetettu ongelma. (Peffers ym., 2007.) Suunnittelutieteelle hyvin ominainen piirre on teknologiasuuntautuneisuus ja siinä arvioidaan, saavutetaanko arvoa tai hyötyä kriteerejä vasten.

Ostrowskin ja Helfertin (2012) mukaan nykyisellään suunnittelutieteellinen tutkimus ei tarjoa johdonmukaisia ja ymmärtäviä vaiheita, jotka ohjaavat

tutkijoita tekniikoiden valinnassa. Ostrowski ja Helfert (2012) esittelevät viitekehyksen, joka käsittää artefaktien luontivaiheen eli meta-suunnittelun. Malli toimii tiedon keräämisen pohjana tuottaen ratkaisuja liiketoiminnan ongelmiin suunnittelutieteen kontekstissa. Yksi mallin osa-alueesta on kyky uudelleen esittää ja käyttää tietoa. Se heijastaa asiaan liittyvää tietämystä, joka perustuu ympäristön käsitteisiin ja suhteisiin. Tämä saavutetaan kirjallisuuskatsauksen, alan ammattilaisten yhteistyön ja tietojen mallintamisen avulla. (Ostrowski & Helfert, 2012.)

Suunnittelutieteen metodologian näkemykset ja periaatteet vaihtelevat eri tutkijoiden välillä. Peffers ym. (2007) ja Hevner ym. (2004) toteavat, että suunnittelutieteen metodologiaa käytetään harvoin soveltamalla, mikä viittaa siihen, että olemassa olevat menetelmät eivät ole riittävän selkeitä tai soveltuvat puutteellisesti. Metodologian noudattamiseen tarvittavia toimintoja kuten menetelmiä, välineitä ja tekniikoita kuvataan vain lyhyesti. Käytännön tasolla suunnittelutieteen on oltava täsmällisempi ja esitettävä yksityiskohtaisempaa tietoa tutkijoille.

Tässä tutkimuksessa pyritään noudattamaan pääpiirteissään Peffersin ym. (2007), Hevnerin ym. (2004) sekä Ostrowskin ja Helfertin (2012) kuvattua rakennetta (kuvio 8). Siinä kuvataan suunnittelutieteen prosessimallin vaiheet sekä perustellaan valittuja prosesseja viitekehyksen kolmelle päätoimelle. Suunnittelutiede ja suunnittelutieteen prosessimalli sekä viitekehys soveltuvat tutkimukselle, koska tavoitteena on konstruktoida uutta todellisuuskuvaa tapausorganisaation problematiikkaan sekä tarkastella artefakteja, joilla pyritään saavuttamaan hyötyä organisaatiolle. Seuraavissa alaluvuissa käsitellään tarkemmin tämän tutkimuksen tutkimusmetodia ja sen vaiheita.



KUVIO 8 Suunnittelutieteen prosessimalli ja viitekehys (Peffersin ym., 2007, s. 54; Hevnerin ym., 2004, s. 80; Ostrowskin & Helfertin, 2012, s. 2 mukaan)

6.1.1 Ongelman tunnistaminen ja motivointi

Suunnittelutieteen prosessin ensimmäinen vaihe käsittää ongelman tunnistamisen ja motivoinnin. Vaiheen tarkoituksena on määritellä ja kuvata tutkimusongelma mahdollisimman tarkasti ja perustella ratkaisun (artefaktin) merkitys. Toteuttamiseksi tarvitaan tietoa ongelman tilasta ja ratkaisun tärkeydestä. (Peffers ym., 2007.) Ensimmäisessä vaiheessa voidaan hyödyntää Hevnerin ym. (2004) käsitteellistä viitekehystä IS-tutkimuksen ymmärtämistä, toteuttamista ja arviointia varten. Viitekehys kattaa kolme osa-aluetta, joita ovat ympäristö, IS-tutkimus ja tietämyskanta.

Ympäristö määrittelee ongelma- tai kohdealueen, jossa tutkittavat ilmiöt sijaitsevat. Se koostuu ihmisistä, organisaatioista ja olemassa olevista teknologioista. Liiketoiminnan tarpeita arvioidaan suhteessa organisaatioiden strategioihin, struktuuriin, kulttuuriin ja liiketoimintaprosesseihin. Liiketoiminnan tarpeet sijoitetaan nykyiseen teknologiainfrastruktuuriin, sovelluksiin, kommunikointiarkkitehtuureihin ja kehittämisen kyvykkyyksiin. Yhdessä nämä elementit määrittelevät liiketoiminnan tarpeet tai ongelmat. (Hevner ym., 2004.)

Tietämyskanta tarjoaa ainekset IS-tutkimuksen toteuttamiselle. Tietämyskanta koostuu perusteista ja menetelmistä. Aiemmat IS-tutkimukset ja vertailuaineistojen tulokset tarjoavat perusteet teorioista, viitekehyksistä, välineistä, käsitteistä, malleista, menetelmistä ja toteutuksista, joita käytetään tutkimustyön kehittämis- ja rakentamisvaiheissa. (Hevner ym., 2004.) Luvussa 6.3 on esitelty tämän vaiheen soveltaminen ja toteuttaminen tapausorganisaation tutkimuksessa.

6.1.2 Ratkaisun tavoitteiden määrittely

Suunnittelutieteen prosessin toinen vaihe käsittää ratkaisun tavoitteiden asettamisen. Ratkaisun tavoitteet voidaan päätellä ongelman määrittelystä. Myös tieto siitä, onko toteutus mahdollista ja järkevää, on olennaista. Tavoitteet voivat olla määrällisiä (esimerkiksi missä suhteessa haluttu ratkaisu on parempi kuin olemassa oleva) tai laadullisia (esimerkiksi kuvaus siitä kuinka uuden artefaktin odotetaan tukevan ratkaisua ongelmiin).

Tässä prosessin vaiheessa tarvitaan tietoa tunnistetusta ongelmasta ja aiemmista ratkaisuista sekä niiden soveltuvuudesta. Määriteltäessä tavoitteita, tulee selvittää artefaktiin kohdistuvat odotukset. (Peffers ym., 2007.) Prosessin toinen vaihe perustuu nykytilan tutkimiseen ja mahdollisimman hyvän tilannekuvan ymmärtämiseen. Nykytilan kuvaamisessa voidaan käyttää perinteisiä tutkimusmenetelmiä kuten havainnointia, haastatteluita tai saatavilla olevaa materiaalia. Luvussa 6.3 on esitelty tämän vaiheen soveltaminen ja toteuttaminen tapausorganisaation tutkimuksessa.

6.1.3 Ratkaisun suunnitteleminen ja kehittäminen

Suunnittelutieteen prosessin kolmatta vaihetta voidaan kutsua konstruktiiviseksi otteeksi ja tämä vaihe sisältää muutoksen suunnittelun ja toteutuksen eli artefaktin rakentamisen. Tällaiset artefaktit ovat mahdollisesti konstrukteja, malleja, menetelmiä tai toteutuksia. Tämä toteuttaminen edellyttää teoreettista tietoa ratkaisun tuottamiseen. Ostrowskin ja Helfertin (2012) mukaan artefaktin luontivaiheessa kerätään tietoa. Tietämys saavutetaan kirjallisuuskatsauksen, alan ammattilaisten yhteistyön ja tietojen mallintamisen avulla. Kirjallisuuskatsauksessa tarkastellaan nykyisen tiedon ja metodologisten lähestymistapojen kriittisiä kohtia tietyllä aihealueella. Tämä voidaan nähdä valmisteluna, tiedon keräämisenä tai perustan rakentamisena, johon artefakti rakennetaan.

Kohderyhmät, suorat havainnot ja haastattelut ovat yleisimpiä yhteistyötapoja. Artefaktin rakentaminen on elävä prosessi, jota harjoittavat alan ammattilaiset. Uudistuksen tai muutoksen toivotaan tuottavan hyötyä tai muuttavan tilannetta toivottuun suuntaan. (Peffers ym., 2007.) Tässä vaiheessa keskitytään käytettävissä oleviin ratkaisumalleihin ja miten ne soveltuvat organisaation problematiikkaan parhaiten. Ratkaisumallien analysointi ja kartoittaminen edellyttävät tutkijalta laajaa tietämystä ja tiedonhakua sekä taustojen selvittämistä. Luvussa 6.3 on esitelty tämän vaiheen soveltaminen ja toteuttaminen tapausorganisaation tutkimuksessa.

6.1.4 Esittely

Suunnittelutieteen prosessin neljäs vaihe on esittely, jossa artefaktia voidaan käyttää yhden tai useamman ongelman ratkaisemiseen. Tyypillisesti tämän käyttöön voi sisältyä simulointia, testaamista, tapaustutkimusta tai muuta vastaavaa toimintoa. Tässä vaiheessa edellytetään tietämystä, kun artefaktia hyödynnetään ongelman ratkaisuun. (Peffers ym., 2007.) Tällä mitataan sitä, onko artefaktin suunnittelussa onnistuttu tavoitteiden mukaisesti. Luvussa 6.3 on esitelty tämän vaiheen soveltaminen ja toteuttaminen tapausorganisaation tutkimuksessa.

6.1.5 Arviointi

Suunnittelutieteen prosessin viides vaihe käsittää arvioinnin, jossa mitataan, tukeeko luotu artefakti asetetun ongelman ratkaisua. Tässä vaiheessa verrataan aiemmin asetettuja ratkaisun tavoitteita artefaktin esittelyssä tuotettuihin tuloksiin. Ongelman tai artefaktin luonteesta johtuen arviointi voi sisältää artefaktin toiminnallisuuden vertailemista ratkaisun tavoitteisiin, määrällisiin suorituskykymittareihin kuten budjettiin tai tuotettaviin määriin, asiakastyytyväisyyteen tai palautteeseen. Arviointi voi sisältää myös määrällisiä mittareita järjestelmän suorituskyvystä kuten vasteajan tai saatavuuden. Hevner ym. (2004) ovat todenneet, että artefaktin arvioinnin mittarit voivat perustua esimerkiksi toiminnallisuuteen, tarkkuuteen, luotettavuuteen, käytettävyyteen tai organisaation

sopivuuteen. Tämän vaiheen lopussa päätetään, jatketaanko prosessin viimeiseen vaiheeseen vai palataanko prosessin kolmanteen vaiheeseen parantamaan artefaktia. Iteroinnilla voidaan parantaa artefaktia sekä lisätä siihen liittyvää tietämystä. (Peffers ym., 2007.) Luvussa 6.3 on esitelty tämän vaiheen soveltaminen ja toteuttaminen tapausorganisaation tutkimuksessa.

6.1.6 Viestintä

Suunnittelutieteen prosessin viimeisessä eli kuudennessa vaiheessa käsitellään tutkimusten tulokset. Ongelman ja sen tärkeyden lisäksi kommunikoidaan artefaktista, sen hyödyllisyydestä ja uutuudesta sekä suunnittelun perusteellisuudesta tarkoituksenmukaiselle yleisölle, ammattikunnalle tai tiedeyhteisölle. Hevnerin ym. (2004) mukaan suunnittelutieteellistä tutkimusta on esitettävä sekä teknologiasuuntautuneille että sopivalle kohdeyleisölle.

Teknologiasuuntautuneet yleisöt tarvitsevat riittävät yksityiskohdat, jotta kuvattu artefakti voidaan rakentaa ja käyttää sopivassa organisaatiokontekstissa. Tämä mahdollistaa, että alan ammattilaiset voivat hyödyntää tutkitun artefaktin hyötyjä ja rakentaa kumulatiivista tietopohjaa artefaktin tutkimisen laajentamiseksi ja arvioimiseksi. Tutkimustietojen viestittäminen ja jakaminen nähdään tärkeänä, jotta tietojärjestelmätutkimus voi kehittyä ja jalostua. Kommunikointi ja viestintä vaativat tietämystä sovellettavan tieteenalan käytänteistä. Tutkimuksen ja tutkimustulosten vertailtavuus säilyvät, kun viestinnässä noudatetaan yhtenäisiä käytäntöjä. (Hevner ym., 2004; Peffers ym., 2007.)

6.2 Tiedonkeruu

Tässä tutkimuksessa käytettiin laadullista tutkimusmenetelmää. Aineistokeruun menetelmiksi valikoituivat haastattelut ja havainnointi sekä tapausorganisaation materiaalit, joilla kerättiin kvalitatiivista tietoa toimintaympäristöstä ja tutkittavasta kohteesta. Teoriataustan kokoamisessa käytettiin tieteellisiä kirjallisuuslähteitä, joiden tarkoituksena oli hahmottaa tutkittavaa kohdetta käsitteleviä teorioita ja käsitteitä. Empiirisen osion tiedonkeruun menetelmiksi valikoituivat tässä tutkimuksessa ryhmän teemahaastattelut ja havainnointi. Näiden menetelmien avulla saatiin esille organisaatiossa työskentelevien kokemuksia tutkittavasta ilmiöstä. Ryhmän haastattelussa on tyypillisesti yhteinen mielenkiinnon kohde ja ryhmän tiedonkeruun etuna nähdään, että yhdellä keskustelukerralla pystytään haastattelemaan useita henkilöitä. DevOps-toimintamallia ei ole tutkittu IT-palvelutuotannon näkökulmasta ja tämän kannalta haastatteleminen ja havainnointi ovat sopivia tapoja tunnistaa yhtäläisyyksiä.

