Tomi Knuuttila

DevOps-käytäntöjen asettamat vaatimukset henkilöstöresursseille

Tekniikan ja innovaatiojohtamisen yksikkö
Pro gradu -tutkielma
Tietojärjestelmätieteen tutkinto-ohjelma

Vaasa 2021

VAASAN YLIOPISTO
Tekniikan ja innovaatiojohtamisen yksikkö
Tekijä: Tomi Knuuttila
Tutkielman nimi: DevOps-käytäntöjen asettamat vaatimukset henkilöstöresursseille
Tutkinto: Kauppatieteiden maisteri
Oppiaine: Tietojärjestelmätiede
Työn ohjaaja: Tero Vartiainen
Valmistumisvuosi: 2021 **Sivumäärä:** 63

TIIVISTELMÄ:
DevOps on kokoelma käytäntöjä, filosofioita ja työkaluja ohjelmistokehityksen organisoimiksesi. Kyseessä on sateenvarjotermi, joka koostuu useista eri osa-alueista. Se ei tarjoa valmiita työkaluja kuten esimerkiksi ohjelmistokehityksen prosessimallia vaan kokoelman käytäntöjä, joiden avulla sopiva prosessimalli voidaan kehittää. Ne eivät kuitenkaan rajoitu vain ohjelmistokehityksen organisoimiseen vaan niiden keskeisenä ideana on organisaation matka kohti oppivaa organisaatiota. Niiden laajamittainen hyödyntäminen vaatii koko organisaation kulttuurinmuutoksen. DevOps-käytäntöjä hyödyntäville organisaatioille tyypillisiä piirteitä ovat korkea automaation aste, matala organisaatio hierarkia, erilaisten mittareiden aktiivinen seuraaminen, pyrkimys nopeaan reagoimiseen, kommunikaation ja yhteistyön korostunut merkitys sekä ketterien ohjelmistokehitysmenetelmien hyödyntäminen.
Pehmeille taidoille ei ole olemassa yhtä yleisesti hyväksyttyä määritelmää, vaan niiden voidaankin katsoa olevan aina sidottuja kontekstiin. Termillä pehmeät taidot tarkoitetaan kuitenkin yleisesti ei-kognitiivisia taitoja kuten sosiaalisia taitoja. Tämän lisäksi termin alle mahtuvat henkilökohtaiset ominaisuudet, luonteen piirteet sekä yksilön elämänkatsomus. Teknologian kehittymisen seurauksena työelämä on murroksessa, jossa toistuvat työtehtävät pyritään automatisoimaan. Näin organisaatioiden henkilöstöstä yhä pienempi osa työskentelee työtehtävissä, joissa vaaditaan vain kovia taitoja. Sen sijaan yhä suurempi osa työtehtävistä nojaa pehmeisiin taitoihin kuten yhteistyökykyyn ja ongelmanratkaisuun. Esimerkkinä tällaisesta muutoksesta on ohjelmistokehitysorganisaatiot, joissa pyritään hyödyntämään DevOps-käytäntöjä. Muutoksen keskeisenä pyrkimyksenä on parantaa yhteistyön ja kommunikaation astetta kokoamalla monialaisia tiimejä, joissa eri toiminnot tekevät keskenään yhteistyötä.
DevOps-käytäntöjä on tutkittu niiden lyhyen historian aikana varsin runsaasti ja laaja-alaisesti sekä akateemisten- että kaupallistentahojen toimesta. Tästä huolimatta olemassa oleva tutkimus ei juurikaan ota suoraan kantaa siihen millaisia pehmeitä taitoja DevOps-käytäntöjen kattava hyödyntäminen vaatii organisaation henkilöstöresursseilta. Tämä siitäkin huolimatta, että DevOps-organisaatioissa korostuvat erityisesti pehmeät taidot kuten sosiaaliset taidot ja ongelmanratkaisukyky. Tämän tutkimuksen tavoitteena oli selvittää henkilöstöresursseilta vaadittuja pehmeitä taitoja organisaatioissa, joissa pyritään työskentelemään DevOps-käytäntöjen mukaisesti.
Tutkimus suoritettiin systemaattisena kirjallisuuskatsauksena olemassa olevasta DevOps-tutkimuksesta. Sen tavoitteena oli määrittää lista sellaisia pehmeitä taitoja, jotka henkilöstöresursseilta vaaditaan DevOps-käytäntöjä hyödyntävässä organisaatiossa. Katsauksessa käytetty tutkimusaineisto rajatiin käsittämään vain akateeminen DevOps-tutkimus, joka oli saatavilla kahdesta eri tietokannasta. Se suoritettiin käyttäen IS tutkimuksen adaptoitua Fink-mallia. Tutkimuksen tuloksena tutkittavista julkaisuista tunnistettiin 23 kappaletta uniikkeja pehmeitä taitoja. Tämän lisäksi työn tuloksena syntyi ajantasainen kirjallisuuskatsaus DevOps-käytännöistä. Tutkimuksen tulokset tarjoavat työkalun esimerkiksi oppilaitosten koulutusten suunnitteluun, organisaatioiden henkilöstöhallinnon rekrytointien suunnitteluun sekä tarjoaa pohjan tulevalle tutkimukselle aiheen ympärille.

AVAINSANAT: DevOps, henkilöstöresurssit, pehmeät taidot

Sisällys

Kuvat

Taulukot

Lyhenteet

AIS – Association for Information Systems

CD – Continuous Delivery

CD – Continuous Deployment

CI – Continuous Integration

IaC – Infrastructure as Code

IS – Information Systems

1 Johdanto

Internet-yhteyksien parantuessa ja teknologian yleistymisen myötä on digitaalisten palveluiden ja sisällön tuottamisen merkitys kasvanut. Digitalisaatio on ajanut myös ne organisaatiot, jotka eivät perinteisesti olleet digitaalisten palvelujen tarjoajia ottamaan valtavaa digiloikkaa niin yksityisellä kuin julkisella sektorilla. Nykyisin valtaosa kuluttajista olettaa palveluiden olevan saatavilla sähköisessä muodossa. Erityisesti nuoremmat sukupolvet ovat tottuneet asioimaan sähköisesti. (Sosiaali- ja terveysministeriö, 2016, ks.4) Ohjelmistokehityksen rooli onkin muuttunut useissa organisaatioissa puhtaasta tukitoiminnosta kiinteäksi osaksi organisaation ydintoimintaa. Organisaatioiden tarjoamat applikaatiot ja rajapinnat ovat usein niiden pääasiallinen kommunikaatioväylä asiakkaalle. Tämän lisäksi ohjelmistoja saatetaan käyttää tukemaan toi optimoimaan organisaation oheistoimintoja. (Amazon Web Services, 2021)

Erityisesti yksityisellä sektorilla digitaalisten palveluiden tarjoaminen on yrityksille tärkeä kilpailukeino niiden kamppaillessa markkinaosuuksista muiden palveluiden tuottajien kanssa. Digitaalisten tuotteiden ja palveluiden heterogeenin luonne lisää kilpailua entisestään kuluttajien pystyessä vaihtamaan palveluiden tarjoajaa pienin tai olemattomin kustannuksin. Kilpailussa pärjäävät parhaiten sellaiset yritykset, jotka kykenevät tuottamaan parhaita palveluita tehokkaimmin ja toimintavarmasti, samalla reagoiden markkinoiden muutoksiin nopeimmin. (Koch, T., Windsperger, J, 2017)

Ohjelmistokehityksen roolin korostuessa on syntynyt tarve pyrkiä tehostamaan organisaatioiden ohjelmistokehitystä ja digitaalisten palveluiden tarjontaa. Viime vuosina suosituimmaksi vaihtoehdoksi on noussut DevOps-viitekehys, joka pyrkii virtaviivaistamaan sekä nopeuttamaan digitaalisten palveluiden kehitystä ja julkaisua. Sen kysyntä kasvoi vuoden 2020 aikana 29,4 % ja sille ennustetaan vuoteen 2027 asti 19,9 % vuotuista kasvua, jolloin sen markkina-arvon ennustetaan olevan 21,4 miljardia. (Global Industry Analysts Inc., 2020)

Sen kasvanutta suosiota selittävät tutkimustulokset, joiden mukaan tehokkaimmin DevOps-käytäntöjä hyödyntävissä organisaatioissa otetaankin uutta koodia käyttöön huomattavasti useammin, koodin läpimenoaika on lyhyempi, virheet harvinaisempia ja virhetilanteista palautuminen nopeampaa. (DevOps Research & Assestment, 2019, s. 21)

Kyseessä on kokoelma käytäntöjä, filosofioita sekä työkaluja, joiden päämääränä on muutos oppivaksiorganisaatioksi iteraation ja jatkuvan oppimisen kautta. Tällainen muutos vaati kuitenkin aina kulttuurisen muutoksen, joka koskee koko organisaatiota. (Forsgren, N., Humble, J. & Kim, G., 2018, s.142–147).

Verrattain uusia käytäntöjä onkin tutkittu varsin kattavasti viime vuosien aikana. DevOps yhteisö itsesään on keskittynyt tutkimaan laaja-alaisesti kuinka eri käytännöt vaikuttavat erilaisiin organisaation ulkoisiin ja sisäisiin mittareihin. Aihetta ovat tutkineet niin Piilaakson suuret teknologia yritykset kuin akateemikot. Laajasta tutkimuksesta huolimatta DevOps-käytäntöjen asettamat vaatimukset henkilöstölle ovat jääneet pienelle huomiolle. Esimerkiksi DevOps-yhteisössä arvostettuja State of DevOps-tutkimuksia julkaiseva DevOps Research & Assestment ei ole ole ottanut aiheeseen ollenkaan kantaa monivuotuisissa tutkimuksissaan. (DevOps Research & Assestment, 2021)

DevOps kulttuuri rakentuu muurien murtamisen, avoimuuden, itseohjautuvuuden, luottamuksen, yhteistyön, kommunikaation, jatkuvan oppimisen ja psykologisen turvallisuuden varaan. Perinteisesti kuitenkin ohjelmistotuotannon eri osa-alueet on vahvasti eroteltu toisistaan. Tällainen asetelma synnyttää helposti kitkaa tuotantoketjun eri osien välille. Ongelman purkamiseksi ei kuitenkaan riitä, että eri toimet liitetään toisiinsa organisaatiokaaviossa. Sen sijaan tarvitaan kulttuurinmuutosta. Tällainen muutosprosessi on kuitenkin hidas sekä altis vaikutteille ja vaatii aina koko organisaation sitoutumisen. Erityisen tärkeää on organisaation johdon sitoutuminen muutokseen. (Ravichandran A., Taylor K., Waterhouse P., 2016, s. 26–33)

On kuitenkin selvää, että kommunikaatiota, avoimuutta ja yhteistyötä korostavaa kulttuuria ei ole mahdollista rakentaa ilman pehmeitä taitoja. Niiden merkitys korostuu kulttuurin lisäksi myös eri toimintojen välisessä yhteistyössä.

Pehmeiden taitojen roolia ei kuitenkaan ole perinteisesti korostetut teknisessä koulutuksessa, vaan niissä on sen sijaan keskitytty kovien taitojen kehittämiseen. Niiden merkitys nousee kuitenkin esille erityisesti, kun organisaatioissa pyritään lisäämään yhteistyötä eri toimintojen välillä tai työskennellessä organisaation sidosryhmien kanssa. (Bancino & Zevalkink 2007)

Tässä tutkimuksessa keskitytään siihen, millaisia pehmeitä taitoja DevOps-viitekehys vaatii organisaation henkilöstöltä. Näin tutkimuksen tutkimuskysymykseksi muodostuu: *Millaisia pehmeitä taitoja DevOps-organisaation henkilöstöltä vaaditaan*

DevOps-viitekehyksen yleistyessä ja sen merkityksen kasvaessa tämän tiedon tärkeys korostuu entisestään. Sitä voidaan hyödyntää oppilaitosten koulutusohjelmia suunniteltaessa, organisaatioiden henkilöstöhallinnossa koulutuksia suunniteltaessa sekä rekrytoinnissa, konsultaatiota tarjoavissa yrityksissä sekä tulevassa tutkimuksessa. Näiden taitojen ymmärtämisen merkitys korostuu erityisesti organisaatioissa, joissa kulttuurinmuutosprosessi on vasta alkamassa. Erityisesti pienemissä organisaatioissa yksittäinenkin virheellinen rekrytointi saattaa vaarantaa koko kulttuurinmuutoksen tai ainakin hidastaa prosessia merkittävästi samalla vaikuttaen organisaatioiden ydintoimintojen tehokkuuteen. Näiden riskien välttämiseksi on ensiarvoisen tärkeää ymmärtää millaisia pehmeitä taitoja DevOps vaatii organisaatioiden henkilöstöltä.

2 DevOps

DevOps on kokoelma ohjelmistokehityskäytäntöjä, filosofioita ja työkaluja ohjelmistokehityksen ongelmakohtien ratkaisemiseksi. Se ei tarjoa valmista prosessimallia tai ratkaisuja siihen, kuinka organisaation toiminta tulisi organisoida. Se voidaankin mieltää työkalupakiksi, josta löytyvien työkalujen avulla organisaatiot voivat suunnitella ja rakentaa juuri heidän tarpeisiinsa sopivan ratkaisun. DevOps-viitekehyksen sisältämät käytännöt, filosofiat ja työkalut eivät ole uusia. Sen sijaan ne ovat hyväksi koettuja, testattuja, tutkittuja ja toimiviksi todettuja ja yhteensopivia. Termin DevOps-alta löytyy useita erilaisia tapoja tehdä asioita, ja viitekehys itsessään ei ota kantaa siihen millaiset ratkaisut sopivat mihinkin organisaatioon tai tilanteeseen.

2.1 Määritelmä

Termille DevOps ei ole olemassa yksiselitteistä yleisesti hyväksyttyä määritelmää. Sille on sen lyhyen historian aikana yritetty tarjota useita erilaisia määritelmiä. Esimerkiksi Jabbari, Ali, Petersen, Tanveer (2016, s.8) ehdottavat määritelmää:

> *DevOps is a development methodology aimed at bridging the gap between Development and Operations, emphasizing communication and collaboration, continuous integration, quality assurance and delivery with automated deployment utilizing a set of development practices.*

Toisaalta Dyck, Penners, Lichter (2015) määrittelevät sen organitorisena ilmiönä:

> *DevOps is an organizational approach that stresses empathy and cross-functional collaboration within and between teams – especially development and IT operations – in soft are development organizations, in order to operate resilient systems and accelerate delivery of changes.*

Kun taas Davis ja Daniels (2016) määrittelevät sen kulttuurillisesta näkökulmasta:

> *Devops is a cultural movement that changes how individuals think about their work, values the diversity of work done, supports intentional processes that accelerate the rate by which businesses realize value, and measures the effect of social and technical change. It is a way of thinking and a way of working that enables individuals and organizations to develop and maintain sustainable work practices. It is a*

> *cultural framework for sharing stories and developing empathy, enabling people and teams to practice their crafts in effective and lasting ways.*

Yhteistä näille määritelmille on kuitenkin se tavoitetila, johon DevOps-käytäntöjen soveltamisella pyritään. Lopullisena tavoitteena on aina tehostaa organisaation ohjelmistotuotannon laatua, vakautta ja reaktiokykyä. Muita yhtäläisyyksiä ovat korkean automaation asteen korostaminen, jatkuva integraatio, jatkuva julkaisu, yhteistyö ja kommunikaatio. DevOps-käytäntöjen soveltaminen on aina askel kohti oppivaa organisaatiota loputtoman iteroinnin ja jatkuvan oppimisen kautta. Ei ole olemassa yhtä yksiselitteistä ratkaisua, joka sopisi jokaiseen organisaatioon.
Tässä tutkimuksessa käytetään löyhää määritelmää termille DevOps: kyseessä on kokoelma ohjelmistokehityskäytäntöjä, filosofioita ja työkaluja, joita hyödyntämällä organisaatiot pyrkivät tehostamaan ohjelmistokehitystä.

2.2 Historia

Termin DevOps historia alkaa vuodesta 2009 kun Andrew Clay Shafer tiivisti otsikon ”10+ Deploys per Day: Dev & Ops cooperation” hashtagiin #devops Twitterissä, ilmaistakseen seuraavansa vuoden 2009 O’Reily – Velocity konferenssia, vaikka ei itse fyysisesti ollutkaan paikalla. Vastauksena tähän viestiin Pris Nasrat ehdotti, että Shafer voisi järjestää oman konferenssinsa Belgiassa. Tästä seuranneen keskustelun seurauksena päädyttiin järjestelmään ensimmäinen DevOpsDays-konferenssi, joka keskittyi erityisesti ohjelmistokehityksen ja operatiivisen-IT:n yhteistyölle. Tämän tapahtumasarjan seurauksena järjestettiin ympäri maailmaa lukuisia DevOpsDays-tapahtumia ja konsepti alkoi elää omaa elämäänsä. (Davis, Daniels, 2016, s.28 – 29)

2.3 Osa-alueet

2.3.1 Kulttuuri

Kulttuurin arvo nostetaan DevOps-viitekehyksessä korkealle. Jokaisen organisaation kulttuuri on aina yksilöllinen. Näin ollen myös DevOps-organisaatioiden kulttuurit ovat

yksilöllisiä ja sidottuja tarkasteltavaan organisaatioon. On kuitenkin olemassa sellaisia kulttuurin piirteitä, joiden voidaan katsoa olevan osa DevOps-organisaatiokulttuuria. Organisaatiot, jotka pyrkivät toimimaan viitekehyksen mukaisesti tavoittelevat näitä tekijöitä tai saavuttavat ne riippuen siitä missä vaiheessa muutosprosessia kohti oppivaa organisaatiota ne ovat.

