Ilari Aalto & Johannes Kumpukoski

HUKKA OHJELMISTOJEN TUOTEKEHITYKSESSÄ

JYVÄSKYLÄN YLIOPISTO
INFORMAATIOTEKNOLOGIAN TIEDEKUNTA
2020

TIIVISTELMÄ

Aalto, Ilari ja Kumpukoski, Johannes
Hukka ohjelmistojen tuotekehityksessä
Jyväskylä: Jyväskylän yliopisto, 2020, 94 s.
Tietojärjestelmätiede, pro-gradu -tutkielma
Ohjaajat: Luoma, Eetu ja Seppänen, Ville

Tutkimuksessa selvitettiin, mitä ohjelmistojen tuotekehityksen hukalla ja sen hallinnalla tarkoitetaan ja miten organisaatiot tunnistavat ja hallitsevat hukkaa. Tutkimuksen lähtökohtana ja tarkoituksena oli selvittää, minkälaista hukkaa eri ohjelmistojen tuotekehityksen tasoilla syntyy ja miten sitä pyritään minimoimaan ja hallitsemaan erilaisilla menetelmillä. Tutkimuksen kohteena oli eri ohjelmistojen tuotekehityksen tasot ja niiden erilaiset hukan tunnistamistavat ja menetelmät hukan hallintaan. Tutkimuskysymys oli: Miten voidaan tunnistaa ja minimoida hukkaa eri ohjelmistojen tuotekehityksen tasoilla? Aineiston keruu suoritettiin puolistrukturoiduilla teemahaastatteluilla. Aineisto analysoitiin teemoittelun avulla. Tutkimusaineisto koostui kolmesta eri ohjelmistojen tuotekehitystä tekevästä yrityksestä. Näistä yrityksistä haastateltiin henkilöitä kolmelta eri ohjelmistojen tuotekehityksen tasolta: 1) Portfoliotaso, 2) tuotetaso ja 3) tuotekehitystaso. Haastatteluiden tulokset osoittavat, että ohjelmistojen tuotekehityksessä syntyy hukkaa kaikilla tasoilla, joita tutkimuksessa tutkittiin ja yritykset pyrkivät aktiivisesti minimoimaan hukan riskiä ja hallitsemaan hukkaa erilaisilla menetelmillä. Toisissa yrityksissä hukan tunnistaminen ja hallitseminen oli suuremmassa roolissa kuin toisessa. Hukka ymmärrettiin lisäarvoa tuottamattomina prosesseina tai tehtävinä ohjelmistojen tuotekehityksen aikana ja niiden poistaminen nähtiin ohjelmistojen tuotekehitystä tehostavana seikkana kaikilla eri ohjelmistojen tuotekehityksen tasolla. Tutkimustulokset osoittavat, että hukan tunnistamista, sen riskin välttämistä ja toteutuneen hukan hallintaa tehdään pääsääntöisesti erilaisten ohjelmistojen tuotekehityksen menetelmien ja prosessien kautta. Yrityksillä ei ollut omia menetelmiä hukan tunnistamiseksi. Tutkimuksen perusteella voidaan päätellä, että ohjelmistojen tuotekehityksen hukka on yrityksille epämieluinen asia ja sitä aktiivisesti pyritään hallitsemaan. Yrityksillä ei ole konkreettisia keinoja tai menetelmiä hukan tunnistamiseen, riskin minimoimiseen tai hallintaan. Johtopäätöksenä esitetään, että ohjelmistojen tuotekehityksen hukan aktiivinen tunnistaminen, sen riskin minimointi ja toteutuneen hukan hallinta tehostavat ohjelmistojen tuotekehitystä ja tekevät eri ohjelmistojen tuotekehityksen prosesseista enemmän arvoa tuottavia ohjelmistolle.

Asiasanat: Ohjelmistojen tuotekehitys, hukka, ohjelmistoyritykset

ABSTRACT

Aalto Ilari and Kumpukoski Johannes
Waste in Software Development
Jyväskylä: University of Jyväskylä, 2020, 94 p.
Information Systems, Master's Thesis
Supervisors: Luoma, Eetu and Seppänen, Ville

The study explored what is waste in software development and how organizations identify and manage waste in software development. The purpose of the research was to find out what kind of wastes are there in software development and how the risk of the waste is minimized and managed in the different levels of software development. The subject of the study was the different levels of software development and the different methods of identifying, minimizing the risk and managing the waste. The research question was: How can an organization identify, minimize the risk and manage waste in software development. The research material for the study was collected by using qualitative research method through semi-structured theme interviews, which were analyzed by theming the results of the interview's answers. The research material consisted three different organizations interviews of the: 1) portfolio level, 2) product level and 3) software development level. The results of the study show, that the waste occur in every level that were studied, and companies try to actively to identify, minimize the risk and manage the waste with different methods. In other companies the identifying and managing the waste was in bigger role than in others. Waste was seen as a process or task that did not produce any value during the software development process and removing waste was seen to improve the effiency on all levels of the software development process. The research results also show that the identifying, minimizing the risk and management of waste is mainly done through different software development methods. The organizations did not have their own methods on waste management in software development. The organizations in the study did not have concrete measures or methods to identify waste, minimize the risk or manage it. Based on the study, it can be concluded that waste in software development is undesirable and the organizations try to manage it actively. The conclusion is that when waste is actively identified, the risk of it is minimized and realized waste is managed, the software development is more efficient and the process of software development produces more value to the software.