Tutkimuksen lähteinä on käytetty aiheesta kirjoitettuja englanninkielisiä tietojenkäsittelytieteiden ja ohjelmistotuotannon tieteellisiä julkaisuja ja kirjoja, joissa on hyödynnetty hakutermejä ”perinteinen ja ketterä IT” (engl. traditional

and agile IT), ”bimodaalinen IT”, ”kahden nopeuden IT” (engl. Bimodal IT, two-speed IT, double-speed-IT), “IT-palvelunhallinta” (engl. ITSM, IT Service Management), “ITIL”, “IT-palvelutuotanto” (engl. IT Service Operation) ja “DevOps”. Sähköiset lähteet, jotka on julkaistu tunnetuissa ja tietojärjestelmätieteen arvostetuissa julkaisufoorumeissa, muodostavat merkittävän osan tutkimuksen lähdekirjallisuudesta. Sähköisinä lähteinä on käytetty alan lehtiä (engl. journals) ja tieteellisissä konferensseissa julkaistuja tutkimusartikkeleita (engl. proceedings). Tässä tutkimussuunnitelmassa on hyödynnetty ohjelmistotuotannon (Software Engineering (SE)) alan lehtiä sekä tietojärjestelmätieteen lehtiä. Tiedonhankintaan on käytetty NELLI-portaalia, Google Scholar-, IEEE Xplore Digital Library- ja AMC Digital Library-tietokantoja sekä Jyväskylän yliopiston kirjaston kansainvälisen e-materiaalihakupalvelua (JYKDOC). Tässä tutkimuksessa kirjallisuuslähteiden valinnalle annettiin seuraavat kriteerit:

- kirjallisuuslähde oli kirjoitettu suomen- tai englanninkielellä
- kirjallisuuslähde vastasi tutkimuskysymyksiin tai tuotti tutkimuskysymyksiin tietoa
- kirjallisuuslähde on julkaistu vuosina 2000-2018
- kirjallisuuslähteen otsikossa, tiivistelmässä, johdannossa tai asiasanana ilmaistaan vähintään yhteen tämän tutkimuksen tutkimuskysymykseen, aiheeseen tai sisältöön
- kirjallisuuslähde oli vertaisarvioitu tai julkaistu lehdessä ja arvostetussa julkaisufoorumissa ja siihen on viitattu runsaasti

Tässä tutkimuksessa kirjallisuuslähteiden luotettavuutta pyrittiin varmistamaan vertaisarvioinneilla tai viittausmäärillä. Ilman luotettavuutta tutkimuksella ei ole tieteellistä arvoa eikä se anna tietoa. Asiaankuuluvuudella tai merkityksellisyydellä tarkoitetaan kirjallisuuskatsauksen hyötyä ja hyödyllisyyttä.

6.2.1 Havainnointi

Havainnointia eli observointia voidaan käyttää itsenäisesti tai haastattelun lisänä tai tukena. Sen avulla saadaan tietoa käyttäytyvätkö ihmiset siten kuin sanovat toimivansa tai miltä asiat näyttävät. Havainnoinnin etuna nähdään, että sen kautta saadaan suoraa ja välitöntä informaatiota yksilön, ryhmien ja organisaatioiden toiminnasta näiden luonnollisissa ympäristöissä. Havainnointi menetelmänä soveltuu hyvin tilanteisiin, jotka muuttuvat nopeasti tai tilanteet ovat vaikeasti ennakoitavissa. (Hirsjärvi, Remes & Sajavaara, 2004, 201-203.) Havainnointia on kritisoitu kontrolliefektin takia. Tällä tarkoitetaan sitä, että havainnoija voi läsnäolollaan häiritä tai jopa muuttaa tutkittavaa tilannetta. Tämän lisäksi tutkija voi sitoutua tutkittavaan ryhmään tai tilanteeseen emotionaalisesti ja siten heikentää tutkimuksen objektiivisuutta. (Hirsjärvi ym., 2004, 202-203.)

Havainnointi voi olla suoraa tai osallistuvaa havainnointia. Suorassa havainnoinnissa tutkija tarkkailee tutkittavaa tilannetta ilman, että tutkittavat huomaavat tutkijan läsnäoloa. Suoraa havainnointia käytetään esimerkiksi eri-

laisten fysikaalisten tai teknisten ilmiöiden havainnoinnissa. Osallistuva havainnointi voi olla luonteeltaan aktiivista tai passiivista. Ensin mainitussa tutkija vaikuttaa läsnäolollaan aktiivisesti tutkittavaan ilmiöön esimerkiksi organisoimalla ryhmien toimintaa tietyllä tavalla. Jälkimmäisessä eli passiivisessa havainnoinnissa tutkija on mukana samanlaisena osallistujana tutkittavassa kohteessa siinä kuin muutkin vaikuttamatta tilanteen kulkuun. Osallistuva havainnointi tarkoittaa sitä, että tutkija on fyysisesti mukana tutkimuskohteessa. Havainnoinnissa on huomioitavaa, että tutkija ei välttämättä aina varmuudella tiedä, mihin kysymyksiin hän saa vastauksia. Tutkimuksen kohteena olevassa ympäristössä on oltava mukana, jotta tutkimusaineistosta tulee validi. (Anttila 2006, 20, 22-24.)

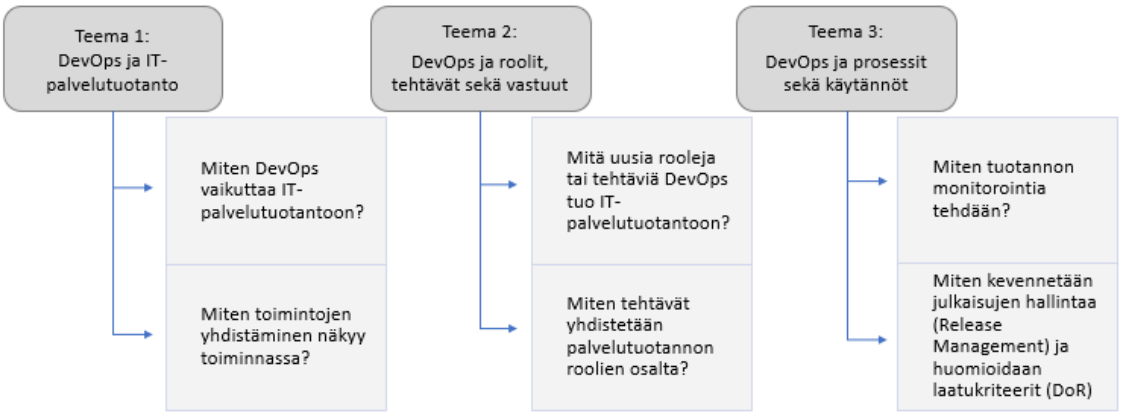
6.2.2 Teemahaastattelu

Teemahaastattelua käytetään erityisesti tilanteissa, joissa halutaan selvittää vähän tunnettuja tai tiedettyjä asioita tai vastaus perustuu haastateltavien henkilöiden omiin kokemuksiin (Hirsjärvi & Hurme, 2000, 35; Metsämuuronen, 2005, 226). Teemahaastattelu on luonteeltaan puolistrukturoitu haastattelumenetelmä. Puolistrukturoidulle haastattelulle on tyypillistä, että haastattelun näkökohta on joltain osin kiinnitetty ja haastattelu kohdistetaan määriteltyihin aihepiireihin. Teemahaastattelun avulla voidaan tutkia yksilöiden kokemuksia ja tuntemuksia. (Hirsjärvi & Hurme, 2000, 47-48.)

Ominaista teemahaastattelussa on, että haastateltavilla on samankaltaisia kokemuksia tilanteista (Hirsjärvi & Hurme, 2000, 47-48). Haastattelussa ei käytetä yksityiskohtaista kysymysluetteloa, vaan haastattelu toteutetaan aiheiden tai aiheluetteloiden avulla. Näiden pohjalta haastattelua voidaan jatkaa ja syventää. Strukturoituun haastatteluun nähden teemahaastattelu on luonteeltaan vapaamuotoisempi keskustelu. Haastattelulle on tyypillistä, että tutkijan lisäksi tutkittava toimii myös omilla vastauksillaan haastattelussa tarkentajana ja syventää teema-aluetta. (Hirsjärvi & Hurme, 2000, 66-67.) Haastateltavat voivat vapaasti kommentoida tai täydentää sekä tehdä huomioita tutkittavasta ilmiöstä. Tämän lisäksi haastattelussa voidaan kohdistaa kysymyksiä koko ryhmälle tai ryhmän jäsenille.

6.2.3 Haastatteluteemat

Tässä tutkimuksessa teemahaastattelun tiedonkeruun välineeksi luotu haastattelurunko koostui kolmesta teemakokonaisuudesta (kuvio 9). Teemat on pääosin luotu suoraan tutkittavien ilmiöiden, DevOpsin ja IT-palvelutuotannon pohjalta. Teemahaastattelun runko on kuvattu liitteessä 1. Haastattelun teemat käytiin läpi pilotin aikana. Tarkoituksena oli myös selvittää, miten tieteellisestä kirjallisuudesta löydetyt havainnot ilmenevät haastateltavan organisaation problematiikassa ja vahvistaako ne lähdekirjallisuudessa olevia käsityksiä.



KUVIO 9 Haastatteluteemat

6.3 Suunnittelutieteen käyttö tutkimuksessa

Tässä tutkimuksessa noudatettiin suunnittelutieteen (Design Science) -prosessia, jossa tuloksena syntyi suunnitteluperiaatteita. Tutkimusprosessi aloitettiin problematiikan tunnistamisella. Tämän tutkimuksen tapausorganisaation antamien tietojen pohjalta määriteltiin ongelma, johon lähdettiin tunnistamaan vaikuttavia tekijöitä ja ratkaisuvaihtoehtoja. Annettuihin tietoihin perustuen määriteltiin varsinainen tutkimusongelma. Määrittelyssä pyrittiin huomiomaan tapausorganisaation erityiset painopisteet ja niiden sovittaminen tieteelliseen kontekstiin. Tutkimusongelma on esitelty johdantoluvussa.

Tutkimusongelman määrittelyn jälleen määriteltiin tavoitteet. Tapausorganisaation antamien tavoitteiden pohjalta tutustuttiin tutkimusongelmaan, tutkimusongelmaan liittyviin teorioihin ja taustamateriaaliin. Tapausorganisaation tilanteen ja tunnistettujen ilmiöiden perusteella tutkimukseen valittiin teoria-aineistoa, johon tutkimuskysymykset pohjautuvat. Tutkimuskysymykset on esitelty johdantoluvussa.

Tutkimuskysymyksien ja tutkimuksen taustamateriaalin perusteella muodostettiin teorioita yhdistävää synteesiä, jonka tavoitteena oli antaa lähtökohdat tutkimuskysymyksien sisällölle. Kohteena oleva viitekehys tutkimusongelman ratkaisemiseksi tapahtui teoriaosuuden avulla. Teoreettiset lähtökohdat on esitelty luvuissa kaksi, kolme ja neljä.

Muodostettua artefaktia demonstroitiin ja testattiin tapausorganisaatiossa. Tavoitteena oli selvittää ja testata artefaktin soveltuvuus ja toimivuus. Demonstraatiosta saadut palautteet analysoitiin ja muodostettiin uudet käsitykset artefaktin toimivuudesta ja rakenteesta. Artefaktin kehittämiseksi siirryttiin takaisin tavoitteiden määrittelyvaiheeseen ja selvitettiin, kykeneekö artefakti edelleen vastaamaan tutkimuskysymyksiin teoriaan pohjautuen.

Lopuksi suoritettiin artefaktia koskeva arviointi. Tässä käsiteltiin, miten hyvin artefakti eli syntyneet suunnitteluperiaatteet ratkaisevat tutkimuksen alussa esitetyn tutkimusongelman ja vastaavat tutkimuskysymyksiin. Tulokset on esitelty luvussa seitsemän. Tutkimusprosessin viimeisessä vaiheessa kom-

munikoidaan tulokset (suunnitteluperiaatteet) tapausorganisaation ja tieteellisen yhteisön kanssa.

Suunnittelutiede ratkaisee tutkimusongelman rakennettujen ja arvioitujen artefaktien avulla, jotka on suunniteltu kohtaamaan liiketoiminnan tarpeet. Suunnittelutieteen tutkimuksen tavoitteena on hyödyllisyys. Tutkimustyön tarkastus ja arviointi voi johtaa teorian tai artefaktien heikkouksien tunnistamiseen sekä parannustarpeisiin ja uudelleenarviointiin. (Hevner ym., 2004.)