DevOps-organisaatiokulttuurissa pyritään suosimaan yhteistyötä yksilöiden jalustalle nostamisen sijaan. Yhteisiin tavoitteisiin pyritään tekemällä yhteistyötä jatkuvien iteraatioiden kautta. Tavoitteita mitataan aktiivisesti ja niiden avulla pyritään parantamaan eri osa-alueita aina asiakastyytyväisyydestä uusien ohjelmistoversioiden läpimenoaikaan. Jatkuvien iteraatioiden aikana pyritään oppimaan matkalla, kuinka organisaation toimintaa voitaisiin parantaa. Haitalliseksi katsottua toimintaa kuten vastuun välttelyä, virheistä rankaisemistä, syyllisten etsimistä ja niiden rankaisemista pyritään välttämään. Näitä asioita pyritään välttämään useammastakin eri syystä. Koska kyseessä organisaation yhteinen pyrkimys kohti oppivaa organisaatiota tulee virheistä pyrkiä oppimaan sen sijaan että niistä rangaistaisiin. Tällä tavalla muodostuu tilanne, jossa organisaation jäsenet eivät uskalla toimia koska pelkäävät aina rangaistuksia ja tämä syö resursseja, henkilöstön moraalia ja poistaa mahdollisuuden oppimiseen. (Ravichandran, Taylor, Waterhouse, 2016)

Psykologinen turvallisuus onkin erityisen tärkeää DevOps-kulttuurissa, sillä se mahdollistaa tehokkaan ryhmätyöskentelyn, innovaation ja avoimen tiedon kulun. Psykologisella turvallisuudella tarkoitetaan sitä, ettei ketään rangaista sen perusteella, että hän jakaa ideoita, kysyy jotain tai nostaa esiin ongelmia ja huolenaiheita. Tällaisessa ympäristössä henkilöstö uskaltaa ottaa riskejä ja asettaa itsensä haavoittuvaan tilaan, luottaen siihen, etteivät muut hyväksikäytä hänen potentiaalisesti haavoittuvaa asemaansa. Tämä edesauttaa oppimista sekä yksilön että organisaation tasolla, tehostaa organisaation tiedonkulkua ja sitä kautta toimintaa. Organisaatiossa, jossa vallitseva tila ei ole psykologisesti turvallinen henkilöstö ei uskalla tuoda esiin mielipiteitään, ottaa riskejä sekä epäröi heille epäsuotuisan tiedon jakamisen suhteen. Tämä johtaa kommunikaatiokatkoksiin ja siitä aiheutuviin ongelmiin, innovaation vähentymiseen sekä heikentää yhteenkuuluvuuden

tunnetta. Psykologisen turvallisuuden kulttuuri vaikuttaakin suoraan organisaation tehokkuuteen. (Davis & Daniels, 2019, s.4; DevOps Research & Assessment, 2019, s.54)

2.3.2 Kommunikaatio

Yhteistyön edellytyksenä on tehokas kommunikaatio. Kommunikaatio auttaa ihmisiä ymmärtämään toisiaan paremmin ja mahdollistaa tiedon kulkemisen organisaatiossa. Tämän lisäksi se auttaa ihmisiä ymmärtämään toisiaan ja auttaa löytämään yhteisiä näkökantoja sekä edesauttaa yhteistyön kulttuurin ja yhteisön muodostumisessa. (Davis & Daniels, 2019, s.5)

Samalla se lisää yksilöiden ja yhteisön yhteenkuuluvuuden tunnetta. Tämä yhteenkuuluvuuden tunne vaikuttaa olennaisesti siihen mitä yksilöt ja yhteisö kestävät vastoinkäymisiä. Samalla se edesauttaa tiedon kulkemista organisaatiossa. Kommunikaatiolla onkin suuri merkitys ymmärryksen lisäämisessä sekä yksilö että organisaatiotasolla. Sen avulla organisaatiossa voidaan jakaa tietoa ja oppia muilta. Tämä on erityisen tärkeää sellaisen tiedon kannalta, joka elää vain organisaation sisällä dokumentaation ulkopuolella eli niin sanotun hiljaisen tiedon kannalta. Tällainen tieto on kuitenkin usein todella merkittävässä roolissa, sen avulla organisaatiossa voidaan oppia historiasta ja välttää toistamasta virheitä, jotka on jo kertaalleen tehty. Kommunikaatio on myös merkittävässä roolissa organisaation henkilöstön hyvinvoinnin ja motivaation kannalta. Jotta yksittäinen organisaation jäsen tuntisi olevansa arvostettu yhteisön jäsen tulee hänen saada positiivista palautetta työstään. Erityisen tärkeää positiivinen palaute on sellaisina ajankohtina, kun yhteisön jäsenen työpanos on ollut suuri, tämä on tehnyt jotain tavallisesta poikkeavaa, oppinut jotain tai muutoin ylittänyt itsensä. Tällaiset hetket ovat merkittäviä yhteisön jäsennelle itselleen ja samalla tarjoavat mahdollisuuden palkita työpanoksesta lisäten näin todennäköisyyttä, että vastaavaa tapahtuu myös tulevaisuudessa. (Davis & Daniels, 2016, s. 90)

2.3.3 Yhteistyö

Yhteistyö määritellään prosessiksi, jonka aikana ryhmä ihmisiä työskentelee yhdessä saavuttaakseen ennalta määrätyn tavoitteen tai määränpään. Termi DevOps syntyi vuonna 2009 kuvaamaan ohjelmistokehityksen (development) ja ylläpidon (operations) yhteistyötä. Edelleen eri toimintojen välinen yhteistyö on olennainen osa DevOps-käytäntöjä. On kuitenkin painotettava, että yhteistyö ei rajoitu vain ohjelmistotuotannon ja ylläpidon yhteistyöhön vaan se voi koskea mitä organisaation toimintoa logistiikasta markkinointiin. Yhteistyö rakentuu päivittäisissä kohtaamisissa, yhteisissä tavoitteissa, osallistumisessa, sekä organisaation tavoissa. Erilaisia yhteistyön ilmentymiä ovat esimerkiksi koodin katselmointi, tilanne päivitykset tai viikoittaiset demot, jossa esitellään omaa työtä muille. (Davis & Daniels, 2016, s.63 – 71)

Yhteistyö DevOps-tiimissä voidaan jakaa kolmeen eri vaiheeseen: tutkimus, implementointi ja julkaisu. Tutkimus vaiheen aikana kehitystiimi keskittyy suunnittelemaan mitä seuraavan ohjelmistokehitysiteraation aikana tehdään. Tämän vaiheen aikana keskitytään suunnittelemaan, keräämään vaatimusmäärittelyjä, käydään läpi arkkitehtuurillisia ratkaisuja ja suunnitellaan tulevaa ohjelmistokehitystä. Tämän vaiheen aikana tiimi saattaa tehdä kattavasti yhteistyötä eri toimintojen kanssa varmistaakseen, että muutos on linjassa muun organisaation toiminnan kanssa. Jos kehitettävällä ominaisuudella on riippuvuuksia organisaation muihin toimintoihin, on tärkeää, että näiden tarve selvitetään tässä vaiheessa ja resurssien jakamisestä sovitaan etukäteen. Onkin olennaista, että kommunikaatio toimii myös organisaation muiden toimintojen välillä.

Tutkimusvaihetta seuraa implementaatiovaihe, jonka aikana edellisessä vaiheessa tehty suunnitelma toteutetaan. Ohjelmistokehityksen aikana voi esiintyä useita erilaisia yhteistyön muotoja. Kirjoitettava koodi tulee liittää samaan lähteeseen versionhallinnassa. DevOps-kontekstissa tämä tarkoittaa jatkuvaa integraatiota. Siinä kehittäjät luovat omia kehityshaarojaan kehitettävän ominaisuuden luomiseksi. Kun työ on valmis, kehitetty koodi testataan yhdessä sovittujen käytäntöjen mukaisesti. Sen jälkeen se käy läpi koodinkatselmointi vaiheen, jossa ennalta sovittu määrä muita tiimin kehittäjiä lukee koodin läpi antaen palautetta ja korjausehdotuksia. Kun ehdotetut korjaukset on suoritettu, se hyväksytään yhteispäätöksellä ja liitetään versionhallinnan päähaaraan, josta se on

muiden kehittäjien saatavilla. Jos ohjelmistokehityksen aikana ilmenee ongelmia niistä, keskustellaan ja ongelmakohtia pyritään ratkomaan yhdessä. Ohjelmistokehityssyklin jälkeen tiimi läpikäy edellisen syklin retrospektiivipalaverissa, jossa pohditaan, mikä meni huonosti ja mikä hyvin. Näiden pohdintojen ja löydösten perusteella pyritään oppimaan ja parantamaan prosessia ennen seuraavaa ohjelmistokehityssykliä.
Implementaation jälkeen siirrytään julkaisuun. Tämä vaihe ei välttämättä tapahdu ajallisesti ohjelmistokehityssyklin jälkeen vaan se voi tapahtua sen aikana jatkuvan julkaisun periaatteen mukaisesti. Jatkuvan kommunikaation periaatteen mukaisesti jokaisen tiimin jäsenen tulee olla tietoinen tapahtuvista julkaisuista. DevOps-viitekehyksen mukaisesti vastuu kehitettävästä ohjelmistosta ja sen ylläpidosta on kuitenkin aina tiimin harteilla, joten julkaisun jälkeen tiimi on vastuussa sen ylläpidosta. Ylläpitoa varten tiimin tulee määritellä yhteiset säännöt, miten sen suhteen toimitaan. Jos tiimin jäseniä on tarpeeksi, voidaan taata, että joku on aina tavoitettavissa. Tätä varten tuleekin sopia vuorot, joiden aikana sovitut tiimin jäsenet ovat tavoitettavissa. Näin voidaan samalla mentoroida tuoreita tiimin jäseniä ja lievitetään vaativan tilanteen luomaa stressiä. Vikatilanteet tulee myöhemmin yhdessä läpikäydä *post mortem*-käytännön mukaisesti. Siinä vikatilanne läpikäydään huolellisesti juurisyiden selvittämiseksi ja dokumentoidaan. Kun syy on selvinnyt, suunnitellaan ratkaisu, jotta tällainen vika voidaan välttää tulevaisuudessa. Olennainen osa tätä käytäntöä on opitun tiedon jakaminen kaikille, jotta siitä opitaan kollektiivisesti. (Davis, Daniels, 2019, s.19 – 31)

2.3.4 Palautesilmukka

Nopea adaptoituminen muuttuviin vaatimusmäärittelyihin ja loppukäyttäjien palautteeseen samanaikaisesti pysyen laadukkaana ja luotettavana on oleellinen tapa ohjelmistoja ja IT-palveluja tuottaville yrityksille luoda kilpailuetua. DevOps-organisaatioissa pyritään saavuttamaan kaikkia näitä osa-alueita järjestelemällä prosesseja uudelleen sekä hyödyntämällä kattavasti ohjelmistoautomaatiota. Tähän pyritään, jotta loppukäyttäjältä kehittäjille tulevaan palautteeseen pystyttäisin reagoimaan mahdollisimman nopeasti. Palautteeseen reagoimiseen vaikuttaa kaksi asiaa: kuinka nopeasti palaute saavuttaa kehittäjän ja kuinka suuri julkaistu muutos oli. Mitä nopeammin palaute tavoittaa kehittäjän,

sitä nopeammin kehittäjän on mahdollista reagoida siihen. Palautteeseen reagoiminen on aina sitä helpompaa mitä nopeammin se saavuttaa kehittäjän. Jos muutoksesta on kulunut jo aikaa, joutuu kehittäjä palaamaan siihen mitä on tehty aiemmin ja pohtimaan ovatko sen jälkeen tehdyt muutokset saattaneet vaikuttaa asiaan. Mitä pienempi edellinen muutos oli sitä tarkemmin kehittäjät voivat arvioida mitä saatu palaute tarkoittaa. Nopea ja pienet muutokset mahdollistavatkin esimerkiksi A/B-testauksen. (Forsgren, Humble. & Kim, 2018, s.48)

Jokainen muutos vakaaseen järjestelmään on kuitenkin aina riski. Vaikka DevOps-käytäntöjen mukaisesti kehittävän ohjelmisto testataan automatisoidusti ei voida taata, ettei muutos aiheuttaisi ongelmia tuotannossa. Testikattavuus ei välttämättä riitä kattamaan jokaista järjestelmän osaa tai testeissä itsessään on virheitä. Osa ongelmista saattaa esiintyä viiveellä tai niiden havaitseminen vie aikaa. Järjestelmän systemaattinen tarkkailu, analysointi ja läpikäyminen on kuitenkin aikaa vievää sekä altista inhimillisille virheille. Tämän takia automaation hyödyntäminen järjestelmän tilaa tulee tarkkailla jatkuvasti hyväksikäyttäen ohjelmistoautomaatiota. Jos järjestelmässä esiintyy poikkeavuuksia tai vikatiloja, voidaan ohjelmistoautomaation keinoin vaihtaa jaettava ohjelmistoversio vanhempaan tai kerätään dataa, jota voidaan hyödyntää järjestelmän korjaamisessa. Jos ongelma esiintyy vain tietyissä olosuhteissa, on sen paikallistaminen hidasta ja vaikeaa ilman olemassa olevaa tietoa siitä millaisissa olosuhteissa ongelma esiintyy. Tämän järjestelmän tavoitteena on tarjota palautesilmukka DevOps-tiimille, joka mahdollistaa reagoimisen järjestelmässä esiintyviin virheisiin mahdollisimman nopeasti, vaikka niitä ei havaittaisi testauksen aikana. Virheiden etsiminen ja korjaus on kallista ja aikavievää työtä, joka vaikeutuu mitä pidempi aika virheen syntymisestä on. Jos virhettä ei huomata ajoissa voi sen jälkeen tapahtua useita julkaisuja, joka vaikeuttaa oleellisesti virheen etsimistä. (Figalist, Biesdorf, Brand, Feld ja Kiermeier, 2019, s. 1)

2.3.5 Mittarit

Jotta asioita voidaan kehittää, tulee niitä pystyä mittaamaan luotettavasti. Tämän lisäksi on olennaista ymmärtää mitä asioita mitataan sekä syy-seuraussuhteet, jotka vaikuttavat näihin mittareihin. Koska jokainen DevOps-organisaatio eroaa toisistaan vaihtelevat

relevantit mittarit organisaation mukaan perustuen organisaation toiminnan ajureihin. Mittarit saattavat vaihdella myös organisaation sisällä eri toimintojen välillä, jos toimintojen tavoitteet eivät ole samat. Mittaaminen ei rajoitu vain tekniikkaan ja prosesseihin vaan myös organisaatiota ja sen kulttuuria tulee mitata aktiivisesti. (Sharma, 2017, s. 33 - 35)

Organisaation ohjelmistojen toimituksen suorituskyky korreloi suoraan sekä organisaation kaupallisen tehokkuuden että myös ei kaupallisten tavoitteiden saavuttamisen kanssa. Se muodostuu kahdesta eri komponentista, ohjelmistokehityksestä sekä julkaistavien ohjelmistojen ylläpidosta. Niiden suorituskyvyn arvioimiseksi tulee organisaation kyetä mittaamaan niiden tehokuutta erikseen. Suorituskyvyn mittarit ovat kuitenkin aina riippuvaisia organisaation toiminnan ajureista ja näin ovat aina spesifejä jokaiselle organisaatiolle. Mittareiden tulisi kuitenkin täyttää kaksi ehtoa: niiden tulee mitata tulosta eikä tuotosta sekä mitata tulosta organisaation näkökulmasta. Ensimmäinen ehto varmistaa, ettei eri toimintojen välille synny ristiriitaa ja kaikki hyötyvät hyvästä tuloksesta. Toinen ehto on, että mittari mittaa tulosta eikä tuotetta. Näin voidaan varmistaa, ettei mittari mittaa työtä, joka ei suoraan heijastele koko organisaation kannalta positiiviseen tulokseen. Mittaamalla esimerkiksi ohjelmistokehityksen suorittamien tehtävien määrää saattaa antaa väärän kuvan ohjelmistokehityksen tilasta sillä se ei ota kantaa siihen millaisia tehtäviä ohjelmistokehityksessä tehdään. Kun mittarit ovat paikallaan voidaan niistä saatua dataa alkaa käyttämään aktiivisesti organisaation toiminnan kehittämiseen. Luotettavan datan avulla organisaatiossa voidaan myös alkaa testaamaan erilaisia hypoteeseja siitä mitkä asiat vaikuttavat organisaation tehokkuuteen. Vaikka mittarit ovatkin organisaatiospesifejä voidaan avain mittareita vertailla samalla toimialalla toimivien organisaatioiden kanssa, joka auttaa hahmottamaan organisaation suorituskykyä ja markkina-asemaa. (Forsgren, Humble & Kim, 2018, s. 44 - 61)

Myös organisaatiokulttuurin terveys korreloi suoraan organisaation tehokkuuden ja eikaupallisten tavoitteiden saavuttamisen kanssa. Kulttuurin mittaamiseksi tulee kuitenkin olla ensin jokin malli, jota voidaan käyttää mittaamisen lähtökohtana. Kulttuurien mallintamiseen on olemassa useita erilaisia malleja, joita on mahdollista käyttää kulttuurien mittaamiseen. Forsgren, Humble & Kim (2018, s.63) suosittelevat DevOps-

organisaatioiden mallintamiseen sosiologi Ron Westrumin kehittämää mallia. Sen mukaan organisaation kulttuurin perusteella voidaan ennustaa, kuinka informaatio kulkee organisaatiossa ja kuinka tehokas se organisaatio on. Kulttuuria voidaan mitata kyselytutkimuksilla. On kuitenkin olennaista varmistaa, että jokainen kyselyyn vastannut on ymmärtänyt kysymykset samalla tavalla ja että kysely mittaa oikeita asioita, jotta kyselyn tuloksia voidaan pitää tilastollisesti uskottavina. Westrumin teorian mukaan mitä paremmin informaatio kulkee organisaatiota pitkin, sitä tehokkaampi organisaatio on. Hyvän organisaatiokulttuurin ennakkoedellytyksenä on, että koko organisaatiossa tehdään yhteistyötä ja ihmiset luottavat toisiinsa. Hyvässä organisaatiokulttuurissa tehdään parempia päätöksiä, sillä informaatiota on paremmin saatavilla ja virheelliset päätökset voidaan nopeammin korjata psykologisesti turvallisessa ympäristössä. Tämä johtaa siihen, että organisaatiossa tehdään kokonaisvaltaisesti laadukkaampaa työtä, kun virheitä löydetään nopeammin ja niistä uskalletaan puhua avoimesti. (Forsgren, Humble & Kim, 2018, s. 63 - 71)

2.3.6 Automaatio

Prosessien automatisointi vähentää manuaalisen työn, energian ja resurssien tarvetta. Samalla se parantaa prosessien laatua, luotettavuutta, täsmällisyyttä ja sen tuloksena lopputulos on aina toistettavissa. Potentiaalisia sovelluskohteita ovat järjestelmän infrastruktuuri, järjestelmä resurssien allokointi, ohjelmiston testaus, ohjelmiston kääntäminen sekä sen tarvitsemien artefaktien generointi, mittarit, monitorointi ja palautejärjestelmät. (Davis & Daniels, 2016, s. 192)

Järjestelmän tarvitseman infrastruktuurin ja sen resurssien allokoinnin automatisointi tapahtuu hyödyntämällä sen kuvaukseen suunniteltuja kuvauskieliä (infrastructure as code, IaC). Niiden avulla kuvataan, millaisia resursseja kehitettävä ohjelmisto tarvitsee ja miten näitä resursseja allokoidaan eri tilanteissa. Käytettävät työkalut ovat kuitenkin aina riippuvaisia siitä mikä on haluttu lopputulos ja näin myös riippuvaisia organisaatiosta, joka haluaa automatisoida infrastruktuuriaan. Niitä hyödyntämällä resurssien määrittelyyn voidaan käyttää samoja prosesseja kuin muuhunkin kehitettävään ohjelmistoon. Sitä voidaan kehittää lokaalissa ympäristössä, se voidaan laittaa versionhallintaan ja sitä

voidaan testata ennen julkaisua ja olemassa olevasta infrastruktuurista on olemassa aina ajantasainen dokumentaatio. Kun nämä kuvaukset on kirjoitettu kerran, voidaan niitä hyödyntää niin kauan, kun niiden katsotaan olevan riittäviä. Kun niitä halutaan päivittää, on mahdollista vain päivittää olemassa olevat kuvaukset siten että ne sopivat uuteen tilanteeseen. Jos kehitettävästä järjestelmästä halutaan luoda useita eri versioita, voidaan samoja kuvauksia uudelleen käyttää. Tällainen käytäntö vähentääkin inhimillisten riskien mahdollisuutta sekä vapauttaa resursseja muihin tehtäviin, parantaa ohjelmistokehitystiimin tehokkuutta sekä lisää ketteryyttä ja helpottaa riskien hallintaa. (Davis & Daniels, 2016, s. 181 - 185)

Nykyaikaiset ohjelmistot koostuvat useista osista, jotka tulee kääntää oikeassa järjestyksessä. Tämän lisäksi niillä on usein riippuvuuksia eri kirjastoihin, joka lisää vielä kompleksisuutta. Ohjelmistojen monimutkaistuessa on myös niiden käännösprosessista tullut monimutkaisempi, joka on synnyttänyt tarpeen tämän prosessin automatisoinnille. Sitä varten on kehitetty käännösautomaatiotyökaluja, jotka huolehtivat siitä, että ohjelmiston eri osat käännetään oikeassa järjestyksessä ja että olemassa olevat ulkoiset riippuvuudet otetaan käännökseen mukaan ja tarvittavat artefaktit generoidaan. (Davis ja Daniels, 2016, s. 185 - 187)