Keywords: Software development, waste, software companies

KUVIOT

TAULUKOT

SISÄLLYS

1 JOHDANTO

Ohjelmistojen tuotekehityksessä pyritään tehokkaaseen työskentelyyn. Työskentelyn tehokkuutta pyritään ylläpitämään ja kehittämään erilaisilla johtamismenetelmillä. Johtamismenetelmien avulla pyritään parantamaan ohjelmistojen tuotekehityksen tehokkuutta. Hukka on jokin operaatio tai toiminto ohjelmistojen tuotekehityksessä, joka ei luo arvoa prosessin aikana. (Al-Baik ja Miller, 2014), (Hicks, 2007). Hukka käsitteenä on alunperin teollisuuden ja tarkemmin autoteollisuuden käsite, joka pyrittiin tunnistamaan ja sitä pyrittiin minimoimaan erilaisilla johtamismenetelmillä ja -filosofioilla. Yksi tunnetuimmista hukkaa minimoivista johtamisfilosofioista on Lean-ajattelutapa. Sen tärkein tavoite on tunnistaa hukka, eliminoida se tai minimoida sen riski.

Hukka on alun perin johdettu tuotantoteollisuudesta, ja käännetty ohjelmistojen tuotekehitykseen, sen takia se ei ole täysin linjassa ohjelmistojen tuotekehityksen hukkaan (Al-Baik ja Miller, 2014; Poppendieck ja Poppendieck, 2003; Poppendieck ja Poppendieck, 2006). Ohjelmistojen tuotekehityksen hukkaa on tutkittu vain vähän ja näissäkin tutkimuksissa on käytetty pääasiallisina hukan määritelminä tuotantoteollisuuden hukan määritelmiä teoreettisena pohjana (Al-Baik ja Miller, 2014). Mikäli ohjelmistoja toteuttava yritys pystyy tunnistamaan hukan ja eliminoimaan sen ja sen riskit, on yrityksen ohjelmistojen tuotekehitys tehokkaampaa. Tämän takia hukan tutkiminen puhtaasti ohjelmistojen tuotekehityksen näkökulmasta on tärkeää tieteellisesti sekä käytännöllisesti.

Tutkimuksen tutkimuskysymys on: miten hukkaa voidaan vähentää portfoliotasolla, tuotetasolla ja tuotekehitystasolla. Tutkimuskysymykseen vastaaminen edellyttää, että ymmärretään myös, mitä hukka on ja mistä se syntyy jokaisella tasolla.

Tutkimusmenetelmänä tässä pro gradu -työssä käytetään systemaattista kirjallisuuskatsausta. Systemaattinen kirjallisuuskatsaus tehtiin perustuen Okolin ja Schabramin (2010) systemaattisen kirjallisuuskatsauksen menetelmään. Kirjallisuuskatsauksen tarkoitus on luoda tieteellinen teoreettinen perusta empiiriselle tutkimukselle, joka myöhemmin toteutetaan. Katsauksessa käytiin läpi suuri määrä aiheeseen liittyviä artikkeleita ja teoksia, jotka löydettiin kirjallisuuskatsauksen protokollassa määriteltyjen hakusanojen mukaan.

Löydetyt artikkelit valittiin protokollassa määritellyillä valintaperusteilla ja niistä tehtiin analyysi. Protokollalla tarkoitetaan suunnitelmaa, jonka mukaan systemaattinen kirjallisuuskatsaus tehdään. Protokollan dokumentointi on tärkeää, sen perusteella toteutetaan artikkeleiden haku ja valinta. (Okoli & Schabram, 2010).

Protokolla testattiin ja validoitiin ennen kirjallisuuskatsauksen toteuttamista. Testaamisen yhteydessä havaittavat puutteet korjattiin. Tällaisia puutteita oli esimerkiksi sanan Waste suomenkielinen vastine "Roska", joka aiheutti sen, että löytyvät artikkelit käsittelivät roskien hallintaa ohjelmistoilla. Nämä artikkelit poistettiin läpikäytävistä aineistoista. Protokolla koulutettiin molemmille kirjallisuuskatsauksen tekijöille ja ensimmäinen hakuvaihe toteutettiin tekijöiden toimesta omatoimisesti, jonka jälkeen omista löydöksistä tehtiin valinnat analyysivaiheeseen.

Hakusanat, joilla kirjallisuutta haettiin tietokannoista, olivat waste, software development, software product development, portfolio management, software product, product management, information system development, information system product, identification, information technology, elimination, management, software engineering ja näiden yhdistelmiä.
E-kirjastot, joita käytetään kirjallisuuden etsimiseen ovat: AIS Electronic Library, ACM, IEEE. Kirjastot valittiin niiden kattavuuden ja luotettavuuden perusteella. Kirjastojen valinta arvioitiin vertaisarvioiden ja artikkeleiden saatavuuden perusteella.