7 ARVIOINTI JA TULOKSET

Uuden toimintamallin toteuttaminen organisaatiossa edellyttää aina muutosta nykyiseen toimintaan. Muutoksen onnistunut läpivienti edellyttää IT-palvelutuotannon ja DevOpsin yhteistoiminnan ohjaamista kehittymään haluttuun suuntaan. Tässä tutkimuksessa käytettyjen lähteiden näkökulmista vaikuttaisi, että uudet toimintaympäristöt, teknologia ja työvälineet sekä uudenlaiset palvelut ja järjestelmät mahdollistavat myös toisenlaisen toiminnan. Toisinaan muutokset aiheuttavat merkittäviä muutoksia ja häiriöitä organisaatiotasolla kuten kuviosta 3 ilmenee (Haffke ym., 2017b).

Tässä tutkimuksessa tutkitaan, mitä kirjallisuuden perusteella on löydettävissä IT-palvelutuotannon ja DevOpsin yhdistämiseen aiemmista tutkimuksista ja tarjoavatko olemassa olevat mallit vastauksia tähän problematiikkaan. Aihetta lähestytään IT-palvelutuotannon näkökulmasta. Tutkimuksessa käsitellään ryhmähaastatteluista kerättyä aineistoa ja sen tuloksia. Ensimmäisessä alaluvussa kuvataan lyhyesti haastateltavat. Toisessa alaluvussa kuvataan ratkaisun arviointia. Kolmannessa alaluvussa analysoidaan haastattelu- ja havainnointiaineistoa ja viimeisessä alaluvussa keskitytään ratkaisun tuloksiin eli syntyneisiin suunnitteluperiaatteisiin ja tapausorganisaation haastateltavien antamaan kuvaan DevOps-toimintamallista ja sen käytänteistä.

7.1 Haastateltavat ja ryhmän teemahaastattelut

Tapausorganisaatio edustaa rahoitustoimialaa ja organisaation henkilöstöstä kohderyhmäksi valikoitui DevOps-pilottiin osallistuneet henkilöt. Pilotin henkilöistä seitsemän edusti tapausorganisaation palvelutuotantoa ja kaksi kehittämisen organisaatiota. Pilotin edetessä ryhmähaastatteluita järjestettiin lähes viikoittain. Tutkimuksessa puolen tunnin mittaisten haastattelujen aikana oli mahdollista kuulla useamman henkilön näkemyksiä. Haastattelut ja keskustelut sijoittuivat neljän kuukauden ajalle, maaliskuusta kesäkuuhun. Yhteenveto ryhmähaastattelun henkilöistä ja taustoista on kuvattu taulukossa 3.

TAULUKKO 3 Yhteenveto ryhmähaastattelun henkilöistä

Henkilö	Rooli	Tehtäväkuva pilotissa	Näkökulma
H1	Palvelutuotantopäällikkö	DevOpsin vaikutusten ja rajoitteiden tunnistaminen, DevOps-mallin käytäntöjen testaaminen ja iterointi omalla vastuualueella.	Palvelunhallinta, palvelutuotanto, Ops
H2	Palvelutuotantopäällikkö	DevOpsin vaikutusten ja rajoitteiden tunnistaminen, DevOps-mallin käytäntöjen testaaminen ja iterointi omalla vastuualueella.	Palvelunhallinta, palvelutuotanto, Ops
H3	Palvelutuotantopäällikkö	DevOpsin vaikutusten ja rajoitteiden tunnistaminen, DevOps-mallin käytäntöjen testaaminen ja iterointi omalla vastuualueella.	Palvelunhallinta, palvelutuotanto, Ops
H4	Palvelutuotantopäällikkö	DevOpsin vaikutusten ja rajoitteiden tunnistaminen.	Palvelunhallinta, palveluntuotanto, Ops
H5	Palvelutuotantopäällikkö	DevOpsin vaikutusten ja rajoitteiden tunnistaminen.	Palvelunhallinta, palveluntuotanto, Ops
H6	Julkaisupäällikkö	DevOpsin vaikutusten ja rajoitteiden tunnistaminen, DevOps-mallin käytäntöjen testaaminen ja iterointi omalla vastuualueella.	Dev+Ops
H7	Palvelutuotantopäällikkö	DevOpsin vaikutusten ja rajoitteiden tunnistaminen, DevOps-mallin käytäntöjen testaaminen ja iterointi omalla vastuualueella.	Dev+Ops
H8	Migraatiopäällikkö	DevOpsin vaikutusten ja rajoitteiden tunnistaminen.	Palvelunhallinta, palvelutuotanto, Dev+Ops
H9	Migraatiopäällikkö	DevOpsin vaikutusten ja rajoitteiden tunnistaminen.	Palvelunhallinta, palvelutuotanto, Dev+Ops

Kuviossa 9 määriteltyjä haastatteluteemoja käsiteltiin useaan kertaan ryhmähaastatteluissa pilotin aikana. Asiantuntijoiden käsitykset DevOpsin käytäntöjen ja periaatteiden hyödyntämisestä jalostuivat pilotin edetessä sitä mukaa kun tietämys aiheeseen lisääntyi. Haastattelun ensimmäisessä teemassa keskusteltiin yleisesti ketterästä DevOps-toimintamallista ja palvelutuotannosta. Teemassa pohdittiin DevOpsin vaikutuksia IT-palvelutuotantoon ja DevOpsin operoinnin (Ops) yhdistämistä palvelutuotantoon. DevOpsin ajattelumallin ja käytänteiden selvittämisen kautta voidaan ymmärtää toimintamallia paremmin käytännössä sekä vahvistaa tutkimuskirjallisuudesta saatuja oletettuja käsityksiä.

Haastattelun toisessa teemassa käsiteltiin DevOps-toimintamallin rooleja painottuen palvelutuotannon näkökulmaan. Lisäksi pohdittiin ja verrattiin Ops-alueen rooleja ja tehtäviä nykyisen toimintamallin rooleihin ja tehtäviin. Roolien osalta keskityttiin ensisijaisesti Reliability Engineerin ja palvelutuotantopäällikön työnkuviin. Haastattelun kolmannessa teemassa keskityttiin rajattuun osaan DevOpsin prosesseista ja käytännöistä. DevOpsissa tuotantoympäristön monitorointi korostuu keskeisenä tehtävänä, josta kirjallisuuslähteiden mukaan vastaa tuotantotiimit. Viimeisenä teemana käsiteltiin julkaisuvaiheen keventämistä ketteräksi ja siihen liittyviä julkaisun kriteeristöjä. Teemahaastatteluissa esille tulleet tutkimusalueen ulkopuoliset asiat rajattiin tästä tutkimuksesta.

7.2 Ratkaisun arviointi

Tässä tutkimuksessa esitetään teoriaosuuden ja empiirisen tutkimuksen avulla tuotettu konstruktio eli lopputuotos ja arvioidaan sen toimivuutta. Tuotetun ratkaisun arviointi ja käytännön toimivuuden kokeilu tai testaus sisältyvät olennaisena osana suunnittelutieteelliseen tutkimusprosessiin. Tapausorganisaation tarpeisiin kehitetään toimivaa DevOps-toimintamallia ja tässä tutkimuksessa tuotetaan suunnitteluperiaatteita toimintamallia varten sekä hyödynnetään kirjallisuudesta löytyvää tietoa, joita ovat teoreettinen tieto, aiemmat tutkimukset ja niiden saamat tulokset sekä käytännön menetelmätietous. Suositusten eli suunnitteluperiaatteiden pohjalta toimintamallia jatkokehitetään tapausorganisaation määriteltyihin vaatimuksiin soveltuvaksi, ottamaan huomioon myös IT-operaatioiden jatkuvan hallinnan ja liiketoiminnan toimintakyvyn säilyttämisen edellyttämät jatkuvat palvelut 'devaamisen' ohella. Tässä tutkimuksessa DevOpsiin liittyviä toimintatapoja ja periaatteita testattiin kevyesti sekä huomioitiin nykyiset palvelutuotannon ja tapausorganisaation vaatimukset mallin toimivuuden arvioimiseksi ja päätöksentekoa varten.

Tutkimuksen luvussa 6.1 kuvatun suunnittelutieteen prosessin mukaan tässä luvussa tapahtuva konstruktioiden tuottaminen aiemman tutkimustiedon pohjalta liikkuu tutkimusprosessin suunnittelu- ja kehitys-, demonstraatio-, sekä arviointivaiheissa (esitelty luvuissa 6.1.3, 6.1.4 ja 6.1.5). Suunnittelutieteen prosessin kahden ensimmäisen vaiheen ongelman tunnistaminen ja motivointi (esitelty luvussa 6.1.1) sekä tavoitteen määrittely (esitelty luvussa 6.1.2) suoritettiin tutkimustyön alkuvaiheessa, jolloin tulevaa pro gradu -tutkielmaa alettiin suunnittelemaan tutkimussuunnitelman ja tutkimusraportin muodoissa. Tutkimussuunnitelman tarkoituksena oli toimia alkuvaiheessa tutkimusprosessissa alustavana aihiona, jonka perusteella esiteltiin aihealuetta ja tulevaa tutkimustyötä. Myöhemmässä vaiheessa laadittu tutkimusraportti esitteli tutkimusalueeseen liittyvää teoriatietoutta tutkimussuunnitelmaa laajemmin kirjallisuuskatsauksen muodossa, jossa esiteltiin IT-palvelunhallintaa, bimodaalista IT:tä ja ketterää DevOpsia koskeva teoriaosuus.

Tutkimuksessa muodostettiin käsitys eli konstruktio tapausorganisaation IT-palvelunhallinnan ja DevOpsin yhdistämisestä ennen ja jälkeen DevOpsin hyödyntämistä ja käyttöönottoa. Tutkimuksessa tarkasteltava tilanne sisältää IT-palvelunhallinnan, jatkuvien palvelujen ja DevOpsin alueilla yhteensovittamiseen liittyvien asioiden tunnistamisen ja mahdolliset rajoitteet. Tämän lisäksi tarkasteltiin, kuinka DevOpsin ja IT-palvelutuotannon roolit ja ohjelmistojulkaisuhallinnan laatukriteeristöt pitäisi hoitaa jatkuvien palveluiden näkökulmasta. Rooli- ja tehtäväkuvausten avulla pyrittiin osoittamaan DevOpsin vaatimat muutokset sekä mahdolliset hyödyt tapausorganisaatiolle. Rooli- ja tehtäväkuvaukset muodostettiin teemahaastattelujen perusteella saavutetuista tutkimustuloksista.

Tämän tutkimuksen konstruktioiden rakentumisen merkeissä rooleja ja tehtäviä sekä ohjelmistojulkaisuhallinnan laatukriteeristöjä iteroitiin suunnitte-

lu- ja kehitysvaiheesta arviointivaiheeseen. Iteroinnin tuloksena päätettiin suunnitteluperiaatteista, kuinka roolit ja laatukriteeristöt tulisi toteuttaa tapausorganisaatiossa. Tiedon mallinnus suoritettiin luvussa 6.2.3 kuvattujen teemahaastattelujen perusteella. Teemahaastattelussa keskityttiin selvittämään käytännön toimintatapoja, joita DevOps-mallissa ja tehtävärooleissa tarvitaan.

7.2.1 Tilanne ennen DevOps-pilottia

Tässä alaluvussa esitetään karkealla tasolla nykytila-analyysin avulla niitä menetelmiä ja toimintatapoja, joita toteutettiin tapausorganisaatiossa ennen ketterän DevOps-toimintamallin käyttöönottoa. Nykyisessä toimintatavassa korostuvat erilliset organisaation toiminnot. Siiloutuminen on esiintynyt erillisinä toiminnallisina funktioina kuten kehitys-, testaus- ja ylläpitotoiminnot. Nykyisellään kehityksen ja tuotannon väliset käytännöt ovat vaihdelleet. Toimintamallit sekä niiden noudattaminen ovat olleet epäyhtenäisiä.

Ohjelmistoversioiden julkaisujen osalta noudatetaan etukäteen sovittuja käytäntöjä. Kehittämisjunien ja hankkeiden aikaansaamat muutokset kootaan jakeluversioiksi. Muutosten järjestelmällisellä käsittelyllä varmistetaan, että jakeluversioissa käyttöönotettavat muutokset aiheuttavat mahdollisimman vähän virheitä, häiriöitä tai keskeytyksiä tuotannossa. Jakeluversiokokonaisuudet hallinnoidaan, testataan ja käyttöönotetaan. Ohjelmistoversioiden sisältö tiedotetaan ja ohjeistetaan loppukäyttäjille. Dokumentointi nähdään kehitysprosessiin olennaisesti kuuluvaksi tehtäväksi. Julkaisujen tuotantoon hyväksymisessä noudatetaan organisaatiossa noudatettavia julkaisujen ja käyttöönottojen laatukriteeristöjä. Kriteeristöjen avulla tarkistetaan, että versioiden käyttöönottoon liittyvät tarvittavat vaatimukset on täytetty ennen ohjelmistoversion tuotantoon siirtoa. Kriteeristöjen noudattamisen käytännöt ovat vaihdelleet eri palveluiden ja järjestelmien osalta.