Käännettävä ohjelmisto on kuitenkin syytä myös testata ennen sen julkaisua. Tällaiseen tilanteeseen voidaankin hyödyntää testiautomaatiota. Testiautomaation avulla pyritään varmistamaan, etteivät uudet muutokset ole rikkoneet olemassa olevaa ohjelmistoa. On kuitenkin huomioitava, että ohjelmistojen kääntäminen saattaa usein viedä huomattavasti aikaa. Tämän vuoksi kehitettävässä ohjelmistossa esiintyvät virheet on parempi huomata ennen käännösprosessia. Tähän voidaan myös käyttää testiautomaatiota suorittamalla yksikkötestit uudelle koodille. Näin virheet saadaan nopeammin kiinni, joka helpottaa niiden korjaamista ja säästää kehittäjien aikaa. (Sharma, 2017, s. 143 - 145)

Monitoroinnilla tarkoitetaan ylläpidettävän järjestelmän ja ohjelmiston tilan seurantaa sekä jatkuvaa tietojen keräämistä. Tietojen keräämiselä tarkoitetaan järjestelmän tilasta kertovien tietojen generointia, organisointia sekä tallentamista. Järjestelmät koostuvat usein useista erilaisista palveluista, joista jokainen saattaa tuottaa huomattavan määrän tietoa omasta tilastaan. Kaiken saatavilla olevan tiedon tallentaminen ei ole välttämättä

mielekästä vaan näiden tietojen tulee olla hyvin kohdennettuja ja organisoituja jotta niitä voidaan tehokkaasti hyödyntää tarvittaessa. Monitoroinnin automatisoinnilla pyritään siihen, ettei järjestelmän tilaa tarvitse aktiivisesti tarkkailla vaan voidaan luottaa siihen, että järjestelmän tilan poiketessa halutusta järjestelmä hälyttää sitä ylläpitävälle taholle automaattisesti ennalta määriteltyä sähköistä viestivälinettä käyttäen. Hälytyksien automatisoinnin yhteydessä on kuitenkin oleellista suunnitella niiden käyttö etukäteen. Suunnitellessa tulee huomioida hälytyksen syyn vaikutukset, kiireellisyys, ketä asia koskee ja käytettävissä olevat resurssit. Jos asian vaikutukset eivät ole järjestelmän toimivuuden kannalta välitöntä huomiota tarvitsevia niin käytettävä hälytysmuoto voi olla hienovaraisempi. Jos asia vaatii taas välitöntä huomiota niin hälytyksen tulee tavoittaa ylläpitävät tahot mahdollisimman tehokkaasti ja nopeasti. On myös mahdollista, että hälytys syntyy esimerkiksi palvelusta, jota ylläpitää organisaatiossa toinen toiminto. Tällaisessa tapauksessa hälytyksen tulisi informoida kyseistä palvelua ylläpitävää osapuolta. On myös syytä ennalta läpikäydä mihin tilanteisiin organisaatiolla resursseilla kyetään reagoimaan. (Davis ja Daniels, 2016, s. 188 - 191)

2.3.7 Jatkuva integraatio

Ohjelmistokehitys tehdään lähes poikkeuksetta ryhmätyönä, jossa samanaikaisesti useat eri kehittäjät tekevät muutoksia koodiin. Versionhallinta järjestelmät mahdollistavat useiden kehittäjien osallistumisen kehitykseen itsenäisesti. Kehittäjät luovat tehtäväänsä varten oman version ohjelmistosta eriyttämällä versionhallinnan päähaarasta tehtävää varten oman kehityshaaran. Modernissa ohjelmistokehityksessä pyritään myös aktiivisesti hyödyntämään jo olemassa olevaa koodia kuten avoimen-lähdekoodin projekteja ja erilaisia ohjelmointirajapintoja. On myös mahdollista, että organisaation muut toiminnot saattavat tehdä jotain osaa kehitettävästä ohjelmistosta tai että siinä hyödynnetään organisaation ulkopuolisia resursseja. Tämä tarkoittaa sitä, että olemassa olevaa ohjelmistoa muuttaa samanaikaisesti useita ihmisiä, joiden kaikkien työ tulee liittää osaksi kehittävää ohjelmistoa ennen kuin se on valmis. Uuden koodin integrointi olemassa olevaan ohjelmistoon on kuitenkin monimutkainen ja aikaa vievä prosessi, jonka vaativuus on suoraan verrannollinen integroitavan muutoksen kokoon. Koodin integroiminen

osaksi olemassa olevan ohjelmiston lähdekoodia onkin yksi ohjelmistokehityksen keskeisistä tehtävistä. Tämän prosessin suorittamista jatkuvasti kutsutaan jatkuvaksi integraatioksi (continuous integration, CI). (Sharma, 2017, s. 38)

Perinteisesti uusia ominaisuuksia olemassa olevaan ohjelmistoon on kehitetty niiden omissa kehityshaaroissaan, jotka saattavat elää päiviä tai jopa viikkoja ennen kuin ne integroidaan yhteiseen koodikantaan. Pitkällä aikavälillä uutta koodia on kuitenkin kirjoitettu todennäköisesti paljon sekä uuden ominaisuuden kehityshaaraan että versionhallinnan päähaaraan. Tämä kasvattaa riskiä siihen, että integroitavat muutokset eivät ole enää yhteensopivia versionhallinnan päähaaran kanssa. Mitä laajempia muutokset ovat kummassakin haarassa sitä vaikeampi niitä on korjata, joka synnyttää tarpeen uudelleen kirjoittaa osia integroitavasta koodista. Vaihtoehtoisesti jatkuvan integroinnin periaatteen mukaisesti integroimalla jatkuvasti pieniä muutoksia ovat sekä rikkoutumisen todennäköisyys että mahdollisten korjausten pinta-ala pienempiä. Integraatioiden onnistumisen testaamiseksi voidaan hyödyntää testiautomaatiota. Ennen koodin lopullista liittämistä päähaaraan voidaan vielä varmistaa, ettei integroitava koodi riko mitään. (Davis ja Daniels 2016, s. 38)

2.3.8 Jatkuva toimitus

Jatkuvalla toimituksella (continuous delivery, CD) tarkoitetaan kokoelmaa työkaluja, jotka mahdollistavat muutosten viemisen tuotantoon nopeasti ja turvallisesti. Nämä työkalut ovat jatkuva integraatio, jatkuva testaus ja kattava kokoonpanon hallinta. Jälkimmäisellä tarkoitetaan kykyä pystyttää ympäristöjä, kääntää ja testata koodia automatisoidusti ilman tarvetta manuaaliselle konfiguroinnille. Jatkuvan toimituksen implementointi vaatii useiden eri toimintojen yhteistyötä sekä palautesilmukoiden luomista julkaistavan ohjelmiston laadun takaamiseksi. Oikein implementoituna jatkuva toimitus kuitenkin mahdollistaa uusien julkaisujen luomisen tarpeen mukaan. Se tarjoaa kehittäjille työkalut, jotka mahdollistavat ongelmien havaitsemiseen mahdollisimman nopeasti sekä toistuvien tehtävien kuten kääntämisen ja testaamisen automatisointiin. Kun ongelmat havaitaan heti niiden syntyessä, on niiden korjaaminen helpompaa ja vaikutukset jäävät pienemmiksi. Kun toistuvat tehtävät voidaan automatisoida jää kehittäjille

enemmän aikaa keskittyä ohjelmistonkehitykseen sekä ongelmanratkaisuun. Samalla syntyy ympäristö, jossa kehittäjät ottavat vastuun työn tuloksesta, joka heijastelee positiivisesti laatuun ja kehitettävän järjestelmän vakauteen. Organisaatioissa, joissa jatkuvaa toimitusta hyödynnetään, on sillä havaittu olevan positiivinen vaikutus organisaatiokulttuuriin, kehitettävän ohjelmiston ja järjestelmän laatuun, ohjelmiston toimituksen tehokkuuteen sekä organisaation tehokkuuteen. (Forsgren, Humble ja Kim, 2018, s. 77 - 88)

Jatkuva toimitus mahdollistaa jatkuvan julkaisun (continuous delivery, CD). Samalla tavalla kuin jatkuva toimitus voidaan nähdä jatkumona jatkuvalle integroinnille, voidaan jatkuva julkaisu nähdä jatkuvan toimituksen seuraavana askeleena. Siinä missä jatkuva toimitus mahdollistaa julkaisun koska tahansa, jatkuvan julkaisun mallissa jokainen muutos todella julkaistaan. Sen tavoitteena on minimoida muutosten läpimenoaika (lead time) jolloin se on nopeammin loppukäyttäjien saavutettavissa. Näin kehitettävästä ohjelmistosta on mahdollista saada palautetta nopeammin. Nopean palautteen avulla kehittäjien on mahdollista oppia mitä loppukäyttäjät haluavat ja näin ohjelmistoa voidaan iteroida nopeammin loppukäyttäjien toiveiden mukaan. Sen ansiosta myös näiden muutosten julkaiseminen on nopeampaa ja helpompaa. (Davis & Daniels, 2016, s.39)

3 Pehmeät taidot

3.1 Määritelmä

Pehmeille taidoille ei ole olemassa yhtä yksiselitteistä määritelmää. Niiden määritelmä on aina kontekstisidonnainen. Taidot, jotka tietyssä kontekstissa määritellään koviksi taidoiksi, voidaan toisessa kontekstissa määritellä pehmeiksi taidoiksi. Niiden voidaan kuitenkin laajasti sanoa olevan kokoelma persoonallisuuspiirteitä, sosiaalisia taitoja, henkilökohtaisia ominaisuuksia ja optimismia. Esimerkiksi seuraavat piirteet luetaan pehmeisiin taitoihin: kommunikointikyky, itseohjautuvuus, ajanhallinta, vastuunottaminen, itsetunto, sosiaalisuus ja empatia. (Schulz, 2008, s. 147 - 149)

Tässä tutkimuksessa pehmeät taidot määritellään kahden ehdon kautta. Kyseessä on pehmeä taito, jos se täyttää toisen tai molemmat seuraavista ehdoista: kyseessä on henkilökohtainen ominaisuus tai ei kognitiivinen taito. Ei kognitiivisilla taidoilla tarkoitetaan taitoja, jotka eivät perustu opittuun tietoon.

Koska laajempi määritelmä on kontekstisidonnainen, määritellään tässä tutkimuksessa käytettäväksi kontekstiksi DevOps-organisaation päivittäinen toiminta sen muutosprosessin vaiheesta riippumatta.

3.2 Osa-alueet

Johtuen siitä, ettei pehmeille taidoille ole olemassa yksiselitteistä määritelmää on myös niiden eri osa-alueille olemassa useita erilaisia määritelmiä. Ne voidaan kuitenkin jakaa kahteen selkeään osa-alueeseen, henkilökohtaisiin- ja sosiaalisiin taitoihin, määritelmästä riippumatta.

Henkilökohtaisilla pehmeillä taidoilla tarkoitetaan niitä taitoja, jotka vaikuttavat siihen, miten yksilö toimii ilman kanssakäymistä muiden kanssa. Tällaisia ovat esimerkiksi ongelman ratkaisukykyyn, oppimiseen ja ajattelemiseen liittyvät taidot.

Sosiaalisilla taidoilla tarkoitetaan niitä taitoja, jotka vaikuttavat yksilön kanssakäymiseen muiden kanssa. Tällaisia taitoja ovat esimerkiksi kommunikointi taitoja tai muiden tunteiden tulkitsemista. (Cimatti, 2015)

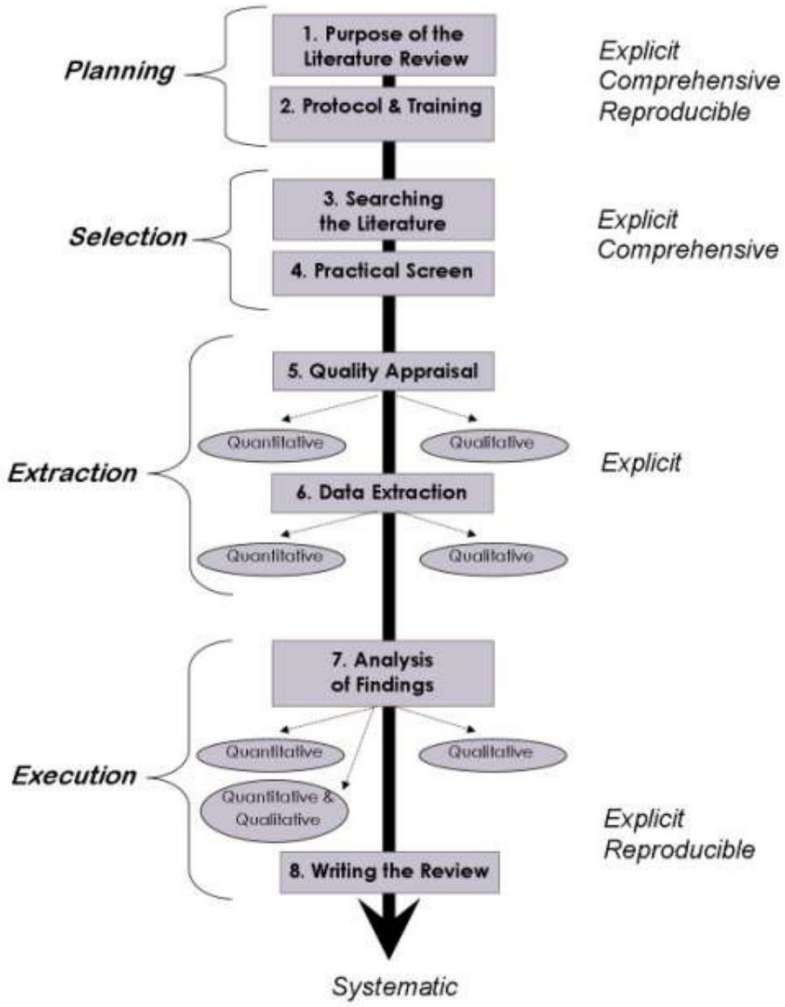
3.3 Pehmeät taidot ohjelmistokehityksessä

Ohjelmistokehitysorganisaatioihin haetaan yhdeksää eri pehmeää taitoa. Kommunikaatiotaitoja, henkilösuhdetaitoja, analyyttisiä- ja ongelmanratkaisutaitoja, kykyä työskennellä osana ryhmää, organisointitaitoja, kykyä oppia uusia asioita nopeasti, kykyä työskennellä itsenäisesti sekä innovatiivisuutta että avoimuutta ja kykyä adaptoitua muutokseen. Kommunikaatiotaidoilla tarkoitetaan henkilön kykyä viestiä muiden kanssa siten että viesti vastaanotetaan ja ymmärretään siten kuten se on tarkoitettu. Henkilösuhdetaidot ovat niitä taitoja, joiden avulla henkilö toimii vuorovaikutuksessa muiden kanssa niin suotuisissa kuin epäsuotoisissakin tilanteissa. Analyyttisillä- ja ongelmanratkaisutaidoilla tarkoitetaan henkilön kykyä ymmärtää, viestiä ja ratkaista monimutkaisiakin ongelmia sekä tehdä päätöksiä olemassa olevan tiedon perusteella. Kyky työskennellä ryhmässä tarkoittaa, että henkilö kykenee tehokkaasti työskentelemään muiden kanssa jakaen heidän kanssaan yhteisiä tavoitteita. Organisointitaidoilla tarkoitetaan henkilön kykyä käyttää resursseja kuten aikaa tehokkaasti. Uusien asioiden nopealla oppimisella tarkoitetaan tietotekniikan kontekstissa tapahtuvaa kehitystä eli sitä kykyä, kuinka nopeasti henkilö pystyy oppimaan uusia tekniikoita, konsepteja ja toimintatapoja. Kyky työskennellä itsenäisesti tarkoittaa sitä, että henkilö kykenee suoriutumaan tehtävistään ilman valvontaa. Innovatiivisuudella tarkoitetaan henkilön kykyä löytää ja luoda uusia luovia ratkaisuja erilaisiin ongelmiin. Avoimuus ja kyky adaptoitua muutokseen tarkoittaa kuinka hyvin henkilö vastaanottaa muutoksen ja pystyy sopeutumaan siihen.
(Ahmed, Capretz & Campbell, 2012)

Näiden taitojen kysyntä vaihtelee henkilöstöresurssien roolien mukaan. Esimerkiksi ohjelmistotestaajien kannalta taitoa työskennellä tiimissä ei pidetä erityisen tärkeänä, mutta jokaisessa muussa roolissa sen merkitystä korostetaan. Kommunikaatiotaitoja vaaditan työtehtävästä riippumatta, kun taas avoimuutta ja adaptoitumiskykyä ei pidetä missään roolissa erityisen tärkeänä. (Faheem, Capretz, Bouktif, Campbell 2013)

4 Menetelmä

Tutkimus menetelmäksi valikoitu systemaattinen kirjallisuuskatsaus. Systemaattinen kirjallisuus katsaus on tutkimusmuoto, jossa käydään läpi tutkittavan aihepiirin julkaisuja ja pyritään luomaan niiden perusteella tiivistelmä tutkimuksen kannalta olennaisesta ja mielenkiintoisesta aineistosta. Sitä käytetään usein tukevana tekniikkana tutkimuksen kontekstin asettamiseksi, jonka avulla selvitetään tutkimuksen alkuasetelma. Se on kuitenkin täysin pätevä myös itsenäisenä tutkimusmuotona. (Salminen, 2011, s.9)



Kuva 1 IS tutkimukseen adaptoitu systemaattisen kirjallisuuskatsauksen Fink-malli (Okoli & Schabram, 2010, s. 9)

Systemaattisessa kirjallisuuskatsauksessa olemassa olevaa tutkimusta kartoitetaan ja seulotaan läpi systemaattisesti sekä kriittisesti tutkimuksen kannalta olennaisen tiedon

löytämiseksi. Tässä tutkimuksessa menetelmänä käytettiin IS tutkimukseen adaptoitua Fink-mallia. Käytetty malli koostuu kahdeksasta eri vaiheesta: kirjallisuuskatsauksen tarkoituksen määrittely, protokollan määrittely, julkaisujen etsiminen, käytännön seula, laadullinen seula, tutkimuksen kannalta mielenkiintoisen tiedon talteen kerääminen, synteesi ja selonteko. (Okoli, Schabram, 2010, s. 6)

Tutkimusmenetelmän valintaan vaikutti erityisesti kaksi hyvin käytännön läheistä syytä. Ensinnäkin koska olemassa oleva tutkimus ei erottele kovia ja pehmeitä taitoja DevOps-kontekstissa, on olennaista koostaa mitä aiheesta tiedetään jo olemassa olevan tutkimuksen perusteella.

Toiseksi jokainen organisaatio on yksilöllinen ja näin ollen myös jokaisen organisaation lähtökohdat DevOps-viitekehyksen käyttöönottoon ovat erilaiset. Tämän takia myös jokainen DevOps-muutosprosessi on organisaatiokohtainen. Myös se miten organisaatiossa tulkitaan ja sovelletaan olemassa olevaa kirjallisuutta ja tutkimuksia on aina yksilöllistä. Tämän vuoksi myös organisaatioiden tarpeet henkilöstön pehmeiden taitojen suhteen saattavat poiketa toisistaan. On kuitenkin olemassa vain rajallinen määrä olemassa olevaa tutkimusta ja kirjallisuutta, josta jokaisen organisaation tulee hakea tietoa sekä vaikutteita. Näin suorittamalla tutkimus systemaattisena kirjallisuuskatsauksena, palaamalla tiedon alkulähteille, pyritään minimoimaan riskit siitä, että tutkimuksen tuloksista löytyisi sellaisia pehmeitä taitoja, jotka eivät ole oleellisia DevOps-organisaatiossa vaan liittyvät jonkun tietyn organisaation toimintaan.

4.1 Tarkoituksen määrittely

Tutkimuksen tarkoituksena on tutkimuskysymyksen mukaisesti löytää lista sellaisia pehmeitä taitoja, joita DevOps-organisaatiossa työskentelemiseen tarvitaan sekä samalla luoda ajankohtainen kirjallisuuskatsaus DevOps-käytännöistä.