Tutkimuksen toisessa osiossa toteutetaan kvalitatiivinen tutkimus. Tutkimusmenetelmäksi valittiin puoliavoin teemahaastattelu sen vuoksi, koska tutkimuksen tarkoitus on ymmärtää tiettyä ilmiötä. Tässä tutkimuksessa ilmiö on hukka ohjelmistojen tuotekehityksen kolmella eri tasolla.
Toteutettava tutkimus on laadullinen tapaustutkimus. Tapaustutkimuksella tarkoitetaan tutkimusta, jossa valitaan yksi tai pieni joukko tapauksia, joihin tutkimus kohdistetaan. Aineistoa voidaan kerätä tapaustutkimuksessa erilaisilla metodeilla kuten havainnoimalla, haastatteluilla tai tutkimalla dokumentteja.
Tämä menetelmä valittiin sen vuoksi, koska tutkimuksen tarkoitus on tutkia ilmiötä oikeassa ympäristössä. Valitulla menetelmällä ymmärretään paremmin kehittävien tuotteiden ja niiden valmistajien välisiä vuorovaikutuksia. Tutkimuksen tapaus on ymmärtää ilmiötä/ ilmiöitä, jotka aiheuttavat hukkaa ohjelmistojen tuotekehityksessä.

Empiirinen materiaali kerätään haastattelemalla eri yrityksistä edellä mainituilla kolmella tasolla työskenteleviä henkilöitä. Haastatteluilla saamme aineistoa jokaiselta tasolta analysoitavaksi ja ymmärtääksemme mitä hukka näillä tasoilla on.

Tutkimus jakautuu kahdeksaan lukuun. Luvussa 2 syvennytään ohjelmistojen tuotekehitykseen tutustumalla mitä sillä tarkoitetaan ja tutkimalla mitä prosesseja tuotekehitys sisältää. Luvussa 3 tutustutaan ohjelmistojen tuotekehityksessä esiintyvään hukkaan ja koetetaan ymmärtää mitä se voi olla. Luvun lopussa hukasta esitetyistä näkemyksistä luodaan taulukko, jota tullaan käyttämään tutkimuksessa selvittäessä mitä hukka eri tasoilla on. Luvussa 4 tutustutaan hukan hallintaan ja vähentämiseen pyrkiviä ohjelmistojen tuotekehitysmenetelmiä. Luvussa 5 kuvataan tutkimusmenetelmä,

tapaustutkimuksen kohteeksi valitut yritykset, tutkimusmalli sekä tiedonkeruu ja analysointi. Luvussa 6 raportoidaan tapaustutkimuksen tulokset pääasiassa tutkimusmallin mukaisella rakenteella. Luvussa 7 esitetään tiivistetysti tutkimuksen tulokset, esitetään niiden pohjalta johtopäätöksiä, verrataan tuloksia aiempiin tutkimuksiin, tarkastellaan tulosten hyödynnettävyyttä sekä esitetään reliabiliteetti- ja validiteettitarkastelu. Tutkimus päättyy johtopäätöksiin ja suositeltuihin jatkotutkimuksiin luvussa 8.

Tutkimuksella on toimeksiantaja, joka on suomalainen keskisuuri ohjelmistojen tuotekehityksen konsultointia tekevä yritys. Yritys oli mukana tutkimuksen suunnittelussa ja haastateltavien yritysten määrittelyssä.

2 OHJELMISTOJEN TUOTEKEHITYS

Tuote on valmis, fyysinen tai aineeton kappale tai palvelu, joka on valmistettu jonkun toimesta. Tuotekehitys tarkoittaa prosessia, jonka aikana ideasta valmistetaan valmis tuote markkinoille myyntiin. Tuotekehitys on ketju, joka muodostuu toisiaan seuraavista vaiheista, joilla kaikilla on tärkeä rooli valmiin tuotteen valmistusprosessissa. Ohjelmisto on kokoelma toimintaohjeita, jotka kertovat laitteelle, kuinka työskennellä. Ohjelmointi tarkoittaa ohjeiden kirjoittamista. Ohjelmistot ja niiden tuotekehitys ei prosessina poikkea teollisesti tuotettavien fyysisten tuotteiden kehityksestä (Krishnan & Ulrich, 2001.).
Ohjelmistotuotteiden kehitystä voi pitää prosessina, jonka aikana idea valmiista ohjelmistosta muutetaan päätösten, teknologian ja ohjelmoinnin avulla myytäväksi ohjelmisto tuotteeksi (Krishnan & Ulrich, 2001). Nambisan & Wilemon (2000) toteavatkin, että tuotteen tuotekehityksen kolme tärkeintä tekijää ovat ihmiset, teknologia ja prosessit.

Ohjelmiston kehitysprosessilla tarkoitetaan rakennetta, jonka mukaisesti ohjelmisto kehitetään (Krishna & Sreekanth, 2016; Vadivu & Tapaskar, 2011). Rakenne on viitekehys, joka jäsentää ohjelmiston kehityksen ja vaiheistaa sen toteutuksen. Prosessin tarkoitus on tehdä ohjelmistojen tuotekehityksestä tehokasta (Krishna & Sreekanth, 2016). Pelkkä hyvä idea myytävästä tuotteesta ei ole tae sen kaupallisesta menestyksestä. Menestymiseen vaaditaan, että tuote on laadukas, se sisältää tarvittavat ominaisuudet ja se valmistetaan aikataulussa sekä kustannustehokkaasti (Wallin, Ekdal & Larsson, 2002).