Palvelutuotantopäällikkö vastaa palvelunhallinnasta kokonaisuudessaan. Tuotannon hoitamisessa ulkoisella palveluintegraattorilla on vahva sovelluksen tekninen vastuu häiriön- ja ongelmanhallinnan sekä palvelun teknisen elinkaarenhallinnan osalta. Uudet palvelut, jotka on toteutettu uusimmilla teknologioilla, noudattavat jo enenevissä määrin ketterän toimintamallin periaatteita. Olemassa olevissa tai jo pidemmän aikaa kehitetyissä järjestelmissä toiminta on perinteisemmän IT:n kaltaista.

7.2.2 Tilanne DevOps-pilotin jälkeen

Tässä alaluvussa esitetään tavoitetila-analyysin avulla niitä menetelmiä ja toimintatapoja, joita suositellaan toteutettavaksi ja käytettäväksi DevOps-toimintamallissa tapausorganisaatiossa. Tämän vaiheen suunnitteluperiaatteet on toteutettu testikokeilun perusteella. Pilotissa havaittiin, että DevOpsin kaltainen jatkuva toimitus ei tue tai edistä kaikkia palveluja tai järjestelmiä tarkoituksenmukaisella tavalla. DevOps soveltuu parhaiten palveluille tai järjestelmille, joita kehitetään jatkuvasti ja tehdään uudemmalla teknologialla. Järjes-

telmät, joita ei kehitetä aktiivisesti, toimivat paremmin nykyisissä toimintamalleissa. DevOps ei myöskään sovellu kaikkeen tekemiseen tai jatkuva julkaisu voi olla haasteellista toteuttaa pitkään kehitetyille monoliittisille järjestelmille. Käytetty arkkitehtuuri tai teknologia ei välttämättä tue automaatiota tarkoituksenmukaisella tavalla.

DevOpsiin liittyvä automaatio mahdollistaa jatkuvan julkaisun ja toistettavien, manuaalisesti tehtävien työvaiheiden minimoinnin. Monitoroinnilla voidaan valvoa järjestelmien, palvelujen ja palvelimien suorituskykyä sekä palauttaa tuotannon tilanne virheiden sattuessa. Automaation laajentumisen ja kattavan monitoroinnin avulla mahdolliset ongelmat, puutteet tai muutostarpeet voidaan havaita aikaisessa vaiheessa. Monitorointi auttaa myös palvelun toiminnan kehittämistä ja seurantaa pitkällä aikavälillä. Ohjelmistoversioiden pilkkominen pieniin julkaisuihin mahdollistaa uusien ominaisuuksien nopean tuotantoon viennin sekä toisaalta myös virheiden korjaamisen ketterämmin. Aikataulusta ja teknisistä valmiuksista johtuen monitorointia ei kyetty iteroimaan pilotin aikana

Kehityksen ja tuotannon yhteistyöllä pyritään tehokkaampaan ja avoimempaan yhteistyöhön. Kaikki työt viedään yhteiseen työjonoon ja priorisoidaan yhdessä kehittämisen ja tuotannon kesken. Tuotantoon luovutukset tapahtuvat joissakin palveluissa DevOps-tiimin sisällä. Tämä keventää käyttöönottoon liittyvää prosessia. Myös julkaisujen ja käyttöönottojen kriteeristöä kevennetään jakeluversioiden ja muutosten osalta sopivilta osin.

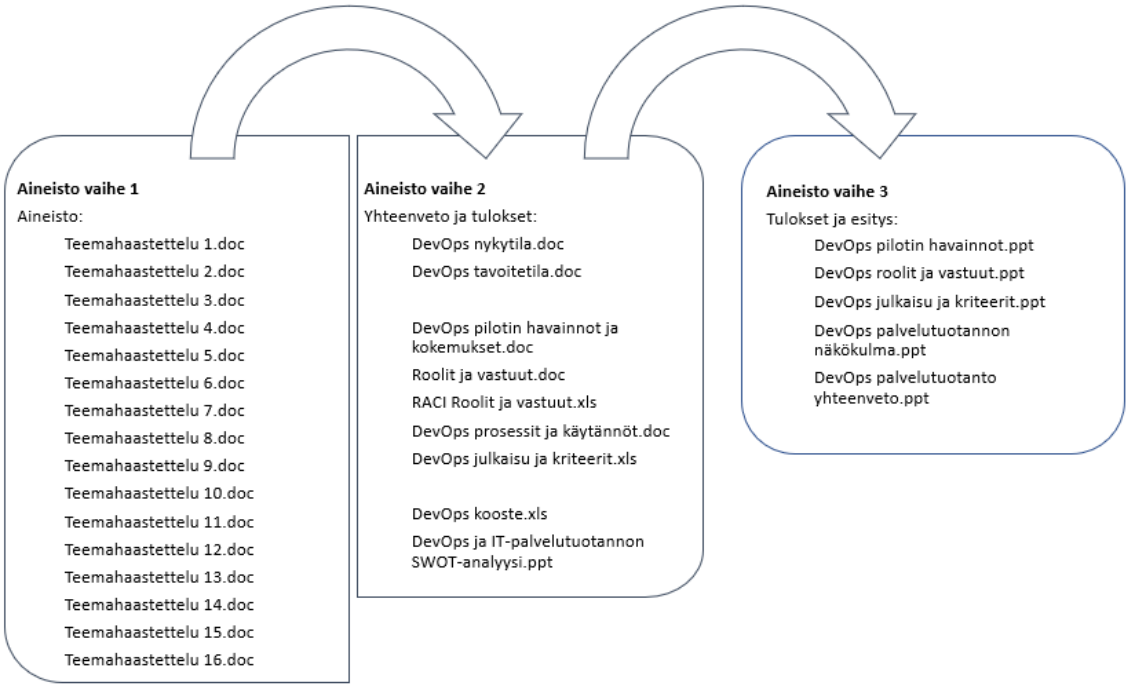
Reliability Engineerin rooli korostuu DevOps-toimintamallissa ja tämän osalta tarvitaan uudenlaista osaamista tai uudenlaisia rooleja. Tutkimuksessa havaittiin, että DevOps-toimintamallin Reliability Engineerin rooli vaikuttaa olevan täysipäiväinen työ kehittämisen ja tuotannon rajapinnassa. Tehtävässä tarvitaan vahvaa teknistä sovellusosaamista, jota nykyisellään palvelutuotantopäälliköllä ei välttämättä ole. Reliability Engineer osallistuu aktiivisesti päivittäin toimitusjunan toimintoihin. Myös ongelman ratkaiseminen ja tuotannon monitorointiin osallistuminen ovat tyypillisiä tehtäviä roolissa. Reliability Engineerin ja palvelutuotantopäällikön välinen työnjako ja sijoittuminen organisaatiossa ja toimittajaympäristössä olisi järkevää eriyttää, koska rooleja ei voi täysin yhdistää roolien luonteen vuoksi.

7.3 Haastattelu- ja havainnointiaineistojen analysointi

Konstruktiot mallinnettiin teemahaastattelun ja havainnoinnin sekä organisaation materiaalien pohjalta. Haastattelut suoritettiin ryhmälle teemahaastatteluina ja keskusteluina, joissa pyrittiin selvittämään tapausorganisaation problematiikkaa koskevia tekijöitä. Avoimia haastatteluja ei nauhoitettu vaan jokaisen haastattelun aikana tehtiin muistiinpanoja, jotka puhtaaksikirjoitettiin tekstidokumentteihin tekstinkäsittelyohjelmalla. Koska teemahaastattelussa tuli esille myös tutkimuksen ulkopuolelle jäävää tietoa, tehtiin referoivaa aineiston keruuta eli erotettiin aineistoista olennainen. Aineisto käsiteltiin puhtaaksikirjoit-

tamalla havainnot, kenttämuistiinpanot sekä teemahaastatteluiden tulokset, joiden perusteella haastattelujen kommenteista ja sisällöistä voitiin poimia olennaisimmat tekijät, jotka vaikuttavat IT-palvelutuotannon ja DevOpsin yhteensovittamiseen.

Kohderyhmän kesken toteutettiin useita teemahaastattelukierroksia. Haastatteluissa käsiteltiin DevOpsin ja IT-palvelutuotannon vaikutuksia toisiinsa sekä niiden yhdistämistä, rooleja ja vastuita sekä prosesseja ja käytäntöjä. Haastattelujen kestot olivat puolen tunnin pituisia. Haastatteluteemojen mukaisesti luotiin materiaalia erilaisia työvälineitä käyttäen kuten tekstinkäsittely-, taulukkolaskenta- ja esitysgrafiikkaohjelmilla. Aineistojen käsittely ja analysointi oli kolmivaiheinen (kuvio 10). Ensimmäisessä vaiheessa teemahaastattelun aineistot kirjoitettiin tekstidokumentteihin. Toisessa vaiheessa haastatteluaineistojen havainnot koottiin teemojen mukaisiin aineistoihin. Lisäksi haastattelujen sekä havaintojen perusteella koostettiin pilotissa mukana olleiden palvelujen osalta kuvaukset nykytilasta ja tavoitetilasta. Tutkimuksessa hyödynnettiin myös SWOT-nelikenttäanalyysia (engl. Strengths, Weaknesses, Opportunities, Threats), jonka avulla pyrittiin tunnistamaan DevOps-toimintamallin ja IT-palvelutuotannon välisiä uhkia, heikkouksia, vahvuuksia ja mahdollisuuksia. Kolmannessa vaiheessa aineistojen tulokset yhdistettiin ja esiteltiin tapausorganisaatiolle päätöksenteon tueksi.



KUVIO 10 Aineistojen analysoinnin eteneminen

Haasteltavien kommenteista pyrittiin löytämään yhtäläisyyksiä ongelmakohtien havaitsemiseksi. Tämän lisäksi pyrittiin tunnistamaan myös toimivia ja hyväksi havaittuja käytäntöjä ja toimintatapoja. Haastatteluiden pohjalta nousi selkeästi esille roolitukseen liittyvät vaatimukset. Päätöksenteon ja vastuiden määrittelyn tueksi otettiin tapausorganisaation nykyiset roolikuvaukset ja laa-

dittiin vastuumatriisi eli RACI. Tämän menetelmän avulla pyrittiin selvittämään eri tahojen välisiä keskinäisiä suhteita. DevOpsin roolia ja tehtäviä vertailtiin palvelutuotantopäällikön ja teknisen sovelluspäällikön rooleihin. Näiden analysoimiseksi vastuumatriisissa kuvattiin palvelunhallintaan ja jatkuviin palveluihin liittyvien tehtävien vastuita ja samalla tehtävien ja eri henkilöiden välisiä riippuvuussuhteita. Matriisissa kuvattiin roolien eri tehtävät ja tehtäviin liittyvät seuraavat käsitteet (taulukko 4). Vastuumatriisin avulla tunnistettiin roolien uudet sekä päällekkäin menevät vastuut ja tehtävät. Näiden avulla pystytään arvioimaan suuntaa, johon osaamisen kehittämistä tulisi lähteä kehittämään ja vahvistamaan ja miten roolit asettuvat tapausorganisaatiossa erilaisten palvelujen osalta.

Vapaa havainnointi tuotti aluksi runsaasti aineistoa, jota tuotettiin myös eri esitysmuodoissa. Kuvauksia tehtiin itse toiminnan ohella eri olosuhdetekijöistä kuten DevOps-tiimissä työskentelystä, Reliability Engineerin roolissa olemisesta, monitoroinnista tai tuotantoon julkaisujen prosesseista. Havainnot dokumentoitiin havaintojakson päätteeksi eli jokaisen tuotantoon viedyn julkaisun jälkeen, ja yhdistettiin tuotettujen aineistojen yhteenvetoihin ja tuloksiin.

TAULUKKO 4 Vastuumatriisin tehtäviin liittyvät käsitteet

Käsite	Kuvaus
A=Accountable	*taho joka on vastuussa siitä, että tehtävä tulee tehdyksi, delegoi tehtävän vastuullisille tekijöille, hyväksyy tehtävän*
R=Responsible	*taho, joka suorittaa tehtävän*
C=Consulted	*taho, joka auttaa tehtävän suorittamisessa ja tukee tehtävän suorittamista*
I=Informed	*taho, jota informoidaan tehtävään liittyen*

7.4 Ratkaisun tulokset

DevOps tähtää erityisesti yhteisosaamisellaan toimintavarman palvelun tai järjestelmän kehittämiseen ja ylläpitoon. DevOpsin tavoitteena on, että tehokkaan tiimityön ansioista ohjelmiston kehittämiseen osallistuvat osapuolet pyrkivät samaan lopputulokseen. Keskeisimpänä tavoitteena on saada kehitys- ja tuotantotiimien välinen yhteistyö toimimaan siten, että nopeasti tehtävät muutokset ja korjaukset saadaan julkaistua tuotantoon nopeasti ja luotettavasti palvelun tai järjestelmän laatua heikentämättä.