4.2 Protokolla

Tutkimus aloitettiin hakemalla valituista kahdesta tietokannasta jokainen julkaisu, jonka tietokantojen hakukoneet palauttivat määritellyillä hakutermeillä.

Julkaisujen keräämisen jälkeen ne seulottiin läpi kaksivaiheisesti. Ensimmäisessä vaiheessa julkaisut kävivät läpi käytännön seulan. Käytännön seulan aikana tarkastetaan, onko julkaisu kirjoitettu sellaisella kielellä, jota tutkimuksen tekijä ymmärtää ja voiko sen perusteella vastata tutkimuskysymykseen. Seuraavassa vaiheessa eli laadullisessa seulassa, jäljelle jääneet julkaisut läpikäytiin ja pisteytettiin. Julkaisut, jotka eivät täyttäneet tutkimuksen kriteerejä jätettiin tutkimuksen ulkopuolelle.

Tämän jälkeen aloitettiin seulonnasta jäljelle jääneiden julkaisujen läpikäyminen. Tässä vaiheessa julkaisut luettiin useaan kertaan läpi. Läpikäymisen aikana julkaisujen kappaleista tehtiin sekä muistiinpanoja että tiivistelmiä mielenkiintoisten ja tutkimus kysymyksen kannalta oleellisista kohdista. Muistiinpanot kirjattiin systemaattisesti käyttäen kaavaa: ensimmäisenä kirjattiin ylös lähde, tämän jälkeen sivunumero, otsikko, avainsanat sekä päivämäärä, jolloin muistiinpano lisättiin. Tämän jälkeen kappaleesta tai kohdasta, josta muistiinpano kirjattiin ylös, kirjoitettiin tiivistelmä. Kaikki muistiinpanot sekä tiivistelmät kirjoitettiin käyttäen sähköisiä työkaluja, jotta niitä voidaan lajitella ja etsiä tehokkaammin tutkimuksen seuraavissa vaiheissa.

Läpikäymisen jälkeen alettiin julkaisuista luomaan synteesiä. Synteesissä luomiseksi käytettiin apuna luokittelu menetelmää, jonka avulla pyrittiin materiaalista löytämään kaikki DevOps-organisaatiossa työskentelyyn vaaditut pehmeät taidot. Luokittelun tuloksien analyysin perusteella kirjoitettiin lopullinen synteesi. Tutkimusprosessin selonteko on kirjattu kappaleeseen tulkinta. Sen avulla tutkimuksen toistaminen tarvittaessa on mahdollista.

4.3 Tutkimuksien haku

Hakuehdoiksi määriteltiin kaksi eri ehtoa. Ensimmäisenä ehtona, julkaisun abstraktissa on mainittava termi DevOps. Tällaiseen ehtoon päädyttiin koska kyseessä on varsin tuore ja vähän tutkittu ilmiö, tiedostettiin että olemassa olevaa tutkimusta on saatavilla vain rajallisesti. Rajaamalla hakutuloksia tällä tavoin pyrittiin varmistamaan, että mukaan saadaan myös sellaiset julkaisut, jotka potentiaalisesti käsittelevät vain yhtä DevOps-viitekehyksen osa-aluetta. Näin tutkimuksen kannalta potentiaalisesti mielenkiintoisten julkaisujen määrää kyettiin kasvattamaan. Tämä hakuehto kuitenkin mahdollisti

ääritapauksen, jossa jokainen tutkimuksessa käytettävä julkaisu käsittelisi jotain DevOps-viitekehyksen osa-aluetta. Tällainen tilanne lisäisi riskiä siitä, että tutkimuksessa jätettäisiin täysin huomioimatta jokin DevOps-viitekehyksen osa-alueista. Koska tässä tutkimuksessa keskityttiin siihen, millaisia pehmeitä taitoja DevOps-viitekehys kokonaisuutena vaatii, huomioitiin tämä riski käytännön seulassa. Tarkistamalla että jokainen julkaisu ei keskity yksittäiseen osa-alueeseen voitiin varmistaa, että jokainen viitekehyksen osa-alue on huomioitu.

Toisena ehtona on, että julkaisussa mainitaan pehmeät taidot tai jokin termeistä, joita kirjallisuudessa käytetään kuvaamaan taitoja, joita ei lasketa koviksi taidoiksi. Tällaisia termejä ovat: sosiaaliset taidot (social skills), luonteenpiirteet (personality traits), ei-kognitiiviset taidot (non-cognitive skills), ei-kognitiiviset kyvyt (non-cognitive abilities), luonne (character) tai sosiaalisemotionaaliset taidot (socio-emotional skills). Huomion arvoista on, että nämä termit eivät ole synonyymeja pehmeille taidoille, vaan niillä tarkoitetaan eri ominaisuuksia, jotka voidaan laskea pehmeiden taitojen alle. (Heckman & Kautz, 2012, s. 4)

Näin pyrittiin varmistamaan, että haun tuloksena saatavat artikkelit ovat relevantteja tämän tutkimuksen kannalta samalla huomioiden olemassa olevan tutkimuksen rajallinen määrä.

Tutkimuksessa käytettiin kahta eri tietokantaa julkaisujen etsimiseen ja hakemiseen. Päätös useamman tietokannan käyttämisestä tehtiin käytännön syistä. Sillä pyrittiin kasvattamaan tutkimuksen kannalta mielenkiintoisten ja vapaasti saatavilla olevien julkaisujen määrää. Hakukoneina käytettiin valittujen tietokantojen omia hakukoneita. Käytetyistä tietokannoista ensimmäinen on Association for Information Systems elektroninen kirjasto AIS Electronic Library. Muotoilemalla hakuehdot sen hakukoneen käyttämään syntaksiin muodostuu ehtolauseeksi:

```
abstract:devops AND ( soft skills ) OR ( social skills ) OR
( personality traits ) OR ( non-cognitive skills ) OR ( non-
cognitive abilities ) OR ( character ) OR ( socio-emotional
skills )
```

Haun tuloksena löytyi 29 ehtoihin sopivaa julkaisua. Niistä 23 oli vapaasti saatavilla.

Toisena tietokantana käytettiin Association For Computing Machinery ACM DL Digital Library tietokantaa. Muotoilemalla hakuehdot sen hakukoneen käyttämään syntaksiin muodostuu ehtolauseeksi:

```
[Abstract: devops] AND [[All: "soft skills"] OR [All: "social skills"] OR [All: "personality traits"] OR [All: "non-cognitive skills"] OR [All: "non-cognitive abilities"] OR [All: "character"] OR [All: "socio-emotional skills"]
```

Tämän haun tuloksena löytyi 12 julkaisua, joista jokainen oli vapaasti saatavilla. Hakuprosessin lopputuloksena tutkimusta varten haettiin yhteensä 35 julkaisua, jokainen vapaasti saatavilla oleva julkaisu. Jokaisen julkaisun on oltava esteettömästi saatavilla, jotta tutkimuksen laatua voidaan myöhemmin arvioida.

4.4 Käytännön seula

Käytännön seulan tarkoituksena on seuloa hakuprosessin tuloksena löytyneestä materiaalista läpi sellaiset julkaisut, joita ei selkeästi voida käyttää tutkimuksessa käytännön syistä. Seulan ensimmäisessä vaiheessa koottiin eri julkaisuista nimi, julkaisu ajankohta, tekijä tai tekijät. Seuraavassa vaiheessa tämän listan perusteella tarkasteltiin, ettei julkaisujen joukosta löydy duplikaatteja. Tämän jälkeen aloitettiin julkaisujen systemaattinen läpikäyminen, jonka aikana niiden tiivistelmät luettiin. Tässä vaiheessa tarkasteltiin kahta asiaa: onko julkaisu kirjoitettu joko suomeksi tai englanniksi ja voiko julkaisun perusteella vastata tutkimuskysymykseen. Tämän prosessin aikana tarkastettiin myös, keskittyykö julkaisu DevOps-viitekehykseen kokonaisuutena vai käsitteleekö se jotain viitekehyksen osa-aluetta. Näin tehtiin koska tutkimuksen hakuehdot mahdollistivat tilanteen, jossa jokainen haettu julkaisu käsittelisi yksittäistä DevOps-viitekehyksen osa-aluetta. Julkaisuja läpikäydessä tarkistettiin, että haetut julkaisut eivät koostuu pelkästään tällaisista julkaisuista.

Käytännön seulan läpäisi 20 julkaisua 35 julkaisusta. Joukossa ei ollut ainuttakaan duplikaattia. Valta-osa julkaisuista käsitteli DevOps-viitekehystä kokonaisuutena. Tutkimuksen ulkopuolelle jätetyistä julkaisuista yksi jätettiin pois koska se oli kirjoitettu kielellä, jota tutkija ei hallitse. Muiden 14 julkaisun perusteella ei ollut mahdollista vastata

tutkimuskysymykseen. Valtaosa näistä keskittyi käsittelemään teknisiä haasteita DevOps-kontekstissa eivätkä näin ottaneet kantaa vaadittuihin pehmeisiin taitoihin.

4.5 Laadullinen seula

Laadullisessa seulassa pyritään varmistamaan tutkimukseen mukaan otettavien julkaisujen laatu. Sen helpottamiseksi tutkimuksessa päädyttiin käyttämään ennalta tunnettuja tietokantoja. Näin pyrittiin varmistamaan, että kaikki sieltä löytyvät julkaisut ovat läpikäyneet jo standardoidun seulan.

Laadullisessa seulassa julkaisut luettiin ensimmäisen kerran läpi niiden arvioimiseksi. Siinä julkaisuja tarkasteltiin kolmen tutkimuksesta pois sulkevan kriteerin kannalta. Ensimmäinen kriteeri on, että tutkimuksen perusteella on mahdollista vastata tutkimuskysymykseen. Koska käytännön seulassa tutkimuksista luetaan vain tiivistelmät, ei niiden pohjalta voida sanoa varmasti onko tutkimus kysymykseen vastaaminen mahdollista julkaisun perusteella.

DevOps-terminä on toistaiseksi vailla yleisesti hyväksyttävää määritelmää ja sen ilmiönä voidaan katsoa kehittyvän jatkuvasti. Tulkinnat siitä mitä DevOps on, vaihtelivat enemmän vanhemmassa kirjallisuudessa ja tutkimuksessa johtuen olemassa olevan tutkimuksen ja kirjallisuuden puutteesta. Ilmiön kasvaessa ja kehittyessä sekä tutkimuksen lisääntyessä ovat tulkinnat siitä yhtenäistyneet. Tämän seikan huomioimiseksi haluttiin tutkimukseen mukaan vain sellaiset julkaisut, jotka on julkaistu viimeisen viiden vuoden aikana. Toinen tutkimuksesta pois sulkeva kriteeri onkin, että julkaisu on vanhempi kuin viisi vuotta, eli kirjoitettu ennen vuotta 2016.

Kolmas kriteeri on sidottu julkaisujen arviointiin. Kaikki julkaisut käytiin läpi niiden arvioimiseksi. Niiden pisteytys tapahtui arvioimalla niiden rakenteellista oikeellisuutta, sisällön pätevyyttä, yleistettävyyttä ja kuinka hyvin konteksti sopii tähän tutkimukseen. Pisteytys toteutettiin siten että julkaisut saivat yhden pisteen jokaista täytettyä kriteeriä kohden. Jos julkaisu ei täyttänyt yhtään näistä kriteereistä se sai nolla pistettä. Jos se täytti kaikki kriteerit, se sai neljä pistettä. Julkaisujen pisteytyksen kokonaisasteikko oli siis 0 – 4. Toisena pois sulkevana kriteerinä tuli jokaisen tutkimukseen mukaan otettavan julkaisun saada vähintään yksi piste pisteytyksessä.

Laadullisen seulan läpäisi 19 julkaisua 20 julkaisusta. Pois jätetyn julkaisun perusteella ei ollut mahdollista vastata tutkimuskysymykseen.

4.6 Katsaus

Käytännön ja laadullisen seulan läpäisi yhdeksäntoista julkaisua. Näistä julkaisuista 14 kappaletta on noudettu AIS Electronic Librarysta ja 5 kappaletta ACM DL Digital Librarysta. Julkaisuista 17 kappaletta on tutkimusartikkeleita, 1 kappale akateemisia julkaisuja ja 1 kappale konferenssijulkaisuja. Vanhimmat julkaisut ovat vuodelta 2017 ja tuoreimmat 2020. Taulukossa 1 näkyy kaikki katsaukseen mukaan otetut julkaisut. Katsauksena aikana julkaisuja läpikäytiin useaan kertaan etsien niistä tutkimuskysymyksen kannalta olennaisia tai mielenkiintoisia kohtia. Protokollan mukaisesti näistä kohdista tehtiin muistiinpanot ja niistä kirjoitettiin tiivistelmät.

TAULUKKO 1. Katsaukseen mukaan otetut julkaisut.

Julkaisu	**Tekijät**	**Vuosi**
A DevOps Collaboration Culture Acceptance Model	T. Masombuka, E. Mnkandla	2018
A New Form of Collaboration in IT Teams - Exploring the DevOps Phenomenon	A. Wiedemann	2017
A Qualitative Study of the Background, Skill Acquisition, and Learning Preferences of Software Testers	R. Florea, V. Stray	2020
Are you ready for DevOps - Required skill set for Devops teams	A. Wiedemann, Wiesche, M.	2018
BizDevOps and the Role of S-BPM	P. Forbrig	2018
Critical Success Factors of Continuous Practices in a DevOps Context	M. van Belzen, J. Trienekens, R. Kusters	2019
Dependency Management in Large-Scale Agile: A Case Study of DevOps Teams	V. Stray, N. B. Moe, A. Aasheim	2019
DevOps: Concepts, Practices, Tools, Benefits and Challenges	G. B. Ghantous, A. Gill	2017
Implementing the Planning Process within DevOps Teams to Achieve Continuous Innovation	A. Wiedemann, M. Wiesche, H. Gewald, H. Krcmar	2019

Julkaisu	Tekijät	Vuosi
Industry-Academy Collaboration in Teach-ing DevOps and Continuous Delivery to Software Engineering Students: Towards Improved Industrial Relevance in Higher Education	K. Kuusinen, S. Albertsen	2019
Integrating Development and Operations in Cross-Functional Teams - Toward a DevOps Competency Model	A. Wiedemann, M. Wiesche, H. Krcmar	2019
Invited Paper A Generalized, Enterprise-Level Systems Development Process Framework for Systems Analysis and Design Education	H. Topi, G. Spurrier	2019
IT Governance Mechanisms for DevOps Teams How Incumbent Companies Achieve Competitive Advantages	A. Wiedemann	2018
Leveraging DevOps for mission critical software	M. Gall, F. Pigni	2018
On Devops and Workforce Morale	J. Shropshire, B. Sweeney	2017
Productivity Gains of DevOps Adoption in an IT Team: A Case Study	M.A. Silva, J.P. Faustino, R. Pereira, M.M. da Silva	2017
SKI: A New Agile Framework that supports DevOps, Continuous Delivery, and Lean Hypothesis Testing	J. Saltz, A. Sutherland	2020
The Empire Strikes Back: The end of Agile as we know it?	J. Babb, J. Nørbjerg, D.J. Yates, L.J. Waguespack	2017
Understanding DevOps Education with Grounded Theory	C. Pang, A. Hindle, D. Barbosa	2020

4.7 Tulkinta

Synteesin luomiseksi käytettiin apuna luokittelumenetelmää. Sen avulla tutkittavasta materiaalista pyrittiin tuomaan esiin kaikki seljlaiset pehmeät taidot, jotka ovat sidoksissa DevOps-viitekehyksen mukaisiin käytäntöihin ja toimintatapoihin.

Luokittelumenetelmässä aineistosta poimituista tutkimuksen kannalta oleellisista kohdista muodostetaan ensin pelkistettyjä ilmauksia. Näiden perusteella luokiteltava aineisto luokitellaan ala- ja yläluokkiin sekä yhdistävään luokkaan. Luokkien muodostaminen tapahtuu etsimällä luokiteltavasta materiaalista eroja ja yhtäläisyyksiä, joiden perusteella luokat muodostetaan. Luokittelun lähtökohtana on aina tutkimuskysymys. (Tuomi & Sarajävi, 2002, s.118)

Koska luokittelun tavoitteena oli löytää tutkimusmateriaalista siinä esiintyvät pehmeät taidot, valittiin yhdistäväksi luokaksi pehmeät taidot. Pehmeille taidoille ei kuitenkaan ole olemassa yleisesti hyväksyttyä määritelmää, ne ovat sidottuja kontekstiin, jonka

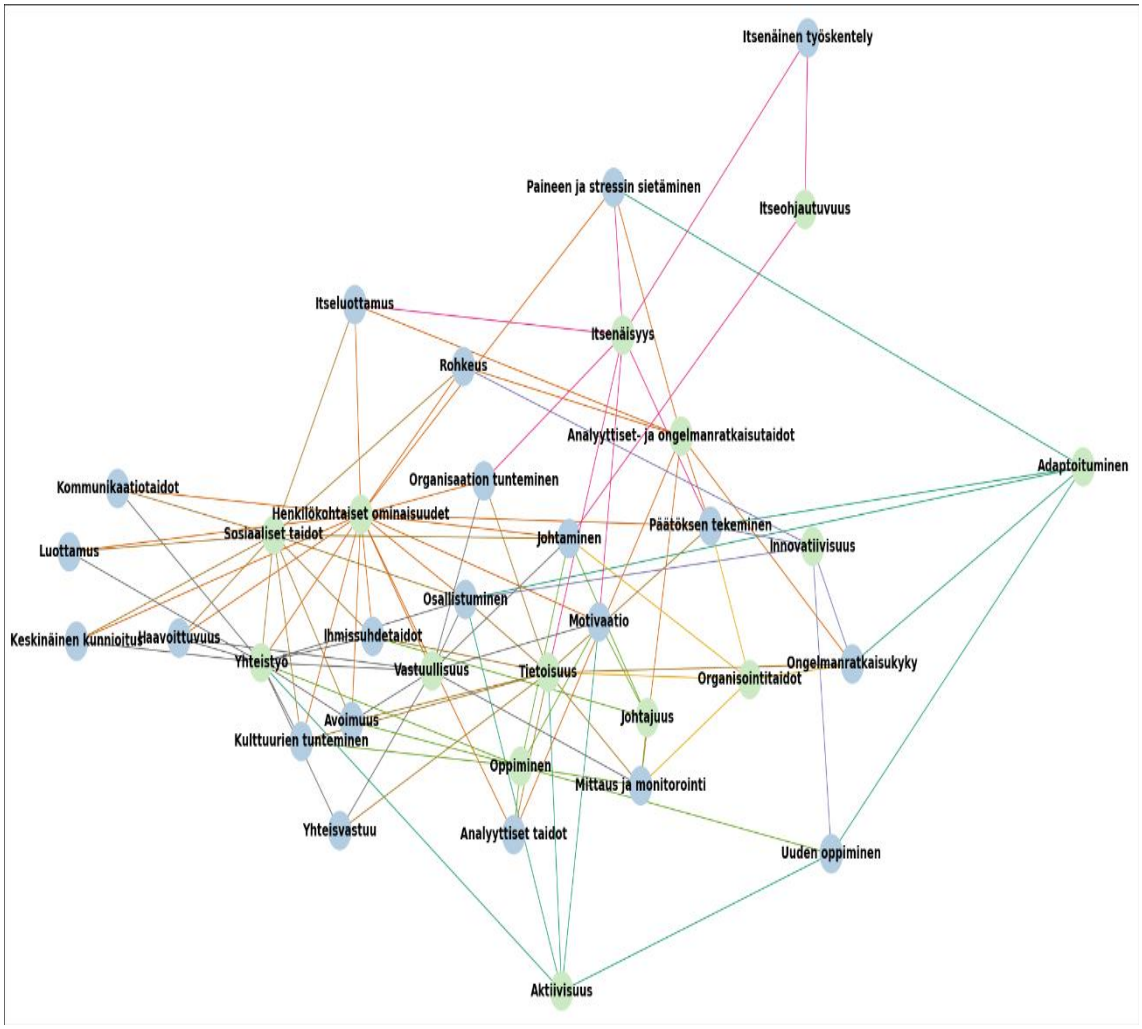
lisäksi niillä on riippuvuuksia toisiinsa. Esimerkiksi ihmissuhdetaidot voidaan jossain kontekstissa katsoa pehmeiksi taidoiksi, mutta ne ovat myös osa sosiaalisia taitoja. Tämän takia alaluokat edustavat joko yksittäisiä pehmeitä taitoja, niiden osaa tai henkilökohtaista ominaisuutta. Yläluokat taas edustavat pehmeitä taitoja tai laajempia kokonaisuuksia.