Ohjelmistokehitysmalleja on useita ja niitä kehitetään jatkuvasti uusia, kaikki mallit sisältävät pääsääntöisesti samat vaiheet ja niiden sisältämät tehtävät. Eri malleissa vaiheiden toteutustavat ja tehtävät poikkeavat toisistaan. Ohjelmistokehitysprosessi muodostuu pääsääntöisesti seuraavista vaiheista ja vaiheiden sisältämistä tehtävistä: vaatimusten analysointi, ominaisuuksien määrittely, ohjelmistoarkkitehtuurin muodostaminen, ohjelmiston toteutus, ohjelmiston testaus, dokumentointi, käyttöönotto ja tuki sekä ylläpito (Krishna & Sreekanth, 2016.).

Vaatimusten analysointi tarkoittaa ohjelmiston tavoitteiden ja toiminnallisuuksien analysoimista. Vaatimuksilla tarkoitetaan sitä, miten valmiin ohjelmiston tulee toimia, vaatimukset myös kootaan yhteen ja dokumentoidaan. Ohjelmiston ominaisuuksien määrittelyllä tarkoitetaan niiden ominaisuuksien kuvaamista, jotka toimivan ohjelmiston pitää sisältää. Ohjelmistoarkkitehtuurin muodostaminen tarkoittaa ohjelmiston sisältävien ominaisuuksien ohjelmistojen ja komponenttien muodostamista. Ohjelmiston toteutus sisältää varsinaisen ohjelmiston ohjelmoinnin. Ohjelmiston testaus tarkoittaa ohjelmoidun koodin testaamista, jotta ohjelma toimii suunnitellusti. Dokumentoinnilla tarkoitetaan ohjelmiston koodin kirjaamista ylös tulevia käyttäjiä ja ylläpitäjiä varten. Käyttöönotto tarkoittaa ohjelmiston julkaisua, tässä vaiheessa ohjelmisto voidaan ottaa käyttöön ja viedä markkinoille myytäväksi. Ylläpito tarkoittaa ohjelmiston jatkuvaa ylläpitoa, se tarkoittaa havaittujen virheiden korjaamista ja uusien ominaisuuksien kehittämistä (Krishna & Sreekanth, 2016.).

Ohjelmiston kehitysprosessi on kuitenkin vain yksi vaihe ohjelmistojen tuotekehityksessä. Kehitysprosessin lisäksi uuden tuotteen kehitys ideasta tuotteeksi sisältää tuotteen ideointia, konseptointia, arviointia ja kaupallistamista (Nambisan & Wilemon, 2000). Ohjelmoinnin lisäksi tuotekehitys edellyttää oikeiden liiketoiminnallisten päätösten tekemistä. Oikeat päätökset perustuvat faktoihin ja huolelliseen tuotteen liiketoiminta-alueeseen vaikuttavien tekijöiden arviointiin. Näitä ovat mm. markkinat, kilpailijat, tekninen toteutettavuus, strategia, tuotteen laatu ja yrityksen omat resurssit (Wallin ym. 2002).

Menestyneen ohjelmiston valmistus edellyttää sekä ohjelmiston tuotekehitysprosessien käyttämistä, että oikeiden liiketoiminnallisten päätösten tekemistä. Tuotekehitysprosessi mallien ja liiketoiminnallisten päätösten teko mallit vaikuttavat ohjelmisto tuotteen kehitykseen eri tavoilla. Ohjelmiston valmistukseen ideasta tuotteeksi on kehitetty koko putken kattavia malleja, näistä yksi on niin kutsuttu Stage-Gate prosessi. Malli sisältää määrän eri kehitysvaiheita ja jokaisen vaiheen päättää tuotteen kehitykseen liittyvien sidosryhmien päätös tuotteen kehityksen jatkosta. Monet yritykset soveltavat ja käyttävät mm. Stage-Gate prosessin sisältämiä vaiheita omissa tuotekehitysprojekteissa omalla tavallaan.

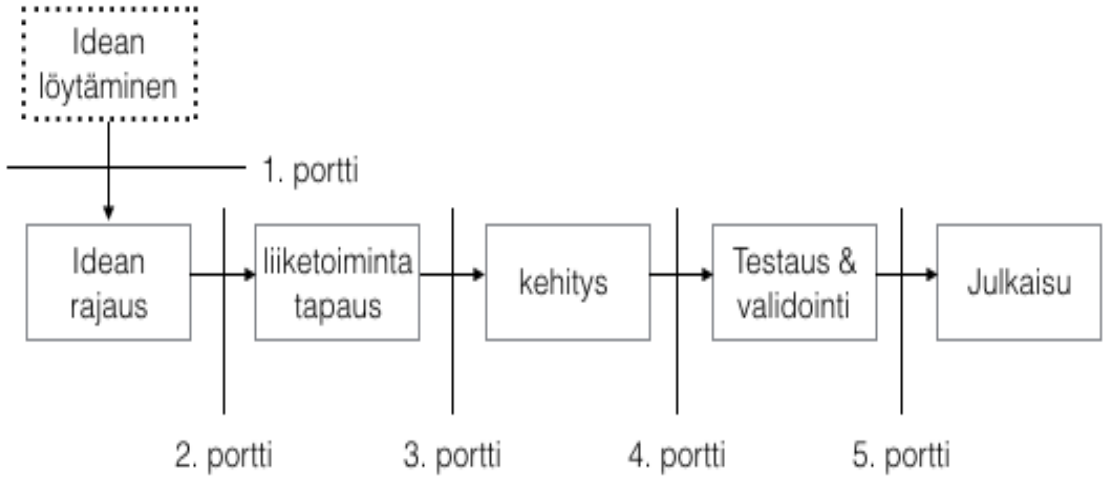
Cooperin Stage-Gate prosessi jakaa tuotekehityksen kuuteen eri vaiheeseen ja viiteen päätöksentekoon, joita malli kutsuu porteiksi. Prosessin sisältämät vaiheet ovat: idea, rajaus, liiketoiminta tapaus, kehitys, verifiointi ja validointi sekä julkaisu. Aina ennen siirtymistä seuraavaan vaiheeseen tehdään päätös, onko tuotekehitystä kannattava jatkaa. Jokainen vaihe sisältää päällekkäisiä eri tiimien toteuttamia aktiviteetteja. Jokaisen vaiheen tarkoitus on kerätä tietoa vaiheen päättävän päätöksenteon tueksi tuotekehityksen jatkosta. Prosessin tarkoitus on tehostaa tuotekehitystä ja vähentää siihen liittyviä riskejä (Wallin ym. 2002.).