Tapausorganisaatiossa tunnistettiin tekijöitä, jotka vaikuttavat DevOpsin käyttöönottoon organisaatiossa. Tutkimuksen aikana tapausorganisaatiossa arvioitiin DevOpsin ja IT-palvelutuotannon rooleja ja tehtäviä, julkaisun ja käyttöönoton prosesseja, tuotantoon oton kriteeristöjä (ohjelmistojulkaisuhallinnan laatukriteeristöt) ja lisäksi arvioitiin kevyesti monitorointia. Tutkimuksessa havaittujen huomioiden perusteella voidaan todeta, että ketterä DevOps-toimintamalli soveltuu parhaiten palveluille tai järjestelmille, jotka ovat aktiivisen kehittämisen kohteena. Kaikilla palveluilla tai järjestelmillä ei välttämättä

ole jatkuvaa toimitusta. DevOps-toimintamallin hyödynnettävyys ja mahdollisuus täytyy erikseen arvioida järjestelmille tai palveluille, joita ei kehitetä säännöllisesti.

Tutkimuksessa tunnistettiin, että rooleissa on paljon yhtäläisyyksiä, mutta myös eroja tietyiltä osin. Merkittävimpänä eroina nousivat osaamiseen liittyvät tekijät. (Dev)Ops-tiimissä Reliability Engineeriltä edellytetään riittävän hyvää alueen substanssiosaamista. Henkilöllä on kyky priorisoida töitä kehityksen ja tuotannon työlistalla. Tärkeäksi tekijäksi nousee myös oman vastuualueensa kehitystoimien johtaminen, jolla parannetaan jatkuvasti tuotannon operatiivista toimintakykyä. Sovellusten ja palvelujen tilanteen ymmärtäminen, sovellusten arkkitehtuurien sekä tietokannan tilanteen tietämys tuotantoympäristössä on olennaista. Nämä osaamiset tukevat myös tuotannon monitoroinnissa tai virhelogien analysoinneissa. DevOpsissa työskennellään tiiviisti yhdessä ja mallin parhaisiin käytäntöihin kuuluu aktiivinen osallistuminen eri toimintoihin kuten esimerkiksi kehitysjunan päivittäisiin tapahtumiin. Näitä ovat muun muassa sprintin eli iteraation suunnittelu, katselmus ja retrospektiivi. Lisäksi palvelun tuotannon tuessa tai koordinoijan (palveluintegraattorin) roolissa oleminen korostuu DevOps-toimintamallissa.

DevOpsin käyttöönotto vaatii uudenlaista osaamista tai mahdollisesti uusia rooleja. Erityisesti teknisen osaamisen tarve, jota tarvitaan palveluiden tilanteen ymmärtämiseksi tuotantoympäristöissä ja tuotannon järjestelmien tai palveluiden monitoroinneissa, korostuu. Oikeanlaista osaamista ja käytäntöjä löytyy paikoitellen paikallisesti tai osaprosesseissa, mutta mahdollisten ulkoisten toimijoiden osaaminen ja yhteensovittaminen tulee huomioida ja arvioida tapauskohtaisesti. Tutkimuksessa havaittujen tulosten perusteella DevOps-tiimin Reliability Engineerin rooli vaikuttaa olevan täysipäiväinen työ kehittämisen ja tuotannon rajapinnassa. Reliability Engineerin ja palvelutuotantopäällikön välistä työnjakoa ja sijoittumista organisaatiossa ja toimittajaympäristössä suositellaan täsmennettävän. Vaikka rooleissa on samankaltaisuuksia, niin roolit ovat luonteeltaan erilaisia ja roolien yhdistämisessä nähdään haasteita. Reliability Engineer vastaa kokonaisvaltaisesti muutamasta sovellusalueesta. Palvelutuotantopäällikön vastuualue käsittää useita kymmeniä sovelluksia. Reliability Engineerin tehtävänä on ensisijaisesti varmistaa palvelun tai järjestelmän tekninen ympäristö sekä järjestelmän laatu ja toimintavarmuus. Palvelutuotantopäällikkö vastaa palvelunhallinnasta kokonaisuudessaan.

Tutkimuksessa tarkasteltiin myös julkaisujen ja käyttöönottojen laatukriteerien keventämistä kuitenkaan poistamatta olennaisia kriteerejä. Kevennettyä kriteeristöä testattiin useilla versiojulkaisuilla. Muutoksen kehittämisessä, testauksessa ja hyväksymisessä noudatetaan hyviä ohjelmistokehityksen periaatteita ja varmistetaan, että kehittämisen ja laadunvarmistuksen kontrollit ovat eriytettyjä. Julkaisun tarkistuslistan keventämisessä pohdittiin kriteereitä, jotka voidaan tehdä jo kehittämisen prosessin aikaisemmissa vaiheissa eikä prosessin lopussa. Ohjelmistoversioiden julkaisujen sisältö ja erityyppiset jakelut vaikuttavat myös kriteerien käyttämiseen. Pienet julkaisut (engl. minor releases) tarkoittavat uusien ohjelmistoversioiden toimittamista vähäisillä muutoksilla lop-

pukäyttäjien käyttöön nopeaan tahtiin. Suuret julkaisut (engl. major releases) sisältävät uusia toiminnallisia tai merkittäviä muutoksia loppukäyttäjille. Kolmantena julkaisuna on ohjelmiston hätämuutos, joka tarvitaan korjaamaan tuotannon tilanne silloin, kun kyseessä on laaja-alainen tai kriittinen häiriö palveluissa. Tällöin palautetaan tuotannon tilanne normaalille tasolle. Eri jakeluissa olevat muutokset testataan ennen tuotantoon siirtoa, mutta testauskäytännöt voivat olla kevyempiä tai dokumentointi voidaan tehdä jälkikäteen. Laatukriteerien käyttöön vaikuttavat, tehdäänkö tuotantoon luovutus ulkoiselle toimittajalle vai DevOps-tiimin sisällä. Tämä voi vaikuttaa palvelun dokumentoinnin tasoon (käyttöoppaat ja ohjeistukset). Merkittävät jakelut ja muutokset esitellään tuotannon CAB-palavereissa (engl. change advisory board). Jos muutokset ovat vähäisiä, niin esittely ei ole välttämättä tarpeen. Kuitenkin laatukriteerit, jotka liittyvät muutosten ja julkaisujen läpivientiin ovat olennaisia kuten muutosten testaaminen testauslausuntoineen, tekninen dokumentointi, versioiden jäädyttäminen, riskien tunnistaminen ja hyväksyntä riskienhallinnan osalta, julkaisujen aikatauluista sopiminen ja riippuvuuksien tarkistaminen, tuotantoon siirtoluvat ja käyttöönottosuunnitelma tuotantovalidoinnin toimenpiteillä.

DevOpsissa korostuu yhteistyön merkitys. Erityisesti eri organisaatioiden välinen tiivis yhteistyö on olennaisen tärkeää onnistumisen saavuttamiseksi. Siiloutuminen on tyypillisimmillään esiintynyt erillisinä toiminnallisina funktioina kuten kehitys-, testaus- ja ylläpitotoimintoina. Tekemistä ei voi optimoida tehtäväksi organisaatioittain. Nykyisellään toimintamallit ja niiden noudattaminen on ollut epäyhtenäistä. Ketteryys vaikuttaa kaikkeen tekemiseen ja koko organisaatioon. Yleiset ja selkeät tekemisen mallit ja virtuaalitiimit luovat perustan koko organisaation ketteryydelle ja mahdollistavat tehokkaan työskentelyn. Läpinäkyvyys ja yhdessä tekeminen korostuvat teemahaastattelujen tuloksissa. Tuotannon ja kehittämisen organisoituminen voi helposti johtaa siiloihin, joissa kehittämisen ja ylläpidon työt ovat omissa työjonoissaan ja prosesseissaan. DevOps-toimintamallissa korostetaan yhteistyötä ja vuoropuhelua. Ohjelmistojen parannus- ja muutostyöt viedään yhteiseen työjonoon ja priorisoidaan yhdessä kehittämisen ja tuotannon kanssa. Tapausorganisaation toimintamallissa virheet ja toimintavarmuus ovat etusijalla. DevOps-toimintamallissa kehittäjille syntyy parempi ymmärrys ja tilannekuva järjestelmän laadusta.

Joissakin tilanteissa DevOps saattaa koitua ongelmalliseksi. DevOpsin riskinä nähdään lisääntyvä työmäärä, joka kohdistuu tuotantotiimiin tai nykyisellään palvelutuotantopäällikön rooliin voimakkaasti rooleja mahdollisesti yhdistettäessä. Tiimin tai tuotannosta vastuussa olevan henkilön työmäärä lisääntyy, koska DevOpsissa työtehtävät, kokonaisvastuu, kehitystehtävien seuranta ja ohjaus sekä palaverit kuormittavat jo ennaltaan suuren työmäärän lisäksi. Ketterässä DevOps-toimintatavassa korostuu ohjelmistoversioiden tiheät päivitykset esimerkiksi viikon välein tehtävät julkaisut. Tämä voi näkyä myös lisääntyvänä työmääränä tuotannosta vastuullisen toimenkuvassa. Nykyisin tuotannosta vastuullisen henkilön vastuualueella voi olla kymmeniä järjestelmiä tai palveluja, johon Reliability Engineer -roolin sovittaminen toimii kokonaisuutena huonosti. Reliability Engineer nähdään dedikoituun alueeseen erikoistuneena

asiantuntijana, joka pystyy ottamaan täyden kokonaisvastuun ketterän tiimin Ops-alueen velvoitteistaan. Yhtenä riskinä nähdään myös DevOpsin hallittavuus, koska toimintamalli on suhteellisen uusi ja käyttökokemusta toiminnasta ei ole kertynyt tarpeeksi. Vaikka DevOpsissa on nähty paljon hyötyjä tietojärjestelmien kehittämisessä ja tuotannossa, ei kuitenkaan olla täysin varmoja sen toimivuudesta (Erich, Amrit & Daneva, 2014). Tutkimuksessa havaittujen tulosten perusteella voidaan todeta, että DevOpsin periaatteet ja käytänteet eivät ole toteutettavissa, jos parhaita käytäntöjä ja hyötyjä ei saavuteta.

Vaikka DevOps on suhteellisen uusi tapa toimia, se vaatii motivointia, hyväksymistä, johdon sitoutumista, resurssointia ja muutoksen johtamista. Monitorointi ja testausautomaatio parantavat palveluiden saatavuutta, suorituskykyä ja vakautta. Monitoroinnin ja automatisoinnin avulla potentiaaliset ongelmat havaitaan mahdollisimman aikaisessa vaiheessa. DevOpsin ketteryys mahdollistaa myös pienten julkaisujen viennit tuotantoon nopeasti. Positiivisina havaintoina nousivat, että pilotin aikana jo tehtyjen toimenpiteiden osalta on havaittu hyötyjä. Esimerkiksi isojen virheiden määrät ovat vähentyneet. Yhdessä työskentely parantaa yhteisyötä kehityksen ja tuotannon välillä ja kommunikaatio lisää myös näkyvyyttä. Palaute- ja korjausprosessi on nopeampaa aikaisempaan verrattuna. Negatiivisina havaintoina esille nousivat SAFen ylikorostuneisuus ja käyttäminen DevOpsissa. SAFe saattaa jossain määrin tuoda jäykkyyttä. Esimerkiksi kehittäminen on vaiheistettu eri ryhmille tai testaus erotetaan kehittämisestä. Olemassa oleva kulttuuri ja prosessit voivat hidastaa myös DevOpsin käyttöönottoa. DevOps on suhteellisen uusi toimintamalli ja kokemustaustaa toimintamallille ei ole riittävästi. Mahdollisuus nopeisiin julkaisuihin voi rohkaista myös hyväksymään ja kumuloimaan teknistä velkaa matkan varrella. Nopeat julkaisut voivat myös johtaa siihen, että tuotantoversioiden jäädyttäminen voi mennä viime hetkille, josta erityisesti testaus kärsii. Huolena nousivat myös jatkuvan julkaisun lisäämä työkuorma Reliability Engineerin roolissa.

8 POHDINTA

Tässä luvussa pohditaan suunnittelutieteellisestä tutkimuksesta ja teemahaastatteluista saatuja tuloksia sekä DevOpsin ja IT-palvelutuotannon yhdistämiseen liittyviä suunnitteluperiaatteita. Luvussa pohditaan DevOps-toimintamallia IT-palvelunhallinnan näkökulmasta sekä näiden aiheuttamia haasteita ja hyötyjä. DevOps hyödyntää ketterää Agile-Lean-lähestymistapaa ratkaisujen kehittämiseksi tai nopeuttamiseksi kehityksestä operointiin saavuttaakseen arvoa. Ketterät menetelmät riippuvat ihmisten, prosessien ja teknologioiden vuorovaikutuksesta ja yhteistyöstä. Konfigurointihallinnan, muutoshallinnan, julkaisu- ja käyttöönoton erityiset prosessit ovat tärkeässä roolissa ketterässä toimintaympäristössä. Kuten ITIL:issä, prosessien integrointi ja yhteensovittaminen auttavat edistämään ketteryyttä. Ketterien menetelmien menestys mitataan etenkin DevOpsissa lisääntyneiden toimitusten volyymien määrällä. Tätä mitataan parhaiten asiakastyytyväisyydellä, kun huomioidaan jatkuva toimittaminen asiakkaiden tarvitsemilla ratkaisuilla ja palveluilla. Kehittyneiden palveluratkaisujen jatkuvan toimituksen on oltava samankaltaisesti tasapainossa asiakkaan kyvyn kanssa omaksua hyötyä. Liian hitaasti toimitettavat toimitukset eivät vastaa asiakkaiden tarpeita tai liian nopeasti toimitettuja palveluita asiakkaat eivät voi hyödyntää. DevOps helpottaa jatkuvaa toimitusta ja jatkuvaa integraatiota.