Sen jälkeen itse luokittelu prosessi aloitettiin listaamalla katsauksen aikana kirjoitetuista tiivistelmistä löytyneet tutkimuksen kannalta mielenkiintoiset ilmaukset ja fraasit eli pelkistetyt ilmaukset jokaista julkaisua kohden. Vertaamalla näiden pelkistettyjen ilmauksien eroja ja yhtäläisyyksiä toisiinsa muodostettiin jokaiselle tutkimuksen julkaisulle alaluokat. Vertaamalla taas alaluokkien välisiä eroja sekä yhtäläisyyksiä, muodostetiin yläluokat. Taulukosta 4 näkyy luokittelun aikana muodostettujen ilmausten sekä ala- ja yläluokkien kokonaismäärät sekä kuinka monta uniikkia luokkaa ja ilmausta luokittelun aikana syntyi.

TAULUKKO 2. Luokittelun tulokset

Luokittelun osa-alue	**Määrä**
Pelkistetyt ilmaukset	180
Uniikit pelkistetyt ilmaukset	142
Alaluokat	324
Uniikit alaluokat	24
Yläluokat	251
Uniikit yläluokat	14

Johtuen ala- ja yläluokkien määrittelyistä sekä pehmeiden taitojen keskinäisistä riippuvuuksista ei luokittelu sellaisenaan suoraan tarjoa listaa pehmeistä taidoista. Kuvassa 2 on kuvattu luokkien välisiä moniulotteisia riippuvuus suhteita. Uniikkien pehmeiden taitojen löytämiseksi prosessin viimeisessä vaiheessa suoritettiin luokittelun tuloksena syntyneiden luokkien seulonta. Tämän prosessin vaiheen tarkoituksena on karsia kaikkien luokittelun tuloksena syntyneiden luokkien joukosta sellaiset luokat pois, jotka eivät edusta pehmeitä taitoja, ovat niiden osia tai ovat duplikaatteja.



Kuva 2 Ylä- (merkitty vihreällä) ja alaluokkien (merkitty sinisellä) väliset suhteet.

Seulonta prosessi aloitettiin yhdistämällä uniikit ala- ja yläluokat keskenään. Näin saatiin 37 luokan lista, joka sisälsi 3 duplikaattia. Duplikaattien eliminoimisen jälkeen seulan seuraavassa vaiheessa jäljelle jääneistä 34 luokasta eliminoitiin 11 luokkaa. Näistä luokista kolme sisälsi useita eri pehmeitä taitoja, esimerkiksi analyyttiset- ja ongelmanratkaisutaidot voidaan jakaa kahteen eri taitoon. Loput kahdeksan eliminoitua luokkaa olivat joidenkin pehmeiden taitojen osa-alueita, esimerkiksi ihmissuhdetaidot, jotka ovat osa sosiaalisia taitoja. Seulontaprosessin lopputuloksena jäljelle jäi 23 uniikkia luokkaa: adaptoituminen, aktiivisuus, analyyttiset taidot, avoimuus, haavoittuvuus, innovatiivisuus, itseluottamus, itsenäisyys, itseohjautuvuus, johtajuus, kommunikaatiotaidot, kulttuurien tunteminen, luottamus, ongelmanratkaisukyky, oppiminen, organisointitaidot,

päätöksen tekeminen, paineen ja stressin sietäminen, rohkeus, sosiaaliset taidot, tietoisuus, vastuullisuus ja yhteistyö

Tämän jälkeen seulontaprosessin läpäisseet luokat läpikäytiin arvioiden täyttävätkö ne tässä tutkimuksessa käytetyn määritelmän pehmeistä taidoista: ei kognitiivinen taito tai henkilökohtainen ominaisuus. Jokainen jäljelle jääneistä luokista täyttää vähintään toisen näistä ehdoista, joten seulonnan tuloksena saatua listaa voidaan pitää myös vastauksena tutkimuskysymykseen.

Koska aiheesta ei ole aikaisempaa tutkimusta tai sitä ei ole helposti löydettävissä, on mahdollista, että tuloksena saatu lista on ensimmäinen koonti DevOps-viitekehyksen vaatimista pehmeistä taidoista.

Tästä huolimatta tuloksena saatua listaa voidaan verrata olemassa olevaan tietoon. Pehmeitä taitoja mainitaan hajanaisesti olemassa olevassa tutkimuksessa ja kirjallisuudessa. Tässä tutkimuksessa lähteinä käytetyssä kirjallisuudessa esiintyvät pehmeät taidot ovat: innovatiivisuus, luottamus, oppiminen, päätöksien tekeminen, paineen ja stressin sietäminen sekä yhteistyötaidot. (Davis ja Daniels, 2019; Forsgren, Humble ja Kim, 2018; Ravichandran, Taylor, Waterhouse, 2016)

Tämän lisäksi tuloksia voidaan verrata yhdeksään ohjelmistokehitysorganisaatioissa kysyttyihin pehmeisiin taitoihin: kommunikaatiotaidot, henkilösuhdetaidot, analyyttiset- ja ongelmanratkaisutaidot, kyky työskennellä ryhmässä, organisointitaidot, kyky oppia uusia asioita nopeasti, kyky työskennellä itsenäisesti, innovatiivisuus, avoimuus ja kyky adaptoitua muutokseen. (Faheem, Capretz, Bouktif, Campbell 2013)

Verrattaessa tutkimuksen tuloksena saatuja pehmeitä taitoja DevOps-kirjallisuudessa esiintyviin pehmeisiin taitoihin, huomataan että tutkimuksen tulokset sisältävät jokaisen edellä mainituista taidoista. Vaikka kyseessä on vain pieni otos olemassa olevasta kirjallisuudesta, voidaan tuloksien todeta olevan linjassa olemassa olevan tiedon kanssa. Verrattaessa taas ohjelmistokehitysorganisaatioissa kysyttyihin taitoihin, nähdään että tulokset ovat linjassa myös niiden kanssa.

Tutkimuksen kannalta mielenkiintoisia tuloksia ovat myös yhdeksän kappaletta taitoja, jotka eivät kuulu kumpaankaan vertailu otokseen. Nämä ovat: aktiivisuus, haavoittuvuus,

itseluottamus, itseohjautuvuus, johtajuus, kulttuurien tunteminen, rohkeus, tietoisuus, vastuullisuus.

4.8 Synteesi

Synteesissä poimitut tiedot kootaan yhteen, järjestetään, vertaillaan ja niiden perusteella kirjoitetaan kvalitatiivinen meta-analyysi. Synteesi voidaan jakaa seuraaviin askeleisiin: kirjallisuuden tunteminen, sen ymmärtäminen, sen jakaminen osiin, sen soveltaminen, sen synteesi ja arviointi. Arvioinnin rooli on erityisen tärkeä. Siinä menetelmä kuvataan siten että se on toistettavissa. Tällä pyritään varmistamaan prosessin tuloksena syntyvän synteesin laatu. (Okoli, Schabram, 2010, s. 31-32)

Synteesin lähestymistavaksi valittiin metasynteesi. Siinä tutkimusaineistoa läpikäydään huolellisesti vertaillen, etsien niistä poikkeavuuksia sekä yhtäläisyyksiä. Tämän prosessin tarkoituksena on saada kattava kokonaiskuva tutkimusaineistosta ja tiivistämään tutkittavaa materiaalia vastavuoroisen käännöksen tekemiseksi. Vastavuoroisen käännöksen tavoitteena on tulkitsevan synteesin luominen. Siinä materiaalia verrataan keskenään ja havaittuja yhtäläisyyksiä muutetaan yhden muotoiseksi metaforia käyttämällä. Tätä prosessia ei kuitenkaan tule tehdä väkisin, jotta aineistosta löytyvä mahdollinen uusi tieto ei jää huomaamatta. (Salminen, 2011)

Schryen (2015) toteaa että järjestys, jossa aineistoa lähdetään synteesissä purkamaan, tulisi olla aina konseptikeskeinen eikä esimerkiksi kronologinen tai kirjoittajakeskeinen.

Tässä tutkimuksessa valittiin lähestymistavaksi luokittelu, jonka avulla kerätty aineisto läpikäytiin sen sisältämän tiedon esiin tuomiseksi. Luokittelun tuloksena koko tutkimusaineistosta löydettiin 142 uniikkia pelkistettyä ilmausta, 25 uniikkia alaluokkaa ja 15 uniikkia yläluokkaa.

Synteesin runko rakentuu luokittelun tuloksena syntyneiden yläluokkien ympärille. Taulukosta 2. on nähtävissä julkaisun numero ja taulukosta 3. tätä numeroa vastaavat yläluokat. Yläluokat kuvaavat DevOps-organisaatioissa oleellisimpia pehmeitä taitoja tai useita pehmeitä taitoja sisältäviä joukkoja. Pehmeät taidot ovat usein vahvasti sidoksissa toisiinsa. Esimerkiksi sosiaalisten taitojen alle kuuluu useita erilaisia taitoja kuten ihmissuhdetaidot, kommunikaatiotaidot sekä itseluottamus. DevOps-kontekstissa tarvitaan

kuitenkin usein laajempia kokonaisuuksia kuten sosiaalisia taitoja. Vaikka ihmissuhdetaidot ovat tärkeitä, ne eivät yksin riitä vaan niitä tukemaan vaaditaan muita samaan yläluokkaan kuuluvia taitoja kuten kommunikaatiotaitoja.

TAULUKKO 3. Julkaisujen numerointi

Julkaisu	**Nro**
A DevOps Collaboration Culture Acceptance Model	1
A New Form of Collaboration in IT Teams - Exploring the DevOps Phenomenon	2
A Qualitative Study of the Background, Skill Acquisition, and Learning Preferences of Software Testers	3
Are you ready for DevOps - Required skill set for Devops teams	4
BizDevOps and the Role of S-BPM	5
Critical Success Factors of Continuous Practices in a DevOps Context	6
Dependency Management in Large-Scale Agile: A Case Study of DevOps Teams	7
DevOps: Concepts, Practices, Tools, Benefits and Challenges	8
Implementing the Planning Process within DevOps Teams to Achieve Continuous Innovation	9
Industry-Academy Collaboration in Teach-ing DevOps and Continuous Delivery to Software Engineering Students: Towards Improved Industrial Relevance in Higher Education	10
Integrating Development and Operations in Cross-Functional Teams - Toward a DevOps Competency Model	11
Invited Paper A Generalized, Enterprise-Level Systems Development Process Framework for Systems Analysis and Design Education	12
IT Governance Mechanisms for DevOps Teams How Incumbent Companies Achieve Competitive Advantages	13
Leveraging DevOps for mission critical software	14
On Devops and Workforce Morale	15
Productivity Gains of DevOps Adoption in an IT Team: A Case Study	16
SKI: A New Agile Framework that supports DevOps, Continuous Delivery, and Lean Hypothesis Testing	17
The Empire Strikes Back: The end of Agile as we know it?	18
Understanding DevOps Education with Grounded Theory	19

TAULUKKO 4. Eri julkaisuista löytyneet yläluokat.

Julkaisun numero	Adaptoituminen	Aktiivisuus	Analyyttiset- ja ongelmanratkaisutaidot	Henkilökohtaiset ominaisuudet	Innovatiivisuus	Itsenäisyys	Itseohjautuvuus	Johtajuus	Oppiminen	Organisointitaidot	Sosiaaliset taidot	Tietoisuus	Vastuullisuus	Yhteistyö
1	X	X	X	X	X	X				X	X	X	X	X
2		X	X	X		X		X	X	X	X	X	X	X
3		X	X	X		X		X	X	X	X	X	X	X
4	X	X	X	X	X	X	X	X	X	X	X	X	X	X
5		X		X		X		X	X	X	X	X	X	X
6	X	X	X	X	X	X	X	X	X	X	X	X	X	X
7	X	X	X	X	X	X	X	X	X	X	X	X	X	X
8	X	X	X	X	X	X		X	X		X	X	X	X
9	X	X	X	X	X	X	X	X	X	X	X	X	X	X
10	X	X	X	X	X	X	X	X	X	X	X	X	X	X
11	X	X	X	X	X	X	X	X	X	X	X	X	X	X
12	X	X	X	X	X	X	X	X	X	X	X	X	X	X
13	X	X	X	X	X	X	X	X	X	X	X	X	X	X
14	X	X	X	X	X	X	X	X	X	X	X	X	X	X
15	X	X	X	X	X	X	X	X	X	X	X	X	X	X
16	X	X	X	X	X	X	X	X	X	X	X	X	X	X
17	X	X	X	X	X	X	X	X	X	X	X	X	X	X
18	X	X	X	X	X	X	X	X	X	X	X	X	X	X
19	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Adaptoituminen on luontainen osa työskentelyä DevOps-organisaatiossa. Nopea reagointi asiakkaiden toiveisiin, teknisiin haasteisiin ja uuteen informaatioon vaatii, että henkilöstö kykenee adaptoitumaan erilaisiin tilanteisiin päivittäin.

Tarve adaptoitumiselle alkaa heti muutosprosessin alkuvaiheessa, kun organisaation eri toimintoja yhdistetään tehokkaamman yhteistyön ja tiedonkulun tavoittelemiseksi. Sen sijaan että ohjelmiston elinkaaren eri vaiheet tapahtuisivat eri toimintojen alaisuudessa, ne yhdistetään saman organisatorisen yksikön alle kommunikaation siiloutumisen estämiseksi. Tätä prosessia kutsutaan organisaation siilojen purkamiseksi. (Masombuka & Mnkandla, 2018)

Tästä seuraa useimmiten tarve muuttaa jo olemassa olevia rooleja siten että ne sopivat muuttuneeseen organisaatiomalliin. Muuttuneet roolit vaativat organisaation henkilöstöltä kykyä adaptoitua uusiin rooleihinsa oppimalla uusia osa-alueita ohjelmistokehitysprosessin elinkaaren eri vaiheista. Erityisesti ohjelmistokehityksessä työskentelevien henkilöiden tulee pystyä omaksumaan laaja-alaista osaamista yhteen osa-alueeseen erikoistumisen sijaan. (Wiedemann & Wiesche, 2018; Wiedemann, Wiesche & Krcmar 2019)

Kattava ymmärrys erilaisista konsepteista käytettävien tekniikoiden takana onkin olennaisempaa kuin yksittäisten työkalujen tai kielten hallitseminen. (Pang, Hindle, & Barbosa 2020)

Roolien muuttumisen seurauksena myös eri toimintojen vastuualueet muuttuvat. Ohjelmistokehityksen ja ylläpidon tulee kantaa yhdessä vastuuta koko ohjelmiston elinkaaresta. (Shropshire & Sweeney, 2017)

Jaetun vastuun lisäksi tämän uuden toiminnon tulee jakaa arvot, kannustimet ja tavoitteet. (Silva, Faustino, Pereira & Mira Da Silva 2018).

Myös ohjelmistokehityksen prosessit tulevat todennäköisesti muuttumaan. DevOps-käytäntöjen mukaisesti tämä tarkoittaa siirtymistä ketteriin menetelmiin. Stray, Moe ja Saltz ja Sutherland (2020) Aasheim (2019), Topi ja Spurrier (2019) nostavat esille ketterien menetelmien adaptoitumista vaativan luonteen. Tarve adaptoitua syntyy luonnostaa nopeissa sykleissä tapahtuvan iteratiivisen ohjelmistokehityksen sivutuotteena.

Ajatus oppimisesta ja nopeasti muutoksiin reagoimisesta on keskeisessä osassa ketterien menetelmien filosofisia periaatteita. (Babb, J. ja muut, 2017)
Koko organisaation laajuinen muutosprosessi ei kuitenkaan ole kivuton vaan sen seurauksena saattaa syntyä muutosvastarintaa sekä pelkoa ja epäilystä uusia toiminta tapoja kohtaan. (van Belzen, Trienekens, & Kusters 2019)
Olemassa olevan kulttuurin muokkaaminen on haasteellista. Se vaatii aiempaa aktiivisempaa kommunikaatiota olemassa olevien asenteiden muokkaamiseksi sekä mahdollisten pelkojen ja epäilysten poistamiseksi. (Ghantous & Gill, 2017)
Muutosprosessi tapahdu hetkessä vaan todennäköisesti osa toiminnoista adaptoituu uusiin toiminta tapoihin muita nopeammin. (Gall & Pigni, 2018)
Ennen kuin käytännöt vakiintuvat, tarvitaan organisaatiossa aktiivisempaa johtamista ja kommunikaatiota. Muutosprosessin alkuvaiheessa eri toiminnot saattavat tarvita aktiivista ulkopuolista tukea ja seurantaa ennen kuin ovat täysin adaptoituneet muutokseen. (Silva, Faustino, Pereira & Mira Da Silva, 2018)

Adaptoitumisen lisäksi DevOps-organisaatioissa vaaditaan henkilöstöltä jatkuvaa aktiivisuutta. Avoimen kommunikaation periaatteen mukaisesti DevOps-tiimien jäsenten tulee jatkuvasti tietoisia ohjelmiston kehityksen tilasta ja jakaa aktiivisesti tietoa siitä mitä ovat tekemässä. (Masombuka & Mnkandla, 2018)
Niissä työskentelee useiden eri ohjelmistokehityksen osa-alueiden asiantuntijoista, jotka tekevät yhteistyötä keskenään. Kommunikaatio vaatii siis myös mentoroivaa otetta ja samanaikaisesti aktiivista oppimista vastapuolelta. Työtä ei myöskään tehdä yksin vaan ongelmia ratkotaan aktiivisesti ryhmässä. (Florea. & Stray, 2020; Wiedemann & Wiesche, 2018)
Aktiivisen kommunikoinnin tarve ei rajoitu vain tiimin sisäiseen viestintään, vaan sitä tarvitaan myös eri toimintojen välillä. Kun ohjelmistokehitykseen otetaan mukaan kaupallisia toimintoja, vaaditaan kaikilta osapuolilta aktiivista kommunikaatiota sekä organisointia yhteistyön sujuvuuden takaamiseksi. (Forbig, 2018)

Aktiivisen työskentelykulttuurin ylläpitämiseksi ja yhteistyön tehostamiseksi organisaation henkilöstö käy päivittäin lukuisa a*d hoc*-keskusteluja sekä tiimin sisällä että eri toimintojen kesken. (Stray, Moe &. Aasheim, 2019)

Aktiivisuutta vaaditaan myös tiimien koordinoinnin ja riippuvuuksien hallinnan suhteen. Riippuvuuksilla tarkoitetaan sellaisia resursseja, joiden puuttuessa ohjelmistokehityksen jotain osa-aluetta ei voida edistää. Tiimien tavoitteiden potentiaaliset ristiriidat tulee tiedosta jo suunnitteluvaiheessa. Tämän lisäksi suunnittelua tulee tehdä aktiivisesti jatkuvan suunnittelun periaatteen mukaisesti. Riippuvuuksien tilaa tulee aktiivisesti seurata ja niiden tarpeet tulee tiedostaa etukäteen. Tämä on erityisen tärkeää niissä organisaatioissa, jotka ovat DevOps-muutosprosessin alkutaipaleella tai organisaatioissa, joissa useampi tiimi työskentelee saman tuotteen parissa. (van Belzen, Trienekens & Kusters, 2019; Wiedemann, Wiesche, Gewald & Krcmar, 2019)

Aktiivisuuden lisäksi DevOps-organisaatioissa vaaditaan analyyttisiä ja ongelmanratkaisutaitoja. Ne liittyvät olennaisesti ohjelmistokehitykseen, järjestelmien suunnitteluun ja niiden ylläpitoon. Niiden merkitys kasvaa nopeatempoisessa ympäristössä, jossa toistuva manuaalinen työ pyritään automatisoimaan. Organisaatioissa, joissa kyetään hyödyntämään automaatiota kattavasti, henkilöstö voi keskittyä aktiivisesti uuden kehittämiseen ja ongelmakohtien ratkomiseen. Sekä uuden kehittäminen että olemassa olevien ongelmakohtien ratkominen vaativat organisaation henkilöstöltä analyyttisia- ja ongelmaratkaisutaitoja. Tämän vuoksi niiden merkitys korostuu DevOps-kontekstissa. Kattava automaation hyödyntäminen ei kuitenkaan poista järjestelmän ylläpidon ja monitoroinnin tarvetta.