Ensimmäisessä vaiheessa prosessia idea uudesta tuotteesta tai paremmasta versiosta tuodaan julki. Tuotepäällikkö tai vastaava kerää tarvittavan informaation ensimmäistä päätöksentekoa varten. Ensimmäistä vaihetta seuraa ensimmäinen päätöksenteko, jossa tehdään päätös, uhrataanko ideaan enemmän resursseja ja aloitetaanko idean perusteella uusi tuotekehitysprojekti. Tilanteessa tuotepäällikkö tai vastaava esittelee idean muille projektiin kuuluville sidosryhmille kuten tuotekehityksestä vastaavalle, markkinoinnille, myynnille ja ylläpidolle ketkä yhdessä tekevät päätöksen tuotekehityksen jatkosta (Wallin ym. 2002.).

Toisessa vaiheessa prosessia pyritään ymmärtämään tuotteen markkinat ja teknologia sekä rajaamaan tarvittavat ominaisuudet seuraavaa päätöksentekoa varten. Toinen päätöksentekovaihe noudattaa samaa ideaa kuin ensimmäisen mutta tässä vaiheessa päätöksenteon tukena, on oltava huomattavasti enemmän informaatiota, jota on kerätty edeltävässä prosessissa (Wallin ym. 2002.).
Kolmannessa vaiheessa prosessia muodostetaan uudesta tuotteesta liiketoiminta tapaus. Tämä sisältää tarkan selvityksen joka kuvaa tuotetta, markkinoita, miten tuote sijoittuu yrityksen portfolioon, tuotekehitysprojektia, kilpailijoita jne. Selvityksen tarkoitus on kartoittaa, onko tuote toteutettavissa. Kolmannessa päätöksenteko vaiheessa tehdään päätös, aloitetaanko tuotteen kehitys. Tässä vaiheessa päätetään investoiko yritys tuotteen kehitykseen vai ei. Päätös tehdään sekä taloudellisesta näkökulmasta, että liiketoiminta tapauksen analyysin perusteella (Wallin ym. 2002.).

Neljännessä vaiheessa prosessia kehitetään itse tuote. Kehitys tapahtuu yrityksen käyttämän mallin mukaisesti ja se sisältää pääsääntöisesti tässä luvussa aikaisemmin kuvatut tuotekehityksen vaiheet. Tämän vaiheen tuotteena pitäisi syntyä tuote, joka on valmis testattavaksi markkinoilla. Neljännessä päätöksenteko vaiheessa päätetään, aloitetaanko tuotteen testaus käyttäjillä. Päätös tehdään arvioinnin perusteella tuotteesta ja sen markkinoista sekä miten se sopii yrityksen portfolioon. Päätöksen perusteella yritys joko aloittaa tuotteen testaamisen, kehittää siihen uusia ominaisuuksia tai hylkää koko tuotteen (Wallin ym. 2002.).

Viidennessä vaiheessa tuotteen kiinnostavuus ja toimivuus varmistetaan testaamalla sitä valitulla joukolla käyttäjiä. Testauksen perusteella tehdään päätös tuotteen julkaisusta. Tässä vaiheessa yritys voi yhä hylätä tuotteen julkaisun. Mikäli tuote päätetään julkaista, käynnistää se samalla myös muut operaatiot kuten tuotteen markkinoinnin ja myynnin. Tällöin tuote myös otetaan yrityksen tuoteportfolioon ja alkaa sen elinkaaren hallinta ja jatkokehitys. Viimeinen vaihe on tuotteen julkaisu, joka sisältää julkaisuun ja muut aktiviteetit yrityksessä tuotteen myymiseksi ja hallitsemiseksi (Wallin ym. 2002.).



KUVIO 1 Stage-gate prosessi

2.1 Ohjelmistojen tuotekehityksen tasot

Ohjelmistojen tuotekehitys voidaan jakaa tehtävien luonteen mukaan kolmeen eri tasoon, portfolio, tuote- ja ohjelmointi. Näistä jokainen sisältää yrityksen ja yksittäisen ohjelmiston menestymiseksi tärkeitä prosesseja. Portfoliotaso tarkoittaa yrityksen nykyisten ja tulevien tuotekehitysprojektien ja resurssien hallintaa. (Vähäniitty & Rautiainen, 2005.) Tuotetaso tarkoittaa yrityksen yksittäiseen ohjelmiston ja sen elinkaareen liittyvien päätösten hallintaa. (Weerd,

Brinkkemper, Nieuwenhuis, Versendaal & Bijlsma, 2006). Tuotekehitystaso tarkoittaa niitä tehtäviä, jotka yhdessä tuottavat yksittäisen asiakkaan käytettäväksi tulevan ohjelmiston tai ominaisuuden. (Weerd ym. 2006).