Nykyisellään tapausorganisaation palvelutuotannon toimintatavassa korostuu palveluajattelu. IT-palvelun tuottamiseen liittyviä periaatteita ovat palveluiden tai järjestelmien saatavuuden ja vakauden turvaaminen, joilla varmistetaan perusedellytykset ja puitteet mahdollisimman korkealle palveluiden tai järjestelmien käyttöasteelle. Palveluajatteluun kuuluu läheisesti myös palvelun laadun arviointi, jossa kohtaavat tuotettavaan palveluun liittyvät odotusarvot. Kuten Iden ja Eikebrokk (2013) sekä Farrohan ja Farrohan (2014) määrittelevät, ITIL- ja DevOps-lähestymistapojen tarkoituksena on tukea laadukkaiden palvelujen toimittamista asiakkaille. DevOps-mallia tulisi toteuttaa huomioimalla IT-palvelunhallinnan tai ITIL:in parhaat käytännöt ja parhaat ominaisuudet olisi koordinoitava yhteistoiminnassa hyödynnettäväksi. Organisaatioiden tulee ymmärtää, että palvelut ovat määriteltyjä suhteita asiakkaiden ja palvelun toi-

mittajan välillä. DevOps, IT-palvelunhallinta ja ITIL auttavat parantamaan IT:n ja asiakkaan välistä suhdetta. Kurinalainen yhdessä työskentely auttaa jatkuvan palvelun parantamisessa ja organisaation suorituskyvyssä. DevOpsia kuvataan kehittämisen kannalta ja siinä korostetaan tuotantoon siirron ketterää ja toimivaa yhteistoimintaa. DevOpsin kehittämisen näkökulman sijaan olisi hyvä painottaa myös tuotannon lähtökohtia ja tarpeita.

Kuten muidenkin ketterien kehitystekniikoiden osalta on havaittu niin perinteinen vesiputousmalli ei ole yhteensopiva myöskään DevOpsin kanssa. Perinteisten tai vanhanmallisten lähestymistapojen paketoiminen uusien lähestymistapojen ympärille ei välttämättä toimi. Palvelun laadunvarmistuksen on varmistettava, että toiminnallisuudet eivät ole ristiriidassa uusien ominaisuuksien kanssa. Lisäksi on erityisesti varmistettava, että konfiguraatioon liittyvät asiat eivät vaikuta muihin palveluihin, jos käytössä on esimerkiksi yhteinen palvelualusta tai -ympäristö. Palveluiden tuki, tapahtumienhallinta ja palvelunvalvontatyökalut luottavat tietoihin, jotka ovat konfigurointihallinnan tietokannassa. Jos DevOps dynaamisesti muuttaa ympäristöä, täytyy myös varmistaa, että tarvittava informaatio on saatavissa ja jäljitettävissä CMDB:ssa. Tämä tarkoittaa sitä, että DevOpsin ja IT-palvelunhallinnan ja ITIL:in täytyy olla linjassa saavuttaakseen hyödyt.

Jotta esteet toimintamallin ja viitekehyksen välillä voidaan poistaa, niin paras lähestymistapa olisi toteuttaa ITIL käyttämällä DevOps-käytäntöjä. Tällä tavalla varmistettaisiin tilanne, jossa molemmat osapuolet hyötyvät. Nykypäivän vaativassa ja kompleksisessa IT-ympäristössä ei voi menestyä ilman vahvaa yhteistyötä. Tänä päivänä edellytyksenä on nopeasyklinen ja automatisoitu IT-ympäristö, jossa laatu, tehokkuus ja optimaaliset kustannukset kohtaavat. Palvelunhallinnan näkökulmasta DevOpsissa ja sen kanssa toimivissa viitekehyksissä tai standardeissa kuten esimerkiksi ITIL:issä, on olennaista keskittyä parempaan kommunikointiin ja yhteistyöhön, jatkuvan toimituksen toteuttamiseen, prosessien automatisoitiin, palautesilmukan vahvistamiseen ja prosessien parantamiseen sekä iteratiiviseen automatisointiin.

Kuten Virmani (2015) toteaa, DevOps on IT-alalla uusi ilmiö ja vaikuttaa organisaatiossa kaikkeen, myös osaamiseen, rooleihin ja työtehtäviin. Tutkimuksessa käytettyjen lähteiden mukaan, sama työntekijä voi hoitaa sekä kehittämiseen että tuotantoon liittyvät tehtävät. Työntekijät sitoutuvat työskentelemään samaa tavoitetta kohti ja kommunikaatio kehityksen ja tuotannon välillä paranee. DevOps on työskentelytapa, jonka avulla kehitys- ja tuotantotiimit työskentelevät tiiviissä yhteistyössä. DevOpsin työtehtävissä onnistumiseen vaaditaan tietynlaisia taitoja kuten teknistä ja toiminnallista osaamista sekä erinomaisia kommunikointitaitoja. Yhteistyö kehittäjien ja tuotannon välillä voi olla toisinaan haastavaa. Kehitys haluaa tehdä julkaisuja entistä useammin, kun vastaavasti tuotanto pyrkii turvaamaan tuotantoympäristön vakautta ja häiriöttömyyttä. Erilaiset näkemykset voivat johtaa ristiriitoihin. DevOps tunnistetaan enemmän ohjelmistokehityksen toimintatapana, jossa tavoitteena on kehittäjien ja tuotannon operoijien yhteistyön lähentäminen, korostaen automaatiota ja yhteistyötä.

DevOps-toimintamallia voidaan toteuttaa eri tavoilla. Tyypillisimmillään kehittäjätiimille varataan operointitiimiin henkilö, joka auttaa kehitystä infrastruktuuriongelmien ilmetessä. DevOpsin operoinnin alueella merkittävässä roolissa toimii Reliability Engineer, joka valvoo palvelua kehityksen jälkeisenä ajanjaksona ja toimii kontaktihenkilönä ongelmien ilmaantuessa. Roolissa työskentely voi edellyttää korkeaa tavoitettavuutta ja mahdollisesti työskentelyä tuotannon tuen keskitetyissä päivystysringeissä. Tämä edellyttää laaja-alaista moniosaamista tuotannossa olevista palveluista, jotta voidaan tunnistaa häiriöt ja osoittaa ne oikeille tahoille. Tunnistettujen vastuutehtävien näkökulmista DevOpsin käyttöönottaminen vaikuttaisi vaativan uudenlaista ajattelua, osaamista tai rooleja organisaatiossa sekä kykyä oppia uutta.

Tämän tutkimuksen tuloksena syntyi tapausorganisaation suosituksia eli niin kutsuttuja suunnitteluperiaatteita IT-palvelutuotannon ja DevOpsin yhteensovittamista varten. Olennaista on tunnistaa, että ketterä DevOps ei sovellu välttämättä kaikkeen tekemiseen. Tuotettavien palveluiden luonne, palveluiden ja järjestelmien aktiivinen kehittäminen sekä käytettävä teknologia ovat keskeisessä asemassa, kun pohditaan ketterän toimintamallin soveltuvuutta. Verkostoituminen ja yhteistyö ovat tärkeitä elementtejä palvelutuotannon ja DevOpsin yhteensovittamisessa. Kehityksen ja tuotannon tiivistä yhteistyötä tukeva kulttuuri ja aito vuoropuhelu nousevat esiin mahdollistajina niin kuin myös yhteisten tavoitteiden ja päämäärien asettaminen. Ketterässä toimintamallissa tuotetaan ohjelmistoversioita usein ja nopeasti, jolloin myös kommunikoinnin ja yhteystyön täytyy olla sujuvaa. Myös hyvin suunnitellun automaation avulla vähennetään manuaalisia työvaiheita ja sitä kautta virheitä. Tietyiltä osin joitain työvaiheita voidaan tehdä kevyemmillä prosesseilla heikentämättä tuotettavan palvelun laatua tai vaarantamatta palvelun turvallisuutta. Prosessien keventämisen kohteet tulee aina tapauskohtaisesti arvioida. Jatkuvan muutoksen aikakausi vaatii myös osaamisen päivittämistä ja mahdollisten osaamisvajeiden tunnistamista. Uuden toimintamallin käyttöönotossa on huomioitava uudet toimintatavat, uudenlainen osaaminen ja mahdollisesti uusien roolien tarve organisaatiossa.

Tässä tutkimuksessa tuotettu tieto ja syntyneet suunnitteluperiaatteet perustuvat yksittäisen tapausorganisaation spesifisiin vaatimuksiin. Tuotettua ratkaisua ei voi sellaisenaan hyödyntää, mutta tieto on joiltain osin yleistettävissä. Kuten tuloksista käy esille, DevOps-toimintamallia voidaan toteuttaa organisaatioissa eri tavoilla. Saatuja tuloksia voitaneen hyödyntää tiukasti säännellyillä toimialoilla tai kriittisissä toimintaympäristöissä, joissa korostuvat järjestelmien turvallisuus ja luotettavuus. Tämänkaltaisissa tapauksissa voidaan pohtia ketterän toimintamallin soveltuvuutta tai työvaiheiden keventämistä tuotettavissa palveluissa. Osaamiseen ja roolitukseen liittyvän tiedon osalta voi olla hyvin tapauskohtaisia muotoja ja tämän tutkimuksen tulokset eivät siten ole suoraan yleistettävissä tai hyödyntämiskelpoisia.

Tämä tutkimus on keskittynyt IT-palvelutuotannon näkökulmaan ja työstä on tarkoituksella rajattu ulos ohjelmistokehittämisen menestystekijät sekä DevOpsin kehittämiseen liittyvien teknisten osuuksien kuvaaminen. Tässä tut-

kimuksessa keskitytään tutkimaan ketterän toimintamallin soveltuvuutta tapausorganisaation palveluihin tai järjestelmiin. Prosessien osalta syvennyttiin yksinkertaistamaan ja keventämään julkaisun ja tuotantoon siirron ohjelmistojulkaisuhallinnan laatukriteeristöjä. Lisäksi IT-palvelutuotannon ja DevOpsin tehtävien ja osaamisen analysointi rajattiin koskemaan palvelutuotantopäällikön ja Reliability Engineerin rooleihin. Monitoroinnin osuus jäi pilotissa vähemmälle aikataulullisista ja teknisistä tekijöistä johtuen, joten aihetta on sivuttu hyvin rajallisesti tässä tutkimuksessa.

9 JOHTOPÄÄTÖKSET JA YHTEENVETO

Tänä päivänä IT-organisaatioiden täytyy reagoida nopeasti liiketoimintaympäristön muutoksiin ja jatkuvasti muuttuneisiin markkinoiden olosuhteisiin saavuttaakseen paremman kilpailuasetelman. Useimmille toimialoille ketterä toimintatapa on tärkeää tai jopa välttämätöntä. IT-organisaatioilla tulee olla kyvykkyyksiä korjata virheitä nopeasti, jotta uusia toiminnallisuuksia tai palveluja saadaan nopeammin markkinoille. Yrityksille on myös tärkeää päästä markkinoille ensimmäisten joukossa. Jos organisaatiorakenne on jäykkä tai hankala muuttaa, niin organisaatiolla on riski jäädä kehityksestä. Tämän lisäksi IT-organisaatioilla tulisi olla kykyä ylläpitää järjestelmien ja palvelujen vakautta, luotettavuutta, korkeaa käytettävyyttä ja laatua.

Tässä tutkimuksessa on tarkasteltu IT-palvelutuotannon ja DevOpsin yhdistämiseen liittyviä tekijöitä rahoitusalan tapausorganisaatiossa. Ilmiöiden välisiä vaikutuksia on tutkittu teoreettisten käsitteiden ja empiirisen tutkimuksen avulla. Tutkimuksen tavoitteena oli tarkastella DevOps-toimintamallia IT-palvelutuotannon näkökulmasta ja kuvata yhdistämisen hyödyt ja haasteet. Tutkimuksessa on selvitetty, mitä DevOps-toimintamallin käyttöönotto edellyttää palvelutuotannolta ja mitkä ovat DevOps-toimintamallin käyttöönoton vaikutukset IT-palvelutuotantoon. Tämän tutkimuksen tarkoituksena oli selvittää DevOps-toimintamallin yhdistäminen IT-palvelutuotantoon ja sen noudattamiin käytänteisiin. Tämän tutkimiseksi tutkimusongelma voidaan esittää tutkimuskysymyksillä seuraavasti:

1. Mitkä ovat DevOpsin ja IT-palvelutuotannon yhtäläisyydet ja erot sekä yhdistämisen haasteet?
2. Minkälaisia valmiita ratkaisumalleja löytyy kirjallisuudesta DevOpsin ja IT-palvelutuotannon yhdistämisestä?
3. Minkälaisia suunnitteluperiaatteita voidaan soveltaa ja hyödyntää ketterän DevOpsin ja IT-palvelutuotannon yhdistämisessä?