Analyyttiset taidot ovatkin vahvasti sidoksissa ylläpitoon liittyviin tehtäviin DevOps-kontekstissa. Niiden varaan rakentuu henkilöstön kyky monitoroida, hallinnoida sekä kehittää kehitettävää järjestelmää. Ne liittyvät myös oleellisesti yksilön kykyyn abstraktoida kompleksisia ongelmia niiden ratkaisemiseksi. (Wiedemann & Wiesche, 2018)

Analyyttiset- ja ongelmanratkaisutaitojen rooli korostuu erilaisissa poikkeustilanteissa, joissa DevOps-tiimin tulee kyetä reagoimaan nopeasti tekniseen ongelmaan. (Masombuka & Mnkandla, 2018)

Jokaisen ohjelmistokehitystiimin jäsenen odotetaan osallistuvan vikatilanteissa ongelman ratkaisemiseen ja analysointiin. Tätä pidetään erityisen tärkeänä koska nopean ohjelmistokehityksen lisäksi yksi DevOps-käytäntöjen hyödyntämisellä tavoiteltava tekijä on kehittävän ohjelmiston luotettavuuden parantaminen. (Wiedemann, Wiesche & Krcmar, 2019)

Onkin tärkeää, että erityisesti ohjelmistokehitys tiimien jäsenet ymmärtävät korkean tason tekniset konseptit kehittävästä ja ylläpidettävästä järjestelmästä. DevOps-tiimien jäseniltä vaaditaankin kattavaa kokemusta erilaisista tekniikoista ja työtehtävistä sekä hyvää itseluottamusta ja ongelmanratkaisukykyä. (Pang, Hindle & Barbosa, 2020)

Tarvittaessa henkilöstön on kyettävä oppimaan uusia konsepteja ja kieliä nopeasti. Tämän lisäksi tiimien jäsenten tulee pyrkiä oppimaan virheistään ja yhdessä parantamaan ohjelmistokehitysprosessia aktiivisesti jatkuvan oppimisen periaatteen mukaisesti. (Babb, Nørbjerg, Yates & Waguespack, 2017; Topi & Spurrier, 2019)

Itseluottamuksen lisäksi työskentely DevOps-organisaatiossa vaatii henkilöstöltä useita muita henkilökohtaisia ominaisuuksia. Henkilökohtaiset ominaisuudet ovat kokoelma yksilön ominaisuuksia kuten itseluottamus, rohkeus, motivaatio tai paineen ja stressin sietäminen.

DevOps-kontekstissa esiin nousevat yhteistyötä, kommunikaatio, itsenäisyyttä ja itseohjautuvuutta sekä ongelman ratkaisua ja päätöksentekoa tukevat ominaisuudet. Yhteistyön kannalta on olennaista, että henkilöstö luottaa toisiinsa, kykenee kommunikoimaan tehokkaasti sekä omaa vahvat ihmissuhdetaidot. Ongelmien ratkomiseksi nopea tempoisessa DevOps-organisaatiossa on taas olennaista, että henkilöstö kykenee rauhallisesti analysoimaan tilannetta, ratkaisemana ongelmia ja kykenee tekemään päätöksiä paineenkin alla. (Wiedemann & Wiesche, 2018)

Yhteistyön sujuvuuden takaamiseksi tarvitaan myös hienovaraisempia ominaisuuksia kuten empatiakykyä ja kulttuurien ymmärtämistä ja tuntemusta. Eri sidosryhmien välisessä yhteistyössä ja kommunikaatiossa on olennaista kyetä ymmärtämään vastapuolen tilanne ja kulttuuritausta, josta tämä tulee. Nämä seikat auttavat oikean

kommunikaatiostrategian valitsemissa ja edesauttavat luottamuksen rakentamisessa osapuolten välille. (Babb, Nørbjerg, Yates, & Waguespack, 2017; Forbrig, 2018; Stray, Moe & Aasheim, 2019)

Henkilökohtaiset ominaisuudet määrittävät myös, kuinka DevOps-muutosprosessi otetaan organisaatiossa vastaan. Yksittäisten avainhenkilöiden heikko motivaatio, luontainen taipumus skeptisyyteen, epäluottamukseen sekä negatiiviseen ajatteluun saattaa vaikeuttaa prosessia ja jarruttaa kulttuurin kehittymistä. Muutoksen voidaan kokea lisäävän stressiä ja painetta sekä pelkoa siitä, kuinka kykenee adaptoitumaan muutokseen, statuksen menettämisestä organisaatiossa, roolin muuttumisesta tai tarpeesta. (Stray, Moe & Aasheim, 2019)

Tällaisessa tilanteessa tarvitaan aktiivista johtamista. Aktiivinen johtaminen vaatii kuitenkin rohkeutta, paineen ja stressin sietämistä sekä itseluottamusta, jotta kykenee vaikuttamaan muutosvastarintaan. Skeptisten ja epäluuloisten organisaation jäsenien motivoiminen vaatii myös pitkäjänteisyyttä, empatiakykyä, kulttuurien ymmärtämistä sekä ongelmanratkaisukykyä ja ihmissuhdetaitoja. (Gall & Pigni, 2018)

DevOps-organisaatioihin haetaankin ennakkoluulottomia motivoituneita yrittäjähenkisiä kiinnostuneita tiimipelaajia, jotka eivät pelkää muutosta tai paineen alla työskentelyä ja haluavat aktiivisesti oppia uutta. Erityisen tärkeää on virheiden myöntäminen, niistä oppiminen sekä niistä opittujen asioiden jakaminen. Tämä vaatii kuitenkin rohkeutta ja luottamusta muita kohtaan asettaa itsensä haavoittuvaan asemaan muiden edessä. Toisaalta tämä taas vaatii muilta taas kykyä samaistua toisen asemaan ja kykyä oppia muidenkin virheistä. (Kuusinen & Albertsen, 2019; Wiedemann 2018)

Yksi DevOps-muutosprosessin tavoitteista on luoda ympäristö, joka edesauttaa innovaatioiden syntymistä. Yhdistämällä korkean automaation asteen, DevOps-viitekehyksen mukaisen kulttuurin, kattavan mittaamisen ja seurannan sekä tiedon jakamisen, organisaatiot pyrkivät edesauttamaan innovaatioiden syntymistä. (Wiedemann, 2017)

DevOps-tiimit muodostuvat ohjelmiston elinkaaren eri vaiheiden asiantuntijoista. Tätä osaamista saattaa täydentää kehitettävän tuotteen kohde toimialan hallitseva erikoisasiantuntija tai kaupallisesta toiminnosta tuleva tuotteen omistajana. Tällainen asetelma

ruokkii innovaatiota kasvaneen kommunikaation ja yhteistyön ansiosta. (Wiedemann, 2017; Wiedemann, Wiesche, Gewald & Krcmar, 2019)

Ketterien menetelmien mukaisesti iteratiivisesti tapahtuva ohjelmistokehitys mahdollistaa nopean reagoinnin loppukäyttäjiltä saadun palautteen mukaan ja näin edesauttaa innovaatioiden synnyttämistä ja jakamista nopeasti eteenpäin. Jatkuvan innovaation edesauttamiseksi yhteistyö eri toimintojen välillä on ensiarvoiseen tärkeää. Tuotteen omistajan (product owner) tulee osallistua DevOps-tiimin toimintaan aktiivisesti tuoden asiakasrajapinnasta kerätyn tiedon lähemmäs varsinaista kehitysprosessia. Ideaalisessa tilanteessa tällä henkilöllä on kokemusta jostain muusta toiminnosta ja pystyy näin auttamaan kehitystiimiä innovoimaan täydentämällä tiimin osaamista omalla asiantuntijuudellaan. Tuotteen omistajan rooli korostuu erityisesti suunnitteluvaiheessa. Nopean reagoinnin mahdollistamiseksi on ensiarvoisen tärkeää, että vastuualueet, riippuvuudet sekä iteraatiossa toteutettavien tehtävien laajuus suunnitellaan huolellisesti. (Wiedemann, Wiesche, Gewald & Krcmar, 2019)

Innovaatiot DevOps-kontekstissa liittyvät useimmiten abstraktien ongelmien ratkomiseen itsenäisesti tai yhteistyössä muiden kanssa. Kyky analysoida ja ratkoa erilaisia ohjelmistokehityksen aikana eteen tulevia ongelmia on olennainen taito jokaiselle DevOps-tiimin jäsenelle. (Wiedemann & Wiesche, 2018)

Itseohjautuvuus on olennainen taito DevOps-organisaation henkilöstölle. Itseohjautuvuus nousee näkyviin erityisesti päätöksenteon, suunnittelun, johtamisen ja itsensä johtamisen yhteydessä.

DevOps-tiimit ovat autonomisia yksiköitä, jotka koostuvat monialaisista osaajista, jotka työskentelevät keskenään kohti yhteistä päämäärää. Vaikka tiimillä onkin useimmiten määritelty johtohahmo, osallistuvat kaikki tiimin jäsenet kuitenkin aktiivisesti päätöksen tekoon tuoden mukanaan oman erikoistumisalueensa. Jokaisen tiimin jäsenen tulee kyetä kuitenkin tekemään päätöksiä tarvittaessa itsenäisesti. Tämä korostuu erityisesti poikkeustilanteissa, joissa tiimien jäsenien on kyettävä tekemään nopeita päätöksiä ilman ulkoista tukea ja paineen alla. Ideaalinen DevOps-tiimin jäsen onkin itseohjautuvat, nauttii päätöksenteosta ja ei pelkää vastuunottamista. Tiimin johtohahmon on taas

kyettävä ennakoimaan potentiaalisia ongelmia, riippuvuuksia ja hallinnoitava käytössä olevia resursseja. (Wiedemann, 2018; Wiedemann & Wiesche, 2018; A. Wiedemann, Wiesche, Gewald & Krcmar, 2019)

Itsensä johtaminen liittyy vahvasti vastuun kantamiseen sekä kommunikaatioon. Jotta työntekijä suoriutuu vastuulleen annetuista tehtävistä, tulee tämän kyetä työskentelemään tarvittaessa itsenäisesti. Itsenäinen työskentely sisältää työtehtävän suunnittelun, siihen tarvittavien resurssien koordinoinnin, tehtävän implementaation sekä tiedon jakamisen. Tämän lisäksi työntekijän tulee tarvittaessa tunnistaa potentiaaliset riippuvuudet ja kommunikoida niiden tarpeesta aktiivisesti. (Stray, Moe & Aasheim, 2019)

Osana DevOps-muutosprosessia organisaatioissa siirrytään pois vahvasti ylhäältä ohjatusta korkean hierarkian mallista kohti matalan hierarkian mallia, joka muodostuu autonomisista yksiköitä. Tästä huolimatta DevOps-organisaatioissa halutuimpien ja etsityimpien taitojen joukossa mainitaankin johtajuus. (Wiedemann, Wiesche & Krcmar, 2019; Kuusinen & Albertsen, 2019)

Johtajuuden merkitys korostuu DevOps-muutosprosessin alkuvaiheessa, kun uudet käytännöt eivät ole vielä vakiintuneet organisaatiossa. Suuret muutokset organisaation toimintatavoissa, käytännöissä ja rooleissa aiheuttavat todennäköisesti muutosvastarintaa, skeptisyyttä sekä pelkoa ja epätietoisuutta. Näiden ongelmien ratkaisuksi vaaditaan aktiivista johtamista, jossa muutoksesta kommunikoidaan avoimesti ja aktiivisesti. (Shropshire & Sweeney, 2017; Gall & Pigni, 2018)

Vaikka DevOps-tiimit ovatkin autonomisia yksiköitä, joiden jäsenet työskentelevät itseohjautuvasti niille valitaan yleensä johtohahmo (team lead). Tämä voi olla joko tuotteen omistaja tai yksi tiimin jäsenistä. Tämän johtohahmon vastuulla toimia tiimin edustajana ja fasilitoida kehitettävän ohjelmiston seuraavan iteraation suunnittelu. Tämän suunnitteluprosessin aikana tiimin johtajan tulee huolehtia toteutettavien toimintojen laajuudesta, mahdollisista riippuvuuksista sekä vastuu alueiden jakamisesta. (Stray, Moe, Aasheim 2019; Wiedemann, Wiesche, Gewald & Krcmar, 2019)

Johtajuus on kuitenkin DevOps-kontekstissa enemmän palvelevaa johtamista kuin hallinoivaa. Johtajan tehtävä on pyrkiä raivaamaan tiimin tieltä esteitä ja edesauttaa tiimiä

onnistumaan omassa tavoitteessaan. Timin jäsenille ei anneta suoraan käskyjä vaan heitä pyritään hienovaraisesti ohjaamaan oikeaan suuntaan. (Babb, Nørbjerg, Yates & Waguespack, 2017)

Tällainen palvelevan johtamisen malli on sisään rakennettu Scrum-viitekehykseen. Määritelmän mukaisesti scrum master toimii palvelevana johtajana ja kehitystiimin valmentajana, jonka tehtävänä on auttaa tiimiä eteenpäin raivaamalla potentiaalisia esteitä heidän tieltään. (Saltz & Sutherland, 2020)

Tällaisissa organisaatioissa scrum master keventää tiimin muiden johtavien osien kuten tiiminjohtajan ja tuotteenomistajan taakkaa osallistumalla suunnitteluun, riippuvuuksien hallintaan sekä koordinointiin ja kommunikaatioon muiden tiimien välillä. Näin scrum master pystyy raivaamaan esteitä kehitystiimin tieltä ja antaa tiimin jäsenille mahdollisuuden keskittyä itse ohjelmistokehitykseen. (Stray, Moe, Aasheim 2019)

Oppiminen on keskeinen osa DevOps-kulttuuria. Uuden oppimiseen kannustetaan ja sitä odotetaan tapahtuvan jatkuvasti eri muodoissa. DevOps-tiimien jäsenet pyrkivätkin opiskelemaan uusia taitoja ja hyödyntävät aktiivisesti tilaisuuksia oppia jotain uutta. Tämä oppiminen saattaa olla muodollista opiskelua kuten verkkokursseja tai konferenseihin osallistumista. Vaihtoehtoisesti se voi olla tiimin sisällä tapahtuvaa oppimista. Tällainen oppiminen voi tapahtua useissa eri muodoissa. Se voi olla kokeneemman tiimin jäsenen mentorointia, palautteen vastaanottamista, koodi katselmointia, pari ohjelmointia, vasta opitun tiedon aktiivista jakamista, virheistä oppimista tai oman erikoisosaamisen jakamista muille tiimin jäsenille. (Wiedemann, Wiesche & Krcmar, 2019)

Vikatilanteista pyritään myös aktiivisesti oppimaan analysoimalla mikä aiheutti poikkeustilanteen. Tätä käytäntöä kutsutaan *Post Mortem*-analyysiksi. Niissä kehittävästä ohjelmistosta vastaava tiimi läpikäy yksityiskohtaisesti syitä siihen miksi tällainen tilanne pääsi syntymään ja kuinka se voitaisiin välttää tulevaisuudessa. (Shropshire & Sweeney, 2017)

Organisaatioissa, joissa hyödynnetään Lean-ajattelua, voidaan pyrkiä analysoimaan asiakkaiden reagointia tuotteeseen testaamalla Lean-hypoteeseja. Niiden ideana on luoda hypoteesi, kuinka loppukäyttäjä potentiaalisesti reagoi johonkin muutokseen

kehitettävässä ohjelmistossa. Tämän jälkeen ohjelmistokehitys tiimi luo minimaalisen testattavan toteutuksen ja analysoi kuinka loppukäyttäjät reagoivat tähän. (Saltz & Sutherland, 2020)

Ketterien menetelmien mukaisesti prosessia pyritään aktiivisesti kehittämään. Tasaisin väliajoin tapahtuvassa retrospektiivipalaverissa (retrospective) tiimin jäsenet läpikäyvät edellisissä kehityssykleissä tapahtuneita asioita. Niistä tarkastellaan onnistuneita ja epäonnistuneita asioita oppimisen mahdollistamiseksi. Löydetyt asiat analysoidaan mahdollisimman tarkasti, jotta kehitystiimi pystyy arvioimaan syitä minkä takia jokin asia on onnistunut tai epäonnistunut. Analysoinnin aikana opittujen tekijöiden avulla prosessia pyritään hienosäätämään ennen seuraavaa ohjelmistokehityssykliä. Vaikka jatkuva integraatio ja julkaisu ovat keskeisiä tekijöitä DevOps-käytännöissä, voidaan joissain organisaatioissa järjestää myös demotilaisuus ohjelmistokehityssyklin päätteeksi. Tämän tilaisuuden tarkoituksena on esitellä kehitettävän ohjelmiston tilaa eri sidosryhmille ja vastaanottaa palautetta sekä kehitysehdotuksia tiimin ulkopuolelta. Tämän tilaisuuden aikana saadusta palautteesta saatuja oppeja voidaan hyödyntää seuraavien kehityssyklien ja ohjelmiston suunnittelussa. (Saltz, Sutherland, 2020)

Babb, Nørbjerg, Yates, & Waguespack (2017) ilmaisevatkin huolensa siitä, että DevOps-käytäntöjen painotus automaation ja jatkuvaan integraatioon sekä julkaisuun heikentää ketterille menetelmille tyypillistä oppimista ja reflektointia ohjelmistokehityssyklien välillä. He argumentoivat, että ketterien menetelmien adaptoiminen tällaiseen ympäristöön saattaa johtaa niiden eroosioon.