2.1.1 Portfoliotaso ohjelmistojen tuotekehityksessä

Menestyminen tuotekehityksessä edellyttää yritykseltä kykyä viedä yksittäiset kehitysprojektit onnistuneesti läpi. Lisäksi menestyäkseen yrityksen on osattava hahmottaa mitä yrityksen kannattaa tehdä nyt, mitä lykätä tulevaisuuteen ja mitä jättää tekemättä. Nykyisen ja tulevan tuotekehityksen jatkuva suunnittelu- ja arviointiprosessi on tärkeää yrityksen kilpailukyvyn ylläpitämiseksi. Portfolion käyttö auttaa yritystä resursoimaan resurssit oikealla hetkellä oikeisiin asioihin, jotta yritys menestyy pitkässä juoksussa. Portfolio eli tuotekehitys salkku kertoo yrityksen nykyiset ja tulevaisuuden kehityshankkeet (Vähäniitty & Rautiainen, 2005.).

Portfoliolla tarkoitetaan yrityksen kaikkia kehitettäviä tuotteita eli ns. tuoteportfoliota (Weerd ym. 2006). Ohjelmistojen tuotekehityksessä portfolio on työkalu, jonka avulla yritys allokoi omia resursseja tuottavuuden kasvattamiseksi, strategian saavuttamiseksi ja riskin hallintaan (Rautiainen, Schantz & Vähäniitty, 2011).

Yrityksen tuotekehitys portfolio sisältää yrityksen meneillään olevat ja tulevat kehitystyöt. Ideoita portfolioon yrityksillä on usein enemmän kuin resursseja näiden toteuttamiseksi. Ideoita voi syntyä niin yrityksen sisältä kuin ulkopuolelta. Muuttuva ympäristö voi myös aiheuttaa yllättäviä tarpeita yrityksen tuotekehitykseen. Yrityksillä pitääkin olla keinot hallita miten uudet tuotteet valitaan portfolioon ja miten ne etenevät ideointivaiheesta kehitysvaiheeseen. Jatkuvaa ideoiden virtaa täytyy osata hallita, jotta yritys osaa jo aikaisessa vaiheessa valita oikeat kehityskohteet yrityksen tuotekehitys salkkuun ja tietää mihin kohdistaa resurssit ja milloin (Vähäniitty & Rautiainen, 2005.).

Portfoliota toteutetaan jatkuvalla portfolion hallinnalla. Tällä tarkoitetaan, että yrityksen on jatkuvasti arvioitava uusien tuotteiden mahdollisuuksia ja niiden edellyttämiä resursseja sekä tehdä päätöksiä näihin liittyen. Arvioinnin seurauksena yritys voi päättää toteuttaa tietyn tuotteen, hylätä koko idean tai lykätä tuotteen toteutusta. Lisäksi arviointi helpottaa hahmottamaan, koska tuote kannattaa tehdä ja miten nopeasti eli kuinka paljon siihen kannattaa resursoida resursseja. Päätösten tekemiseen vaikuttaa yrityksen valitsema strategia seuraaville vuosille, jonka lisäksi päätöksiin voi vaikuttaa yrityksen ulkoiset tekijät kuten muuttunut lainsäädäntö tai kilpailutilanne. Portfolion hallinta on uusien tuotteiden jatkuvaa arvioimista ja tuotekehityksen työn priorisointia ja hahmottamista (Vähäniitty & Rautiainen, 2005.).

Portfolion hallinnalla tarkoitetaan yrityksen tuoteportfolioon liittyvien päätösten tekemistä. Portfolion hallintaan kuuluu päätökset, jotka koskevat yrityksen olemassa olevia tuotteita, niiden elinkaaren hallinta. Portfolion hallintaa on myös mahdollisten uusien tuotteiden arviointi ja päätökset

tuotteiden kehittämisestä tai hylkäämisestä. Lisäksi portfolion hallintaan kuuluu myös ulkopuolisten kumppaneiden ja ulkoistetun kehityksen hallinta (Weerd ym. 2006.).