Tutkimuskysymysten asettelua on avattu luvuissa kaksi, kolme ja neljä selvittämällä ensin IT-palvelunhallinnan, DevOpsin sekä perinteisen ja ketterän IT:n

käsitteitä tieteellisen kirjallisuuden pohjalta. IT-palvelunhallintaa on kuvattu yleisellä tasolla sekä esitelty ITIL-mallia ja palvelun elinkaarimallin prosesseja. Bimodaalisen IT:n osalta on esitelty perinteisen ja ketterän IT:n eroja sekä erilaisia näkökulmia bimodaalisen IT:n hyödyntämiseen erityyppisissä organisaatioissa. Bimodaalisen IT:n vaikutuksia havainnollistettiin Haffken ym. (2017a; 2017b; 2017c) sekä Horlachin ym. (2017a; 2017b) kirjallisuustutkimusten valossa. Bimodaalinen suunnittelu pyrkii ratkaisemaan IT:n ristiriitaisia tavoitteita luotettavuuden ja vakauden osalta ja siten mahdollistamaan IT-toiminnon toimivan kahdessa rinnakkaisessa toimintatavassa - perinteisessä ja ketterässä. Ketterää DevOps-käsitettä esiteltiin palvelunhallinnan ja palvelutuotannon näkökulmista ja havainnollistettiin Fitzgeraldin ja Stolin (2017) sekä Kimin ym. (2014) esittämien lainalaisuuksien valossa. DevOpsin ja IT-palvelunhallinnan käytäntöjä ei tule nähdä vastakohtina vaan toisiaan täydentävinä ja tukevina. Molemmissa määräävänä tekijänä korostuu toimintavarmuuden ja laadun varmistaminen.

DevOpsin ja IT-palvelutuotannon yhteneväisyyksiä on ollut joissakin tapauksissa vaikeampi havaita ja tulkita. Tämän on todettu johtuvan siitä, että DevOps on tieteessä varsin tutkimaton ilmiö. DevOpsin ja palvelunhallinnan näkökulman yhdistävää tieteellistä tutkimusta on toteutettu varsin vähän. Tutkimustiedon puutteellisuuden vuoksi kirjallisuuteen perustuvaa analysointia on täydennetty konstruktiivisella suunnittelutieteellisellä tutkimuksella ja teemahaastatteluilla. Tutkimusmenetelmän kuvaus on tarkemmin esitetty luvussa kuusi. Tutkimuksen kannalta haastattelutulokset tukivat ja vahvistivat IT-palvelunhallinnan ja DevOps-tutkimuksen antamaa kuvaa hyödyistä, periaatteista ja käytänteistä. Tulokset selkiyttivät myös näkemystä DevOpsin ja palvelunhallinnan välillä. Tässä tutkimuksessa syntyneet suunnitteluperiaatteet tukevat IT-palvelutuotannon ja DevOpsin yhteensovittamista. Kehityksen ja tuotannon välisen yhteistyön lisääminen, julkaisuprosessien keventäminen, uudenlaisen osaamisen ja roolien tunnistaminen sekä DevOpsin käyttöönottaminen aktiivisesti kehitettävissä palveluissa ja järjestelmissä tuovat DevOpsia ja IT-palvelutuotantoa lähemmäksi toisiaan. Tulokset toivat teoreettisen tutkimuksen jatkoksi uutta tietämystä ilmiöiden välisistä vaikutuksista ja hyödyistä. Tarkempi analyysi on esitelty luvussa seitsemän.

Tutkimustulokset korostivat, että merkittävänä tekijänä DevOpsissa ja IT-palvelunhallinnassa on tuotannon toimintavarmuuden ja laadun varmistaminen. DevOpsin jatkuvassa integraatiossa ja julkaisussa hyötynä nähdään, että ohjelmistoa rakennetaan iteratiivisesti ja inkrementaalisesti toteuttamalla pieniä muutoksia mahdollisimman virheettömästi ja hallitusti nopeassa aikataulussa tuotantoon. Ohjelmiston laadun ja toimintavarmuuden nähdään paranevan, koska inkrementaalisesti tehdyt muutokset koskevat tyypillisesti tiettyä osaa ohjelmistossa. Pienissä jaksoissa tuotetut muutokset mahdollistavat myös palautteen saamisen nopeasti.

Tutkimustulokset tukivat ohjelmistotuotannon aiheuttavan siiloutumista tuotantoprosessiin, jota esiintyy erillisinä toiminnallisina funktioina kuten kehitys-, laadunvarmistus- ja ylläpitotoiminnot. Toimintojen ongelmat ja ristiriidat

konkretisoituvat selkeästi, koska tiimien tavoitteet ja päämäärät ovat erilaiset. DevOpsin etuna nähdään kehityksen ja tuotannon lähentyminen ja tiivis yhteistyö sekä vuoropuhelu. Keskeisimpänä tavoitteena on saada kehitys- ja tuotantotiimien välinen yhteistyö toimimaan siten, että nopeasti tehtävät muutokset ja korjaukset saadaan julkaistua tuotantoon nopeasti ja luotettavasti palvelun tai järjestelmän laatua heikentämättä.

Tutkimuksessa tunnistettiin, että DevOpsin roolit voivat poiketa merkittävästi perinteisen IT-palvelutuotannon rooleista tai tehtävistä. Tehtävät ovat tyypillisimmillään jakaantuneet useammalle henkilölle. Esimerkiksi palvelutuotantopäällikön, Reliability Engineerin ja teknisen sovelluspäällikön tehtävät eivät täysin kohtaa keskenään, vaan limittyvät tietyssä määrin. DevOpsissa korostuu tuotantoa hoitavien henkilöiden riittävän hyvä alueen liiketoiminta-osaaminen sekä tekninen tuntemus ja tietämys tuotantoympäristöistä, sovellusten arkkitehtuureista ja tietokantojen tilanteesta. Tieteelliset tutkimukset myös korostavat DevOpsin kehittäjien monipuolista roolia, koska heidän on tunnettava, miten järjestelmä on rakennettu ja kehitetty, mutta kehittäjien on keskityttävä myös järjestelmän toimintaan. Turvallisuuden, toimintavalmiuden, infrastruktuurin ja testauksen varmistaminen tulisi olla kehittäjien päivittäistä työtä.

DevOps hyödyntää ketterää lähestymistapaa ratkaisujen kehittämiseksi tai nopeuttamiseksi kehityksestä tuotantoon. DevOpsin tavoitteena on kasvattaa yrityksen arvoa laadukkaalla ja jatkuvalla palvelutoimituksella. Ketterät toimintamallit riippuvat ihmisten, prosessien ja teknologioiden vuorovaikutuksesta sekä yhteistyöstä. Näiden avulla pyritään kehittämään kehitys- ja tuotantotoimintojen välistä vuoropuhelua ja kyvykkyyttä tehdä tuotantoon muutoksia ketterästi sekä reagoimaan palautteeseen nopeasti. Tärkeässä roolissa ketterässä toimintaympäristössä ovat palvelutuotannon konfigurointihallinnan, muutoshallinnan, julkaisu- ja käyttöönoton erityiset prosessit. Näiden prosessien integrointi ja keventäminen auttavat edistämään ketteryyttä. IT-palvelunhallintaa ja ketterää DevOpsia ei tule asettaa vastakkain vaan nähdä ne toisiaan täydentävinä. Nykypäivän vaativassa ja kompleksisessa IT-ympäristössä ei voi menestyä ilman vahvaa yhteistyötä. Tänä päivänä edellytyksenä on nopeasyklinen ja automatisoitu IT-ympäristö, jossa laatu, tehokkuus ja optimaaliset kustannukset kohtaavat. Palvelunhallinnan näkökulmasta DevOpsissa ja IT-palvelunhallinnassa on olennaista keskittyä parempaan vuoropuheluun ja yhteistyöhön, jatkuvan toimituksen toteuttamiseen, palautesilmukan vahvistamiseen ja prosessien parantamiseen sekä iteratiiviseen automatisointiin. Korostamalla luottamusta, tukemalla innovaatiota ja kannustamalla yhteistyöhön pystytään rikkomaan esteet kehittämisen ja tuotannon väliltä.

Tämän tutkimuksen tavoitteena on ollut ymmärtää paremmin DevOps-toimintamallin taustamekanismeja ja haasteita, jotta voidaan tuottaa lähtökohtia ja suunnitteluperiaatteita palvelutuotannon huomioimiselle tapausorganisaatiossa. Vaikka tutkimuksen tarkastelun kohteena ovat olleet DevOps-pilottiin liittyvät huomiot, onnistumiset ja haasteet, tulokset auttavat myös vastaavien pilottien tai varsinaisten käyttöönottojen suunnittelemisessa, jalkauttamisessa ja toiminnassa. Tarkastelemalla suosituksia eli suunnitteluperiaatteita

voidaan todeta, että tutkimuksessa pystyttiin riittävällä tarkkuudella lähestymään tutkimusongelmaa ja vastaamaan tutkimuskysymyksiin. IT-organisaatioiden uudesta DevOps-ilmiöstä huolimatta, tulokset vetävät yhteen merkittäviä tekijöitä, joiden voidaan nähdä olevan yhteydessä tapausorganisaation DevOpsin ja IT-palvelutuotannon yhteensovittamiselle. Tutkimusta pystyttiin toteuttamaan valitun tutkimusprosessin mukaisesti.

DevOpsia voidaan pitää IT-organisaatioiden toimintatapana, jolla tehostetaan ohjelmistojen kehittämistä tehokkaan sisäisen kommunikoinnin avulla. DevOps sopii erityisesti organisaatioille, jotka kehittävät ja ylläpitävät järjestelmiään ja palveluitaan aktiivisesti. Vaikka DevOps on ohjelmistokehittämiseen suuntautunut, niin se asettuu myös IT-ylläpidon puolelle kattamaan järjestelmän tai palvelun elinkaaren kehittämisestä testaamiseen, julkaisuun, monitorointiin ja ylläpitoon. DevOps nopeuttaa ohjelmistojen julkaisemista, edesauttaa virheistä oppimista ja automatisoi manuaalisesti tehtäviä työvaiheita ja prosesseja. Nämä tekijät huomioiden DevOps tuo ketteryyttä myös ylläpidon ja julkaisun puolelle.

DevOpsin käyttöönotto ja hyödyntäminen suurissa organisaatioissa on vasta alkutekijöissään, mutta pienemmät yritykset toteuttavat ketterää toimintamallia tiedostamattaan. DevOpsin käyttöönotto ei ole virtaviivaisesti toteutettavissa organisaatioissa, joilla on monimutkaisia palvelutarpeita tai ketterän toimintamallin toteuttaminen voi vaatia monimutkaisia muutoksia yrityksen organisaatioiden rakenteisiin ja toimintaan. Bucena ja Kirikova (2017) korostavat, että DevOps-käyttöönoton tunnistamiseksi yritysten on ymmärrettävä ketterän lähestymistavan eri näkökohdat, strategia sekä riippuvuudet. Zhu, Bass ja Champlin-Scharff (2016) toteavat, että tyypillisesti DevOps-käytäntöjä sovelletaan organisaatioissa, jotka tarjoavat palveluja internetin kautta (esimerkiksi web-, pilvi- tai mobiilisovellukset). Tämän kaltaiset palveluntarjoajat julkaisevat uusia versioita päivittäin tai viikoittain ja ketteryys on edellytys jatkuvalle julkaisulle. Ketterän toimintamallin soveltaminen on ohjelmistoteollisuudessa vielä varsin harvinainen ilmiö myös alihankinnan ja ulkoistuksen osalta, mutta osittain kuitenkin toteutettavissa. Jatkotutkimuksissa voisi mahdollisesti tutkia DevOpsin sopivuutta ja parhaita käytäntöjä alihankinta- ja ulkoistuskumppaneiden välisessä yhteistyössä. Eri vaiheista vastaavat henkilöt eivät kuitenkaan välttämättä pysty organisaatiorajojen takia työskentelemään niin tiiviissä yhteistyössä kuin DevOps-kehityskulttuuri vaatisi. Tämän tutkimuksen valossa DevOpsin toteuttamisessa on huomioitava niin organisaation tarpeet ja kyvykkyydet mukautua ketterään toimintamalliin kuin myös organisaation tuottamat palvelut. Tämä tukee myös sitä käsitystä, että organisaatioiden voi olla vaikea käyttöönottaa ketterä toimintamalli, koska sen toteuttaminen voidaan tehdä organisaatioissa eri tavalla DevOpsin moniulotteisuudesta johtuen.