DevOps-tiimit ovat itseorganisoituvia yksiköitä jotka muodostuvat monialaisista osaajista joilla on yhteinen näkemys siitä miten ohjelmistokehityksen tehtävät tulisi organisoida. Organisointikyky onkin oleellinen pehmeätaito DevOps-organisaatioissa. (Babb, Nørbjerg, Yates, & Waguespack, 2017; Wiedemann, Wiesche & Krcmar, 2019; Pang, Hindle & Barbosa, 2020)

Organisointikyky on oleellinen osa itsenäistä työskentelyä. Jotta henkilö pystyy ratkomaan ongelmia ja tekemään loogisia päätöksiä paineen alla muiden puolesta tulee hänen kyetä organisoimaan saatavilla olevaa tietoa ja käytettävissä olevaa aikaa. Ollakseen

muiden tiiminjäsenien luottamuksen arvoinen tulee tiimin jäsenen tarvittavan asiantuntemuksen ja tietotaidon omaamisen lisäksi kyetä organisoimaan omaa työtään ja saatavilla olevia resursseja mahdollisimman tehokkaasti. (Masombuka & Mnkandla, 2018)
Sen merkitys korostuu erityisesti ohjelmistokehityssykliä edeltävässä suunnittelussa, jossa DevOps-tiimien jäsenten tulee kyetä ennalta näkemään mahdollisia riippuvuuksia, resurssitarpeita tai muita implementaatiota estäviä tekijöitä sekä organisoida loppukäyttäjiltä saatu palaute syklin aikana suoritettaviksi tehtäviksi. Näiden tehtävien laajuus ja potentiaaliset sivuvaikutukset tulee kyetä arviomaan, jotta tehtävät voidaan organisoida käytettäville henkilöstöresursseille. Tämän lisäksi on olennaista tehdä päätöksiä siitä mitkä tehtävistä otetaan mukaan tulevaan ohjelmistokehityssykliin niin että ne olisivat kaikki suoritettavissa ennen seuraavaa sykliä. (Wiedemann, Wiesche, Gewald & Krcmar, 2019)
Epäonnistuminen resurssien organisoinnissa tarkoittaa käytännössä resurssien hukkaan heittämistä. Ohjelmistokehitysprojekteissa esiintyy kolmen tyyppisiä riippuvuuksia. Tieto riippuvuus tarkoittaa sitä, ettei ohjelmistokehitys on riippuvainen jonkin tiedon saamisesta. Prosessi riippuvuudella tarkoitetaan tilannetta, jossa ohjelmistokehitystä ei voida jatkaa ennen kuin jokin aiemmin prosessi on valmis. Viimeisimpänä ovat resurssi riippuvuudet. Niissä ohjelmistokehitystä ei kyetä jatkamaan ennen kuin tarvittava resurssi on saatavilla. Jos ohjelmistokehityksen kannalta riippuvuutta ei saada ajoissa, jarruttaa tämä pahimmillaan koko ohjelmistokehitystä. Jo yksittäinen tällainen viivästys saattaa aiheuttaa koko projekti viivästymisen, jolla voi olla vaikutuksia koko organisaation mittakaavassa. (Stray, Moe & Aasheim, 2019)

Yhteistyö ja ihmisten välinen kanssakäyminen ovat merkittävässä roolissa DevOps-viitekehystä. Noin puolet kehittäjien työajasta kuluu yhteistyössä muiden kanssa. Ongelmia ratkotaan *ad hoc*-keskusteluissa sekä tiimin sisällä että muiden toimintojen kanssa. (Ghantous. & Gill, 2017; Florea, & Stray , 2020; Stray, Moe & Aasheim, 2019)
Sosiaaliset taidot ovat välttämättömiä sekä tiimin sisäisen että muiden sidosryhmien kanssa tapahtuvan kanssakäymisen mahdollistamiseksi. Ne luovat perustan yhteistyölle, joka on olennainen osa DevOps-viitekehyksen mukaista työskentelyä. Sosiaalisilla

taidoilla tarkoitetaan yksilön kykyä muodostaa ja ylläpitää sosiaalisia suhteita. Wiedemannin ja Wieschen (2018) mukaan DevOps-organisaatiossa työskentelyyn vaaditaan seitsemää erilaista sosiaalista taitoa. Ensimmäisenä he mainitsevat kyvyn vastaanottaa palautetta, joka on olennaista oppimisen ja henkilökohtaisen kehittymisen mahdollistamiseksi. Seuraavana listalla on kyky kommunikoida yleisön mukaisesti. Tämä on olennaista eri toimintojen yhteistyön kannalta ja sen merkitys korostuu teknisten ja kaupallisten toimintojen tehdessä yhteistyötä. Luontaisena jatkuman on kykyä oppia kulttuurien väliseen kommunikaatioon vaadittavia kommunikaatiotaitoja. Seuraavana listalla on halukkuus oppia uusia taitoja. Se on olennainen taito tiimin jokaiselle jäsenellä vastuunjakamisen ja yhteistyön ylläpitämiseksi. Sen puuttuessa yhteistyö saattaa kärsiä koska vastuu ei tällöin jakaudu tasaisesti tiimin sisällä ja näin ollen johtaa huonompaan tehokkuuteen. Sen jälkeen listalta löytyy halukkuus työskennellä tarvittaessa reaktiivisesti sekä ennaltaehkäisevästi. Nämä tekijät vaikuttavat oleellisesti siihen miten erilaiset poikkeustilanteet hoidetaan ja toisaalta, kuinka niitä voidaan pyrkiä välttämään tekemällä etukäteen valintoja, joilla riskejä pyritään minimoimaan. Viimeiset kaksi kohtaa listalla ovat vahvasti sidoksissa toisiinsa: halukkuus jakaa tietoa ja halukkuus oppia muiden jakamasta tiedosta. Nämä ovat olennaisia taitoja oppimisen ja henkilökohtaisen kehityksen kannalta mutta myös tiimin sisäisen yhteistyön toimivuuden takaamisen kannalta. Jotta tiimi voi iteratiivisesti kehittää toimintaansa ja tuotettavaa ohjelmistoa tai palvelua on olennaista, että tiimin jäsenet oppivat toisiltaan jatkuvasti. Tämä liittyy oleellisesti vahvoihin kommunikaatiotaitoihin. Ideaalisessa tilanteessa tiimin jäsenet kommunikoivat, luottavat toisiinsa ja rakentavat kulttuuria kommunikaation päälle. (Wiedemann & Wiesche, 2018)

DevOps-organisaation henkilöstön tulee aina pyrkiä olemaan aktiivisesti tietoinen siitä mitä organisaatiossa, että sen toimintaympäristössä tapahtuu. Tietoisuudella DevOps-kontekstissa tarkoitetaan henkilön luontaista kykyä ylläpitää tilannekuvaa. Tällaista tietoisuutta vaaditaankin koko DevOps-organisaation henkilöstöltä.

Avoimen kommunikaation periaatteen mukaan jokainen DevOps-tiimin jäsen tulee pitää

tietoisena kehitettävän ohjelmiston tilasta jatkuvasti sen koko elinkaaren ajan. Käytännössä tämä tarkoittaa sitä, että jokaisen tiimin jäsenen tulee aktiivisesti kertoa oman työnsä vaiheesta. Vastavuoroisesti muiden kehitystiimin jäsenten tulee aktiivisesti pyrkiä seuraamaan valittuja viestintä kanavia ollakseen tietoisia mitä ympärillä tapahtuu. Tiedon jakaminen on olennaista useista eri syistä. Sen avulla voidaan välttää ohjelmistokehityksessä tapahtuvia konfliktitilanteita, se toimii yhteistyön mahdollistajana ja samalla yhteen sitoo ohjelmistokehitystiimiä. Tämän lisäksi ohjelmistokehitystiimin jäsenien pitää olla tietoisia palvelun ylläpitoon liittyvistä asioista kuten palvelun tila, käännös- ja julkaisuautomaation tila sekä käytettävissä olevista resursseista. (Masombuka & Mnkandla, 2018)

Jos kehitystiimillä on ulkoisia riippuvuuksia organisaation muihin toimintoihin tai organisaation ulkopuolelle, on tietoisuus niiden tilasta ensiarvoisen tärkeää. Tällaisia riippuvuuksia voivat olla jotkin artefaktit, jotka puuttuessaan estävät ohjelmistokehityksen tai haittaavat sitä. Tietoisuus tällaisten riippuvuuksien tilasta on erityisen merkityksellistä suunniteltaessa seuraavaa ohjelmistokehityssykliä. (Wiedemann, Wiesche, Gewald & Krcmar, 2019)

Näiden lisäksi on ohjelmistokehitystiimin oltava aktiivisesti tietoisia kehitettävää ohjelmistoa ja/tai palvelua koskevista laatuvaatimuksista kuten palvelulaatusopimuksista. Tähän liittyen on olennaista, että kehitystiimin jäsenet ovat tietoisia siitä mitä seuraa, jos näitä laatuvaatimuksia ei kyetä täyttämään. (Wiedemann & Wiesche, 2018)

Tietoisuutta vaaditaan myös DevOps-kulttuurinmuutosprosessin kannalta. Erityisen tärkeää on, että organisaation jäsenet ovat tietoisia käynnissä olevasta muutoksesta ja siitä mitä se tarkoittaa päivittäisen työskentelyn kannalta. Tämän vuoksi *ad hoc*-muutosprosessi on mahdollista vain tilanteessa, jossa koko ohjelmistokehitysorganisaatio on tietoisia muutoksen tuomista eduista. (Wiedemann, 2018)

Epätietoisuus vallitsevasta tilanteesta saattaa aiheuttaa muutosvastarintaa ja erilaisia pelkotiloja muutosta koskien, joka hidastaa muutoksen jalkautumista ja jolla saattaa olla pitkäaikaisia ei toivottuja sivuvaikutuksia. Tällaiset ongelmat on kuitenkin mahdollista eliminoida aktiivisella tiedon jakamisella. (Gall & Pigni, 2018)

Kyky kantaa vastuuta on olennainen taito DevOps-organisaatioissa. Se on vahvasti sidoksissa yhteen DevOps-kulttuurin keskeiseen teemaan, yhteiseen omistajuuteen.
Yhteinen omistajuus tarkoittaa, että koko DevOps-tiimi jakaa vastuun kehittämästään ohjelmistosta sekä sen tarjoamisesta. Tämä antaa kehitystiimille vapaat kädet tehdä päätöksiä järjestelmän suhteen mutta samalla sitoo kehittäjät vastuuseen omasta työnjäljestään. Vikatilanteissa tiimin jäsenten on kyettävä tarvittaessa tekemään vastuullisia päätöksiä muun tiimin puolesta itsenäisesti. (Wiedemann, Wiesche & Krcmar, 2019; Silva, Faustino, Pereira & Mira Da Silva, 2018)
Vastuu palvelusta tarkoittaa sitä, että tiimi on myös vastuussa kehitettävän ohjelmiston ja/tai palvelun suunnittelusta. Koska tiimi on vastuussa kehitettävän tuotteen implementoinnista, julkaisusta ja ylläpidosta tulee tämä huomioida suunnitteluvaiheessa. Tiimin on kannettava vastuu kehitettävän ohjelmiston laadusta, tietoturvasta sekä skaalautuvuudesta. Suunnitteluvastuu kuitenkin samalla edesauttaa vastuun kantamisessa. Kun suunnitteluprosessi tapahtuu tiimin sisällä voivat sen jäsenet luottaa siihen, ettei uusia vaatimuksia ilmaannu tiimin ulkopuolelta ilman että ne otetaan huomioon suunnittelussa. (Wiedemann, Wiesche, Gewald & Krcmar, 2019)
Yhteisen vastuun kantamisen kannalta on olennaista, että jokainen DevOps-tiimin jäsen ymmärtää eri roolit ja niiden vastuualueet. Jokaisen tulee ymmärtää roolien vastuualueet, mitä tämän roolien täyttämiseksi odotetaan, miten rooleissa suoriutumista mitataan ja arvioidaan sekä mitä niissä epäonnistumisesta seuraa. Joissain organisaatioissa oppimisen tehostamiseksi voidaan vastuualueita vaihtaa tiimin jäsenten kesken syklisesti siten että tiimin jäsenet oppivat eri roolit ja niiden vaatimukset. (Masombuka & Mnkandla, 2018; Wiedemann, 2017)

Yhteistyökyky on ehdottoman tärkeä pehmeätaito DevOps-organisaation henkilöstölle. Yhteistyö on koko DevOps-kulttuurin kulmakivi, jonka päälle koko kulttuuri rakentuu ja samalla yksi muutosprosessin keskisistä tavoitteista.
Yhdistämällä eri toimintoja useampia eri ohjelmistonelinkaaren eri vaiheiden asiantuntijoita sisältäviksi autonomisiksi DevOps-tiimeiksi organisaatiot pyrkivät kasvattamaan yhteistyön määrää. Tällainen organisaation uudelleen järjestely edesauttaakin eri

asiantuntijoiden välisen kommunikaation ja yhteistyön syntymistä, kun tiimin jäsenet työskentelevät yhteistä tavoitetta kohti. (Wiedemann, Wiesche, & Krcmar, 2019)

DevOps-kontekstissa yhteistyö muodostuu neljästä eri tekijästä: kommunikaatiosta, vastuusta, kunnioituksesta ja luottamuksesta. Kommunikaatio rakentuu avoimen kommunikaation periaatteen päälle, jossa DevOps-tiimin jäsenten odotetaan jakavan tietoa avoimesti keskenään ja pitävän muut ajan tasalla omasta työstään. Vastuulla tarkoitetaan yhteisvastuuta järjestelmän toimivuudesta sekä kehitettävästä ohjelmistosta. Kunnioitus tässä yhteydessä tarkoittaa DevOps-tiimin jäsenten keskinäistä kunnioitusta. Jokaista jäsentä tulee kunnioittaa arvokkaana tiimin jäsenenä ja arvostaa tämän kontribuutiota yhteisen maalin saavuttamiseksi. On myös olennaista, että eri rooleissa toimivia henkilöitä kunnioitetaan tämän periaatteen mukaisesti siten, ettei yksikään rooleista ole arvokkaampi kuin toinen. Tiimin jäsenten välinen luottamus on tärkeää yhteistyön kannalta. Tehokkaan yhteistyön takaamiseksi tiimin jäsenten on kyettävä asettamaan itsensä välillä haavoittuvaan asemaan ja luottamaan siihen, etteivät muut tiimin jäsenet hyväksikäytä tätä heikkoutta. Tämä toimii samalla avoimen kommunikaation mahdollistajana. (Masombuka & Mnkandla, 2018)

Yhteistyö ei kuitenkaan rajoitu vain DevOps-tiimien sisäiseen toimintaan. DevOps-organisaatioissa eri toiminnot saattavat tehdä yhteistyötä keskenään. Tämän vuoksi onkin oleellista, että tiimien jäsenet ymmärtävät sen toimialueen, jossa toimivat ja siihen liittyvät prosessit. (Wiedemann & Wiesche, 2018)

4.9 Tulokset

Tutkimuksen tuloksena tutkittavista julkaisuista tunnistettiin yhteensä 23 kappaletta uniikkeja pehmeitä taitoja. Tulokset sekä niiden kuvaukset löytyvät taulukosta 5. Kuvaukset pyrkivät havainnollistamaan mitä yksittäiset pehmeät taidot tarkoittavat DevOps-kontekstissa. Ne eivät kuitenkaan ole tarkkoja kuvauksia tuloksena saaduista yksittäisistä pehmeistä taidoista.

TAULUKKO 5. Tutkimuksen tulokset

Pehmeätaito	**Kuvaus**
Adaptoituminen	Kyky adaptoitua muutokseen
Aktiivisuus	Kyky aktiiviseen kommunikaation, aulius osallistua
Analyyttiset taidot	Kyky kerätä ja analysoida tietoa sekä käyttää tätä tietoa
Avoimuus	Avoimuus uusia asioita kohtaan,
Haavoittuvuus	Kyky asettaa itsensä haavoittuvaan asemaan
Innovatiivisuus	Kyky innovoida, ajatella laatikon ulkopuolelta
Itseluottamus	Alttius luottaa itseensä
Itsenäisyys	Kyky työskennellä itsenäisesti ilman ulkoista valvontaa
Itseohjautuvuus	Kyky ohjata omaa toimintaansa ilman ulkoista valvontaa
Johtajuus	Kyky johtaa muita
Kommunikaatiotaidot	Kyky kommunikoida tehokkaasti
Kulttuurien tunteminen	Eri kulttuurien ymmärtäminen ja hyväksyminen
Luottamus	Kyky luottaa muihin sekä olla itse luottamuksen arvoinen
Ongelmanratkaisukyky	Kyky ratkoa ongelmia ja ongelmatilanteita
Oppiminen	Kyky ja alttius oppia uutta
Organisointitaidot	Kyky organisoida omaa työtään, resursseja ja projekteja
Päätöksen tekeminen	Kyky tehdä päätöksiä
Stressin sietäminen	Alttius sietää ulkoista painetta ja stressiä
Rohkeus	Kyky tehdä rohkeita valintoja
Sosiaaliset taidot	Kyky luoda ja ylläpitää sosiaalisia suhteita
Tietoisuus	Tietoisuus itsestä ja ulkopuolisista tekijöistä
Vastuullisuus	Kyky kantaa vastuuta
Yhteistyö	Kyky tehdä yhteistyötä muiden kanssa

5 Diskussio

Tutkimuksen tavoitteena oli löytää sellaiset pehmeät taidot, joita DevOps-viitekehys vaatii organisaation henkilöstöltä. Tutkimuksen tuloksena löytyi 23 kappaletta pehmeitä taitoja, joten tutkimusta voidaan pitää onnistuneena. Olemassa olevaa tutkimusta pehmeistä taidoista, joita DevOps-viitekehys vaatii organisaation henkilöstöltä ei ole olemassa tai se ei ole helposti saatavilla. Sen sijaan olemassa oleva tutkimus keskittyy tutkimaan DevOps-viitekehyksen vaatimia taitoja kokonaisuutena erottelematta kovia ja pehmeitä taitoja. Tämä tutkimus laajentaakin olemassa olevaa tietoa keskittymällä puhtaasti pehmeisiin taitoihin.

5.1 Tulosten merkitys

DevOps-viitekehyksen suosio on kasvanut viime vuosina huomattavasti ja sen kasvun oletetaan jatkuvan vielä lähitulevaisuudessa. Sen suosiota ovat nostaneet menestystarina markkinoiden edellä kävijöistä kuten Googleta ja Amazonilta. Saman aikaisesti organisaatiot etsivät uusia toimintatapoja toimintojen siirtyessä sähköiseen muotoon digitalisaation seurauksena. (Global Industry Analysts Inc., 2020)

Samanaikaisesti kasvava automaation hyödyntäminen eri toimialoilla on kasvattanut kiinnostusta pehmeisiin arvoihin. Kasvava automaation aste tarkoittaa sitä, että organisaatioiden henkilöstöstä yhä pienempi osa tekee sellaisia työtehtäviä, joissa suoriutuminen vaatii pääsääntöisesti vain kovia taitoja. Jäljelle jäävät tehtävät nojaavat kasvavassa määrin pehmeisiin taitoihin kuten ongelmanratkaisuun ja ryhmätyöhön. (Bishop, 2019)

Tässä uudenlaisessa työympäristössä ihmiset työskentelevät monimuotoisissa ryhmissä, jotka muodostuvat eri kulttuureista, ikäryhmistä ja joiden jäsenet saattavat sijaita eri puolilla maapalloa. Tällaisessa ympäristössä henkilöstöltä vaaditaan kattavasti erilaisia pehmeitä taitoja kuten kulttuurien ymmärrystä, kykyä ryhmätyöhön ja tehokkaaseen kommunikaatioon. (Dean ja East, 2019, s.1)

Samanaikaisesti DevOps-viitekehyksen suosion kasvaessa yksityisellä sektorilla on yliopistoilla painetta kyetä huomioimaan ilmiö omissa koulutusohjelmissaan. Pelkästään DevOps-organisaatioissa useimmiten käytettyjen työkalujen ja toimintatapojen

opettaminen ei yksin riitä vaan opetusohjelmissa tulee kyetä huomioimaan myös DevOps-kulttuuri. Kulttuurin syntymisen mahdollistavia tekijöitä ovat jaetut arvot, tavoitteet sekä toimintatavat, onnistumisen määrittely, yhteinen omistajuus, kunnioitus, luottamus, jatkuva kommunikaatio, kokeileminen ja oppiminen. (Kuusinen ja Albertsen, 2019)

Nousevan suosion seurauksena tarve tutkimukselle ja tiedolle nousee käytäntöjä soveltavien organisaatioiden määrän kasvaessa. DevOps-viitekehys itsessään sisältää useita teemoja, jotka korostava pehmeiden taitojen merkitystä kuten jatkuva oppiminen, avoin kommunikaatio, yhteistyö ja empatia. Tulevaisuudessa vaikuttavien tekijöiden ennustetaankin olevan varsin hyvin linjassa sekä DevOps-viitekehyksen mukaisen toiminnan kanssa kuin myös tämän tutkimuksen fokuksen kanssa. Koko työelämää koskeva muutos, jonka myötä pehmeiden taitojen merkitys korostuu todennäköisesti lisääkin kiinnostusta niihin myös DevOps-kontekstissa. Tutkimuksen tuloksia voidaankin potentiaalisesti hyödyntää sekä akateemisessa ympäristössä että kaupallisten toimijoiden taholta. Tutkimuksen tulokset tarjoavat pohjan tulevalle tutkimukselle koskien pehmeitä taitoja DevOps-kontekstissa. Samanaikaisesti ne tarjoavat lähtökohdan, jota oppilaitokset voivat hyödyntää tulevien opetusohjelmien ja kurssien suunnittelussa. Kuusinen ja Albertsen (2019) nostavat esiin DevOps-kulttuurin ja sen mahdollistavien tekijöiden opetuksen. Nämä kulttuurin mahdollistavat tekijät ovat vahvasti sidoksissa pehmeisiin taitoihin, joiden opetuksessa tutkimuksen tuloksia voidaan potentiaalisesti hyödyntää.