Rautiainen ym. (2011) toteavat, että yleisin tulkinta portfolion hallinnasta ohjelmistojen tuotekehityksessä on, että se on resursseihin perustuvia päätöksiä yrityksen portfolion sisältämien suunniteltujen ja jo kehityksessä olevien tuotteiden kesken. Leffingwell (2011) kuvaa portfolion hallintaa, että se on johdon määrittämiä yrityksen investointeja eri teemoihin, jotka määrittävät resurssien allokoinnin ja yrityksen tuotekehityksen suunnan. Poppendieck & Poppendieck (2003) kuvaavat portfolion hallintaa niin, että vaadittava tuotekehitys on luokiteltava tyypeittäin, kuten strategiset hankkeet, tuotteiden päivitykset ja ylläpito ja jokaiselle tyypille määritetään tietty kehityssykli. Tämän jälkeen päätetään, miten paljon resursseja yhteen tyyppiin uhrataan syklin aikana esimerkiksi yhden vuoden aikana. Vaihtoehtoisesti esimerkiksi tuotteen ylläpitoon voidaan määrittää tietty määrä resursseja kokonaisuudesta. Lopuksi resurssit kuvataan esim. kalenterin avulla etukäteen koko syklille. Aina, kun lähestytään tietyn tyypin kehitys vuoroa, suunnitellaan sillä hetkellä mitä uutta strategista hanketta kehittää, mitä olemassa olevaa tuotetta päivittää tai mitä korjauksia olemassa olevaan tuotteeseen kannattaa tehdä juuri sillä hetkellä.
Shallowayn (2010) konsepti ns. ketterästä portfolion hallinnasta tarkoittaa sitä, että resursseja hallitaan jatkuvasta niin, että ne jaetaan eri tuotekehitys projektien kesken. Tarkoituksena on, että markkinoille tuotaisiin mahdollisimman nopeasti uusia vähimmäisvaatimukset sisältäviä tuotteita ja ominaisuuksia, jotta tuotteista ja ominaisuuksista saataisiin nopeasti liiketoiminnallista arvoa. Pichler (2010) täydentää konseptia niin, että ohjelmistoyritysten tuotekehitys jonoa pitäisi käsitellä samalla mallilla. Larman ja Vodde (2010) ovat samaa mieltä ja väittävät, että portfolion hallinta on tehokkaampaa, jos eri tuotteiden kehitys jonot yhdistetään yhdeksi ja tätä hallitaan niin, että markkinoille saadaan mahdollisimman nopeasti vähimmäisvaatimukset täyttäviä tuotteita ja ominaisuuksia.

Onnistunut portfolion hallinta on tasapainon saavuttamista yrityksen eri tavoitteiden kanssa. Portfolion tarkoitus on oikeilla liiketoiminnallisilla päätöksillä kasvattaa yrityksen tulosta. Toiseksi portfolion tulee kuvastaa yrityksen strategiaa ja siihen pääsemistä. Kolmanneksi portfolion tarkoitus on varmistaa, että yrityksen aktiviteetit ovat resursseihin nähden tasapainossa. Portfolion hallintaan on useita eri tekniikoita. Portfolion hallintaa voidaan tehdä enemmän liiketoiminnallisesta näkökulmasta panostamalla laskennallisesta korkeimman tuottopotentiaalin omaaviin tuotteisiin. Vaihtoehtoisesti portfolion hallintaa voidaan tehdä pisteytystekniikoilla sijoittamalla resursseja tasaisemmin eri tuotteisiin ja niiden kehitykseen. Hallintaan vaikuttaa yrityksen strategia ja miten yritys itse uskoo strategiaan pääsemiseksi (Rautiainen ym. 2011.).

Portfolion hallinta on tuotesalkun tuotteiden ja niiden kehityksen arviointia, valintaa ja priorisointia, jota tehdään tarkastelemalla tuotteiden kehitysprosessia jatkuvasti. Käytännössä portfolion hallintaa tapahtuu jatkuvasti tuotekehityksen eri vaiheissa koko tuotteen elinkaaren ajan (Rautiainen ym. 2011.). Portfolion käyttöönottamiseksi Rautiainen ym. (2011) esittelevät kaksi vaihtoehtoista tapaa. Ensimmäinen vaihtoehto korostaa päätöksen tekemistä

jokaisen yksittäisen tuotteen kehityksen arvioinnin perusteella. Tämänkaltainen portfolion hallinta tapahtuu ns. alhaalta ylöspäin, kun päätökset tehdään yksittäisen tuotteen perusteella. Toisen tavan mukaan portfoliota tarkastellaan kokonaisuutena ja päätökset tehdään kaikkien portfolion tuotteiden perusteella. Tämänkaltainen portfolion hallinta tapahtuu ns. ylhäältä alaspäin, kun tarkastellaan kokonaisuutta yksittäisten tuotteiden sijaan.

2.1.2 Tuotetaso ohjelmistojen tuotekehityksessä

Ohjelmistomarkkinat ovat muuttuneet räätälöityjen ohjelmistojen kehittämisestä ohjelmistotuotteiden kehittämiseen. Tämä muutos on aikaansaanut uuden tehtävän ohjelmistoyrityksissä, tuotehallinnan (Weerd ym. 2006). Ohjelmiston tuotehallinta on liiketoiminnallinen prosessi, joka ohjaa tuotetta koko tuotteen elinkaaren ajan saavuttaakseen tuotteelle suurimman mahdollisen liiketoiminnallisen arvon. Määritelmä sisältää kaikki tärkeät aktiviteetit, jotka liittyvät ohjelmisto tuotteisiin. Ohjelmiston menestys riippuu kaikista tuotehallinnan aktiviteeteista aina tuotteen strategian ja markkinoinnin suunnittelusta, ohjelmiston julkaisusta ja tuen tarjoamisesta sen jatkokehitykseen (Maglyas, Nikula & Smolander, 2011.).