LÄHTEET

Agutter, C. (2012). *ITIL Foundation Essentials: The exam facts you need*. IT Governance Publishing. Haettu osoitteesta https://www.safaribooksonline.com

Agutter, C. (2017). *Service Integration and Management Foundation Body of Knowledge (SIAM Foundation Bok).* Van Haren Publishing. Zaltbommel.

Anttila, P. (2006). *Tutkiva toiminta ja teos, ilmaisu, tekeminen*. Hamina. Akatiimi.

Ayat, M., Sharifi, M., Sahibudin, S. & Ibrahim, S. (2009). Adoption Factors and Implementation Steps of ITSM in the Target Organizations. *Proceedings of Third Asia International Conference on Modeling & Simulation*. IEEE Computer Society, 369-374.

Babb, J., Nørbjerg, J., Yates, D. J. & Waguespack, L. J. (2017). The Empire Strikes Back. The end of Agile as we know it? *Communications of the Association for Information Systems. Selected Papers of the IRIS*. Issue 8, 43-59.

Balalaie, A., Heydarnoori, A. & Jamshidi, P. (2016). Microservices Architecture Enables DevOps: Migration to a Cloud-Native Architecture. *IEEE Software, 33(3),* 42-52.

Bass, L., Weber, I. & Zhu L. (2015). *DevOps: A softwar architect's perspective*. Addison-Wesley Professional.

Bernard, P. (2012). *Foundations of ITIL.* 2011 Edition. Van Haren Publishing.

BS, T. (2014). *Practical IT Service Management.* 2nd Edition. IT Governance Publishing. Haettu osoitteesta https://www.safaribooksonline.com

Bucena, I. & Kirikova M. (2017). *Simplifying the DevOps Adoption Process.* Riga Technical University. Haettu osoitteesta http://ceur-ws.org/Vol-1898/paper14.pdf

Cannon, D. & Wheeldon, D. (2011). *ITIL Version 3. Service Operation*. OGC.

Conboy, K., Coyle, S., Wang, X. & Pikkarainen, M. (2011). People over Process: Key Challenges in Agile Development. *IEEE Software, 28(4),* 48-57.

Cruz-Hinojosa, N. J. & Gutiérrez-de-Mesa, J. A. (2016). *Literature review of the situation research faces in the application of ITIL in small and medium enterprises*. Department of Computer Science. Spain: The University of Alcala, 124-138.

Daniels, R. & Davis, J. (2016). *Effective DevOps*. O'Reilly Media, Inc. Haettu osoitteesta https://www.safaribooksonline.com

Davies, J. (2016). *ITIL Foundation All-in-One Exam Guide*. McGraw-Hill. Haettu osoitteesta https://www.safaribooksonline.com

Ebert C., Gallardo G., Hernantes, J. & Serrano, N. (2016). DevOps. *IEEE Software, 33(3)*, 94-100.

Erich, F., Amrit, C. & Daneva, M. (2014). A Mapping Study on Cooperation between Information System Development and Operations. ProductFocused Software Process Improvement. *Proceedings of the 15th International Conference on Product-Focused Software Process Improvement, PROFES 2014, (277-280).* London: Springer Verlag.

Farroha, B. S. & Farroha, D. L. (2014). A Framework for Managing Mission Needs, Compliance, and Trust in the DevOps Environment. *IEEE Military Communications Conference*, Baltimore, MD, USA, 2014, 288-293.

Finanssivalvonta 2017. (2017). Pankit tarjoavat monenlaisia palveluja. Haettu 30.3.2018 osoitteesta http://www.finanssivalvonta.fi/fi/Finanssiasiakas/Finanssialan_palveluita/Pankkipalvelut/Pages/Default.aspx

Fitzgerald, B. & Stol, K-J. (2017). Continuous software engineering: A roadmap and agenda. *The Journal of Systems and Software, 123*, 176–189.

Galup, S. D., Dattero, R., Quan, J. J. & Conger, S. (2009). An Overview of IT Service Management. *Communications of the ACM, 52(5),* 124-127.

Gonzalez, D. (2017). *Implementing Modern DevOps*. Packt Publishing. Haettu osoitteesta https://www.safaribooksonline.com

Gregor, S. & Jones D. (2017). The Anatomy of a Design Theory. *Journal of the Association for Information Systems*, 8(5), 312-335.

Griffiths, R., Lawes, A., Brewster, E. & Sandbury, J. (2016). *IT Service Management – Support for your ITSM Foundation exam.* BSC Learning & Development Limited. Haettu osoitteesta https://www.safaribooksonline.com

Gruver, G. & Mouser, T. (2015). *Leading the Transformation. Applying Agile and DevOps Principles at Scale*. IT Revolution. Portland.

Haffke, I. (2017a). *The Implications of Digital Business Transformation for Corporate Leadership, the IT Function, and Business-IT Alignment.* Dissertation. Technische Universität Darmstadt. Haettu osoitteesta https://d-nb.info/1125627395/34

Haffke, I., Kalgovas, B. & Benlian, A. (2017b). Options for Transforming the IT Function Using Bimodal IT. *MIS Quarterly Executive*, *16(2),* 101-120.

Haffke, I., Kalgovas, B. & Benlian, A. (2017c). The Transformative Role of Bimodal IT in an Era of Digital Business. *Proceedings of the 50th Hawaii International Conference on System Sciences, (5460-5469).* Hawaii.

Hevner, A., March, S., Park, J. & Ram, S. (2004). Design science in information systems research. *MIS Quarterly*, *28(1),* 75-105.

Himi, A., Bahsani, S. & Semma, A. (2011). The IT Service Management according to the ITIL framework applied to the enterprise value chain. *International Journal of Computer Science Issues*, 355-363.

Hirsjärvi, S. & Hurme, H. (2000). *Tutkimushaastattelu. Teemahaastattelun teoria ja käytäntö*. Helsinki: Yliopistopaino.

Hirsjärvi, S., Remes. P. & Sajavaara, P. (2004). *Tutki ja kirjoita.* (10. painos). Helsinki: Tammi.

Horlach, B., Drews, P. & Schirmer, I. (2017a). Bimodal IT: Business-IT Alignment in the Age of Digital Transformation, 2016, 1417-1428.

Horlach, B., Drews, P., Schirmer, I. & Böhmann, T. (2017b). Increasing the Agility of IT Delivery: Five Types of Bimodal IT Organization. *Proceedings of the 50th Hawaii International Conference on System Sciences*, 5420-5429. Hawaii.

Humble, J. & Molesky, J. (2011). Why enterprises must adopt devops to enable continuous Delivery. *Cutter IT Journal*, *24(8),* 6-12.

Hüttermann, M. (2012). *DevOps for Developers*. Berkeley, CA: Apress.

Iden, J. & Eikebrokk, T-R. (2013). Implementing IT Service Management: A systematic literature review. *International Journal of Information Management*, *33(3),* 512-523.

Josey, A. (2015). *The IT4IT™ Reference Architecture, Version 2.0 – A Pocket Guide*. The Open Group. Van Haren Publishing. Zaltbommel.

Jöhnk, J., Röglinger, M. & Thimmel, M. (2017). How to implement Agile IT setups: a taxonomy of design options. AIS Electronic Library. *Proceedings of the Twenty-Fifth European Conference on Information Systems (ECIS),* 2017, Portugal.

Karunakaran, S. (2013). *Impact of Cloud Adoption on Agile Software Development*. Software Engineering Frameworks for the Cloud Computing Paradigm. Springer. London.

Kim, G., Behr, K. & Spafford G. (2014). *The Phoenix project. A novel about IT, DevOPS, and helping your business win. The Phoenix project resource guide.* IT Revolution Press, United States of America.

Leffingwell, D. (2018). *SAFe 4.5 Reference Guide: Scaled Agile Framework for Lean Enterprises*. Second edition. Addison-Wesley Professional.

Lwakatare, L. E., Kuvaja P. & Oivo M. (2015). Dimensions of DevOps. *International Conference on Agile Software Development.* 212-217, Springer International Publishing.

Lwakatare, L. E., Kuvaja, P. & Oivo, M. (2016). An Exploratory Study of DevOps Extending the Dimensions of DevOps with Practices. *Proceedings of the Eleventh International Conference on Software Engineering Advances (ICSEA),* Rome, Italy, August 2016.

Macintyre, M., Parry, G. & Angelis, J. (2011). *Service Design and Delivery*. Springer. US.

March, S. T. & Smith, G. F. (1995). Design and natural science research on information technology. *Decision Support Systems, 15(4),* 251–266.

Marrone, M., Gacenga, F., Cater-Steel, A. & Kolbe, L. (2014). IT Service Management: A Cross-national Study of ITIL Adoption. *Communications of the Association for Information Systems, 34(49)*, 865-892.

Metsämuuronen, J. (2005). *Tutkimuksen tekemisen perusteet ihmistieteissä*. Jyväskylä: Gummerus.

Nabiollahi, A. & bin Sahibuddin, S. (2008). Considering Service Strategy in ITIL V3 as a Framework for IT Governance. *Proceedings of Conference: Information Technology, 2008. ITSim 2008.* International Symposium on, vol 1.

Naryan, S. (2015). *Agile IT Organization Design for Digital Transformation and Continuous Delivery.* Addison-Wesley Professional.

Nerur S., Mahapatra, R. & Mangalaraj, G. (2005). Challenges of Migrating to Agile Methods. *Communications of the AMC, 48(5),* 73-78.

Orand, B. (2013). *Foundations of IT Service Management with ITIL 2011. ITIL Foundations Course in a Book*. ITILyaBrady.com.

Ostrowski, L. & Helfert, M. (2012). Reference Model in Design Science Research to Gather and Model Information. *Proceedings of the Eighteenth Americas Conference on Information Systems*, Seattle, Washington, July 29, 2012.

Peffers, K., Tuunanen, T., Rothenberger, M.A. & Chatterjee, S. (2007). A Design Science Researh Methodlogy for Information Systems Research. *Journal of Management Information Systems Research, 24(3),* 45-77.

Puppet. *2017 State of DevOps Report*. (2016). DORA. Haettu 15.4.2018 osoitteesta https://puppet.com/resources/white-paper/2017-state-of-devops-report

Rangama, J., Svendsen, J. G., Scholman, M., Buchanan, S. & Meyler, K. (2017). *Microsoft Hybrid Cloud Unleashed with Azure Stack and Azure*. Sams.

Roche, J. (2013). Adopting DevOps practices in quality assurance. *Communications of the ACM, 56(11),* 38-43.

Scaled Agile Framework (2018). *SAFe 4.5 for Lean Software and Systems Engineering.* Haettu 19.4.2018 osoitteesta http://www.scaledagileframework.com

Suomen Pankki 2016. (2016). Pääjohtaja Erkki Liikanen: Digitalisaatio: Vanhan pelin uudet säännöt vai kokonaan uusi peli? Haettu 30.3.2018 osoitteesta https://www.suomenpankki.fi/fi/media-ja-julkaisut/puheet-ja-haastattelut/2016/paajohtaja-erkki-liikanen-digitalisaatio-vanhan-pelin-uudet-saannot-vai-kokonaan-uusi-peli-maksufoorumi-10.5.2016-helsinki/

Tate, J., Eicher, P., Jagannathan, P. Lad, K. & Burns, C. (2016). *IT Modernization using Catalogic ECX Copy Data Management and IBM Spectrum Storage*. IBM Redbooks. Haettu osoitteesta https://www.safaribooksonline.com

The Open Group. *IT4IT Reference Architecture, Version 2.1 Technical Standard.* (2017). Haettu 2.4.2018 osoitteesta http://pubs.opengroup.org/it4it/refarch21/

Verona, J., Duffy, M. & Swartout, P. (2016). *Learning DevOps: Continuously Deliver Better Software.* Haettu osoitteesta https://www.safaribooksonline.com

Verone J. (2018). *Practical DevOps – Second Edition*. Packt Publishing.

Virmani, M. (2015). Understanding DevOps & bridging the gap from continuous integration to continuous delivery. *Proceedings of the Fifth International Conference on Innovative Computing Technology (INTECH), IEEE Software*, *4(6),* 78-82.

Zhu, L., Bass, L. & Champlin-Scharff, G. (2016). DevOps and Its practices. *IEEE Software, 33(3),* 32-34.

LIITE 1 TEEMAHAASTATTELURUNKO

HAASTATTELUTEEMA 1: DevOps ja IT-palvelutuotanto

1. Miten DevOps vaikuttaa IT-palvelutuotantoon?
2. Miten toimintojen yhdistäminen näkyy toiminnassa?

HAASTATTELUTEEMA 2: DevOps ja roolit, tehtävät sekä vastuut

1. Mitä uusia rooleja tai tehtäviä DevOps tuo IT-palvelutuotantoon?
2. Miten tehtävät yhdistetään palvelutuotannon roolien osalta?

HAASTATTELUTEEMA 3: DevOps ja prosessit sekä käytännöt

1. Miten tuotannon monitorointia tehdään?
2. Miten kevennetään julkaisujen hallintaa (Release Management) ja huomioidaan laatukriteerit (DoR, Definition of Ready)