Kaupallisten toimijoiden on mahdollista hyödyntää tutkimuksen tuloksia lukuisissa eri sovelluskohteissa. Tuloksia voidaan käyttää esimerkiksi henkilöstöhallinnon toimesta työkaluna rekrytointiprosessissa, henkilöstön koulutusta suunniteltaessa sekä organisaation rakenteen muutoksien suunnitteluun. Toisaalta ne mahdollistavat myös yksittäisten henkilöiden soveltuvuuden arvioimisen DevOps-organisaatioon.

5.2 Tulosten luotettavuus

Tutkimuksessa on pyritty noudattamaan hyvä tieteellistä käytäntöä eli sen aikana on noudatettu tieteellisen yhteisön tunnustamia toimintatapoja. Sen aikana on pyritty rehellisyyteen, yleiseen huolellisuuteen ja tarkkuuteen tutkimustyössä, tulosten

tallentamisessa, niiden esittämisessä ja tutkimuksen sekä tulosten arvioinnissa. Tutkimuksessa käytettiin tiedonhankinta, tutkimus- ja arviointimenetelminä tieteellisen tutkimuksen kriteerit täyttäviä eettisesti kestäviä menetelmiä. Siinä on otettu huomioon muiden tutkijoiden työ ja saavutukset kunnioittavalla ja asianmukaisella tavalla viittaamalla heidän julkaisuihinsa asianmukaisella tavalla sekä antamalla niille asiaan kuuluvan tunnustuksen. (Tutkimuseettinen neuvottelukunta, 2021)

Tutkimusprosessin kaikki vaiheet on dokumentoitu ja tutkimusprosessi on suoritettu tämän dokumentaation mukaisesti. Tiedon hankintaan käytettiin tunnettuja, tunnustettujen organisaatioiden tietokantoja julkaisujen laadun takaamiseksi. Tutkimuksessa on käytetty julkaisuja, jotka ovat vapaasti saatavilla, jotta tutkimuksen tulokset voidaan tarvittaessa todentaa.

5.3 Tulokset suhteessa olemassa olevaan tutkimukseen

Olemassa olevaa tutkimusta, joka keskittyisi ainoastaan siihen millaisia pehmeitä taitoja DevOps-viitekehys vaatii organisaation henkilöstöltä, ei ole olemassa tai sitä ei ole helposti saatavilla. Sen sijaan tutkimusta verrataan olemassa olevaan tutkimukseen siitä, millaisia taitoja DevOps-organisaatiossa vaaditaan. Olemassa olevassa tutkimuksessa ja kirjallisuudessa käsitellään pehmeitä arvoja hajanaisesti, joka mahdollistaa tämän tutkimuksen vertaamisen olemassa olevaan tutkimukseen. Vaikka nämä tutkimukset eivät keskity puhtaasti pehmeisiin taitoihin voidaan niiden avulla todentaa tämän tutkimuksen tulosten olevan samassa linjassa olemassa olevan tutkimuksen kanssa.

5.4 Suositukset soveltamiseen

Tutkimuksen tuloksia on mahdollista soveltaa sellaisenaan. Koska kyseessä on jatkuvasti kehittyvä ja laaja-alainen ilmiö, on sovellettaessa suositeltavaa käyttää tukena myös muita julkaisuja. Riippuen sovelluskohteesta tulee pohtia ovatko kaikki listatut taidot relevantteja tässä käyttötapauksessa. Tämän tutkimuksen tulokset ovat koko organisaation kattava listaus niistä pehmeistä taidoista, joita työskentelyyn tarvitaan. Tutkimus ei ota kantaa siihen millaisissa taitoja tarvitaan eri rooleissa. Tämän lisäksi jokainen

organisaatio on yksilöllinen, joka on syytä huomioida sovellettaessa. Myös pehmeiden taitojen luonne tulee ottaa huomioon sovellettaessa. Tutkimuksessa ei ote kuitenkaan kantaa siihen miten näiden taitojen tulee esiintyä vaan ne ovat jokaiselle yksilöllisiä.

5.5 Suositukset tulevalle tutkimukselle

Tutkimuksen tulokset ovat listaus pehmeistä taidoista, joita työskentelyyn DevOps-organisaatiossa tarvitaan. Olemassa olevan tiedon lisäämiseksi tulevassa tutkimuksessa olisi paljon potentiaalia pyrkiä hienovaraisemmin jaottelemaan tätä listausta. Eri pehmeiden taitojen tarpeen jakautuminen roolin tai organisaation muutosprosessin vaiheen mukaan lisäisi tiedon sovellettavuutta. Tämän lisäksi organisaation koolla, toimialalla ja organisaatiomallilla saattaa olla vaikutusta tarvittaviin taitoihin. Tämän lisäksi tutkimus ei ota kantaa siihen millaisia painoarvoja eri taidoille tulisi antaa. Nämä painotukset poikkeavat potentiaalisesti toisistaan, riippuen organisaation koosta, organisaatiomallista tai tuotettavasta lopputuotteesta. On myös mahdollista, että tarvittavien taitojen painotukset ovat riippuvaisia organisaation maantieteellisestä sijainnista. Toinen potentiaalisesti vaikuttava tekijä on etätyön määrä organisaatiossa.

5.6 Arviointi

Tutkimuksessa käytettyjen julkaisujen määrä on pieni, joka rajoittaa tulosten luotettavuutta. Tutkimuksessa on kuitenkin huomioitu DevOps-viitekehyksen jokainen osa-alue puolueettomasti. Tutkimuksessa on myös huomioitu DevOps-ilmiön jatkuva kehitys siten että jokainen siinä käytetty julkaisu on korkeintaan 5 vuotta vanha.

Sekä pehmeät taidot että DevOps ovat vailla yleisesti hyväksyttyä määritelmää. Tästä syystä kumpikin termi on määritelty tässä tutkimuksessa erikseen. Vaikka käytetyt määritelmät ovat varsin geneerisiä, tulee tämä asia huomioida tuloksia tarkasteltaessa. Koska pehmeät taidot ovat myös konteksti sidonnaisia, tulee myös tässä tutkimuksessa määritelty konteksti ottaa huomioon.

Lähteet

Sosiaali- ja terveysministeriö. (2016). Digitalisaatio terveyden ja hyvinvoinnin tukena. Sosiaa li- ja terveysministeriön digitalisaatiolinjaukset 2025. Noudettu 27.1.2021 osoitteesta https://julkaisut.valtioneuvosto.fi/bitstream/handle/10024/75526/JUL2016-5-hallinnonalan-ditalisaation-linjaukset-2025.pdf?sequence=1

Koch, T. & Windsperger, J. (1. Toukokuuta 2017). Seeing through the network: Competitive advantage in the digital economy. J Org Design 6, 6 (2017). DOI: https://doi.org/10.1186/s41469-017-0016-z

DevOps Research & Assestment. (2019). Accelerate: state of DevOps 2019. Noudettu 30.1.2021 osoitteesta https://cloud.google.com/devops/state-of-devops/

Amazon Web Services. (2021). What is DevOps. Noudettu 31.1.2021 osoitteesta https://aws.amazon.com/devops/what-is-devops/

Global Industry Analysts, Inc. (2020). DevOps - Global Market Trajectory & Analytics. Noudettu 31.1.2021 osoitteesta https://www.researchandmarkets.com/reports/4804823/devops-global-market-trajectory-and-analytics

Bishop, C. (2019) The Need For Soft Skills Training Grows As Automation Transforms The Workplace, Forbes, Noudettu 17.4.2021 osoitteesta: https://www.forbes.com/sites/forbeshumanresourcescouncil/2019/04/25/the-need-for-soft-skills-training-grows-as-automation-transforms-the-workplace/

Dean, S.A., East, J.I. (2019) International Journal of Applied Management and Technology 2019, Volume 18, Issue 1, Pages 17–32 DOI:10.5590/IJAMT.2019.18.1.02

Forsgren, N., Humble, J. & Kim, G. (2018) Accelerate - Building and Scaling High Performing Technology Organisations. IT Revolution ISBN: 978-1942788331

DevOps Research & Assestment. (2021) Explore DORA's research program. Noudettu 31.1.2021 osoitteesta https://www.devops-research.com/research.html

Ravichandran, A., Taylor, K. & Waterhouse, P. (2016) DevOps Foundations. In: DevOps for Digital Leaders. Apress, Berkeley, CA. DOI: https://doi.org/10.1007/978-1-4842-1842-6_3

Sharma, S. (2017) The DevOps Adoption Playbook: A Guide to Adopting DevOps in a Multi-Speed IT Enterprise, ISBN: 978-1-119-30874-4

Tutkimuseettinen neuvottelukunta (2020) Hyvä tieteellinen käytäntöä (HTK), Noudettu 17.4.2021 osoitteesta: https://tenk.fi/fi/tiedevilppi/hyva-tieteellinen-kaytanto-htk

Figalist, I., Biesdorf, A., Brand, C., Feld, .S & Kiermeier, M., (2019) Supporting the DevOps Feedback Loop using Unsupervised Machine Learning, 2019 IEEE International Symposium on INnovations in Intelligent SysTems and Applications (INISTA), Sofia, Bulgaria, 2019, pp. 1-6, DOI: 10.1109/INISTA.2019.8778283.

Okoli, C. & Schabram, K. (2010). A Guide to Conducting a Systematic Literature Review of Information Systems Research. Sprouts: Working Papers on Information Systems, 10(26) Noudettu 7.2.2021 osoitteesta https://www.academia.edu/download/3250666/OkoliSchabram2010SproutsLitReviewGuide.pdf

Heckman, J. & Kautz, T. (2012). Hard evidence on soft skills. Labour Economics, Volume 19, Issue 4 2012, S.451-464, ISSN 0927-5371, DOI: https://doi.org/10.1016/j.labeco.2012.05.014

Jabbari, .R, Ali .N & Petersen, .K, Tanveer.B (2016). What is DevOps? A Systematic Mapping Study on Definitions and Practices. The Scientific Workshop XP2016, DOI: https://doi.org/10.1145/2962695.2962707

Davis, J. & Daniels, R. (2019) Collaborating in DevOps Culture. GitHub. Noudettu 7.2.2021 osoitteesta: https://resources.github.com/downloads/CollaboratingDevOpsCulture.pdf

Dyck, .A, Penners, .R & Lichter, .H (2015). Towards Definitions for Release Engineering and DevOps. Release Engineering (RELENG), 2015 IEEE/ACM 3rd International Workshop on, vol., no., pp.3,3, 19-19 May 2015, DOI: https://doi.org/10.1109/RELENG.2015.10

Davis, .J & Daniels, .R. (2016) Effective DevOps. O'Reilly Media, ISBN: 9781491926307

Lippman, .L, Ryberg, .R, Carney, .R & Moore, .A (2015). Workfoce connections: Key "soft skills" that foster youth workforce success: towards consensus across field. Child trends. Child Trends Publication 2015‐24, Noudettu 21.2.2021 osoitteesta: http://www.childtrends.org/wp-content/uploads/2015/06/2015-24WFCSoftSkills1.pdf

Schulz, .B. (2008). The importance of soft skills: Education beyond academic knowledge. Nawa Journal of Communication, 2(1), 146-154. Noudettu 21.2.2021 osoitteesta: http://ir.nust.na/handle/10628/39

Cimatti, .B. (2016). Definition, development, assessment of soft skills and their role for the quality of organizations and enterprises. International Journal for Quality Research. 10. 97-130. DOI: http://dx.doi.org/10.18421/IJQR10.01-05

Bancino, .R & Zevalkink, .C. (2007). Soft Skills: The New Curriculum for Hard-Core Technical Professionals. Techniques: Connecting Education and Careers. 82. Noudettu 21.2.2021 osoitteesta: https://www.researchgate.net/publication/234603133_Soft_Skills_The_New_Curriculum_for_Hard-Core_Technical_Professionals

Ahmed, F., Capretz, L. F., Salah, .B & Campbell, .P. (2013). Soft Skills and Software Development: A Reflection from the Software Industry. International Journal of Information Processing and Management, 4(3):171-191, 2013. DOI: https://arxiv.org/abs/1507.06873

Ahmed, F., Capretz, L. F. & Campbell, .P. (2012) Evaluating the Demand for Soft Skills in Software Development, IT Professional, vol. 14, no. 1, 44-49, 2012, DOI: https://doi.org/10.1109/MITP.2012.7

Salminen, A. (2011). Mikä kirjallisuuskatsaus?: Johdatus kirjallisuuskatsauksen tyyppeihin ja hallintotieteellisiin sovelluksiin. Vaasan Yliopisto. Opetusjulkaisuja 62 Julkisjohtaminen 4. Noudettu 7.3.2021 osoitteesta: https://osuva.uwasa.fi/bitstream/handle/10024/7961/isbn_978-952-476-349-3.pdf?sequence=1

Schryen, G. (2015) "Writing Qualitative IS Literature Reviews—Guidelines for Synthesis, Interpretation, and Guidance of Research," Communications of the Association for Information Systems: Vol. 37, Article 12. DOI: https://doi.org/10.17705/1CAIS.03712

Babb, J., Nørbjerg, J., Yates, D.J .& Waguespack, L.J., (2017) The Empire Strikes Back: The end of Agile as we know it? .Selected Papers of the IRIS, Issue Nr 8 (2017) Noudettu 22.2.2021 osoitteesta: https://aisel.aisnet.org/iris2017/8

Florea, R. & Stray, V. (2020) A Qualitative Study of the Background, Skill Acquisition, and Learning Preferences of Software Testers. In Proceedings of the Evaluation and Assessment in Software Engineering (EASE '20). Association for Computing Machinery DOI:https://doi.org/10.1145/3383219.3383252

Forbrig, P. (2018). BizDevOps and the Role of S-BPMProceedings of the 10th International Conference on Subject-Oriented Business Process Management. Association for Computing Machinery, New York, NY, USA, Article 1, 1–8. DOI:https://doi.org/10.1145/3178248.3178250

Ghantous, G.B. & Gill, A. (2017) DevOps: Concepts, Practices, Tools, Benefits and Challenges. PACIS 2017 Proceedings. 96. Noudettu 22.2.2021 osoitteesta: http://aisel.aisnet.org/pacis2017/96

Kuusinen, K. & Albertsen, S. (2019) Industry-academy collaboration in teaching DevOps and continuous delivery to software engineering students: towards improved industrial relevance in higher education. Proceedings of the 41st International Confe-rence on Software Engineering: Software Engineering Education and Training IEEE Press, 23–27. DOI:https://doi.org/10.1109/ICSE-SEET.2019.00011

Gall, M., Pigni, F. (2018) Leveraging DevOps for mission critical software. Noudettu 22.2.2021 osoitteesta: https://aisel.aisnet.org/amcis2018/ITProjMgmt/Presentations/2/

Masombuka, .T & Mnkandla, E. (2018) A DevOps collaboration culture acceptance model. In Proceedings of the Annual Conference of the South African Institute of Computer Scientists and Information Technologists (SAICSIT '18). Association for Computing Machinery, New York, NY, USA, 279–285. DOI:https://doi.org/10.1145/3278681.3278714

Pang, C., Hindle, A.,& Barbosa, D. (2020) Understanding devops education with grounded theory. In Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET '20). Asso-ciation for Computing Machinery, New York, NY, USA, 107–118. DOI:https://doi.org/10.1145/3377814.3381711

Saltz, J., Sutherland , A. (2020) SKI: A New Agile Framework that supports DevOps, Continuous Delivery, and Lean Hypothesis Testing, Proceedings of the 53rd Hawaii International Conference on System Sciences. Noudettu 22.2.2021 osoitteesta: https://aisel.aisnet.org/hicss-53/st/agile_development/4/

Shropshire, J. & Sweeney, B. (2017) On Devops and Workforce Morale. SAIS 2017 Proceedings. 12. Noudettu 22.2.2021 osoitteesta: https://aisel.aisnet.org/sais2017/12

Silva, M., Faustino, J., Pereira, R., & Mira Da Silva, M. (2018). Productivity Gains of DevOps Adoption in an IT Team: A Case Study., Designing Digitalization (ISD2018 Pro-ceedings) Noudettu 22.2.2021 osoitteesta: http://aisel.aisnet.org/isd2014/proceedings2018/ISDevelopment/8

Topi, H. ja Spurrier, G. (2019) Invited Paper: A Generalized, Enterprise-Level Systems Development Process Framework for Systems Analysis and Design Education, Journal of Information Systems Education: Vol. 30 : Iss. 4 , 253-265. Noudettu 22.2.2021 osoitteesta: https://aisel.aisnet.org/jise/vol30/iss4/6

Stray, V., Moe, N. B., Aasheim, A. (2019). Dependency Management in Large-Scale Agile: A Case Study of DevOps Teams. Proceedings of the 52nd Hawaii International Con-ference on System Sciences. Noudettu 22.2.2021 osoitteesta: https://aisel.aisnet.org/hicss-52/st/agile_development/8/

van Belzen, M., Trienekens, J. & Kusters, R. (2019). Critical Success Factors of Conti-nuous Practices in a DevOps Context. In A. Siarheyeva, C. Barry, M. Lang, H. Linger, & C. Schneider (Eds.), Information Systems Development: Information Systems Beyond 2020 (ISD2019 Proceedings). Toulon, France: ISEN Yncréa Méditerranée. Noudettu 22.2.2021 osoitteesta: https://aisel.aisnet.org/isd2014/proceedings2019/ISDMethodologies/6/

Wiedemann , A., Wiesche, M. & Krcmar, H. (2019) Integrating Develoment and Operations in Cross-Functional Teams - Toward a DevOps Competency Model. Proceedings of the 2019 on Computers and People Research Conference. Association for Computing Machinery, New York, NY, USA, 14–19. DOI:https://doi.org/10.1145/3322385.3322400

Wiedemann, A. & Wiesche, M. (2018) "ARE YOU READY FOR DEVOPS? REQUIRED SKILL SET FOR DEVOPS TEAMS". Research Papers. 123. Noudettu 22.2.2021 osoitteesta: https://aisel.aisnet.org/ecis2018_rp/123

Wiedemann, A. (2017) A New Form of Collaboration in IT Teams - Exploring the DevOps Phenomenon. PACIS 2017 Proceedings. 82. Noudettu 22.2.2021 osoitteesta: https://aisel.aisnet.org/pacis2017/82

Wiedemann, A. (2018). IT Governance Mechanisms for DevOps Teams - How Incumbent Companies Achieve Competitive Advantages. Noudettu 22.2.2021 osoitteesta: https://aisel.aisnet.org/hicss-51/os/it_governance/7/

Wiedemann, M. Wiesche, H. Gewald, H. Krcmar. (2019) Implementing the Planning Process within DevOps Teams to Achieve Continuous Innovation. Noudettu 22.2.2021 osoitteesta: https://aisel.aisnet.org/hicss-52/st/agile_development/9/

Tuomi, J. & Sarajärvi, A. (2002). Laadullinen tutkimus ja sisällönanalyysi. Helsinki: Tammi.