Ohjelmistotuotteet edellyttävät erityisestä tuotehallintaa, koska niiden muokattavuus on helpompaa ja niiden päivittäminen tapahtuu nopeammin. Tuotehallinnasta vastaa usein tuotepäällikkö kenen tehtävänä on ymmärtää tuotteen sidosryhmiä, selvittää ja hallita tuotteen vaatimuksia, määritellä yrityksen tuoteportfolio ja suunnitella portfolion tuotteiden julkaisujen sisältö ja aikataulu (Bjarnason, Wnuk & Regnell, 2010.). Tuotepäälliköllä on lukuisia tuotehallinnan tehtäviä, kuten vaatimusten määrittely, julkaisujen sisältöjen suunnittelua ja uusien tuotteiden ja ominaisuuksien kehittämistä. Näistä tehtävistä tekee haasteellisempaa se, että tuotehallinnasta vastaavan täytyy ottaa kaikki sisäiset ja ulkoiset sidosryhmät huomioon (Weerd ym. 2006).

Tuotehallinta sisältää erilaisia tuotteen elinkaaren hallinnan tehtäviä, jotka Weerd ym. (2006) jakaa hierarkian mukaan neljään eri prosessiin. Portfolion hallinta käsittelee tuotteita yrityksen koko tuote portfoliossa. Tuotteen "roadmapping" eli millä aikataululla tuotteita kehitetään ja milloin uusia ominaisuuksia julkaistaan. Tuotteen julkaisun suunnittelu tarkoittaa mitä tuotteen vaatimuksia tuleviin julkaisuihin toteutetaan ja sisällytetään. Vaatimusten hallinta tarkoittaa tuotteen jokaisen yksittäisen vaatimuksen toiminta tarkoitusta ja miten ja miksi se on tärkeä.

Tuotteen roadmapping termillä tarkoitetaan ohjelmistokehityksessä tuotteen suunnittelua. Tarkemmin tuotehallinnassa, sillä tarkoitetaan tuotteen elinkaaren suunnittelua. Tuotteen suunnittelulla voidaan tarkoittaa tuotteen toteutustapaa, resurssien suunnittelua, milloin mitkäkin ominaisuudet toteutetaan yms. (Weerd ym. 2006.).

Vaatimusten hallinta tarkoittaa aktiviteetteja, joiden avulla kootaan, identifioidaan ja tarkistetaan tulevat vaatimukset. Vaatimusten hallinta tarkoittaa myös vaatimusten suunnittelemista niin, että huomioidaan

vaatimusten riippuvuudet toisista vaatimuksista, ydin ominaisuudet, tuotekehitysmenetelmät ja vaatimusten teemat. Vaatimuksia kerätään tuotetta käyttäviltä asiakkailta, myynnistä ja markkinoinnista, kehittäjiltä, asiakaspalvelusta ja yrityksen johdolta (Weerd ym. 2006.).
Vaatimusten hallinta sisältää seuraavia teknisiä ja ei niin teknisiä aktiviteetteja: vaatimusten kuvaaminen, vaatimusten mallintaminen ja analysointi, vaatimuksista viestiminen, vaatimuksista sopiminen ja vaatimusten kehittäminen (Weerd ym. 2006).

Vaatimusten suunnittelu alkaa kaikkien vaatimusten keräämisellä niin yrityksen sisältä kuin ulkopuolisilta sidosryhmiltä. Kerätyt vaatimukset järjestetään tuotteen vaatimuksiksi. Tuote vaatimukset identifioidaan muuttamalla nämä ymmärrettävään muotoon esim. käyttötapausten avulla, samalla poistetaan samaa tarkoittavat vaatimukset. Tämän jälkeen vaatimukset järjestetään tuotteittain sekä mihin ydin ominaisuuteen vaatimus kuuluu (Weerd ym. 2006.).

Vaatimusten suunnittelussa erotetaan toisistaan vaatimukset, joita tarvitaan nopeammin, vaatimukset, joilla on suurempi liiketoiminnallinen arvo ja sellaiset, joita tarvitaan mahdollisesti tulevaisuudessa mutta ei tällä hetkellä. Lisäksi suunnittelussa erotetaan toisistaan asiakkaiden toivomat vaatimukset, liiketoiminnan kehittämisen kannalta tarvittavat vaatimukset ja näiden liiketoiminnallista arvoa arvioidaan vaatimuksen suunnittelussa (Weerd ym. 2006.).

Toimivan ja menestyvän ohjelmiston julkaisu ympäristössä missä markkinat ja ohjelmiston käyttäjät määrittävät tarpeet on äärimmäisen haastavaa, jotta pystytään saavuttamaan asiakkaan vaatimukset ohjelmistolle. Tämän vuoksi julkaisun suunnittelu ja aikatauluttaminen on tärkeää ja joissain tilanteissa tärkeämpää kuin ohjelmiston sisältämät ominaisuudet. Vaatimusten suunnittelun ja julkaisun suunnittelun välissä tapahtuu ohjelmistojen tuotekehityksen prosessit ja tehtävät. Ohjelmistojen tuotekehityksen prosesseista ja tehtävistä kerrotaan luvussa 2.1.3.

Julkaisun suunnittelu on prosessi, jossa kohtaavat aikataulullisesti vaatimusten teknisen toteutuksen suunnittelu ja toteutus sekä markkinoiden tarpeet. Ohjelmiston julkaisulla tarkoitetaan prosessia, kun ohjelmistoon kehitetyt ominaisuudet tuodaan käyttäjille saataville. Julkaisun hallinnan tärkeimmät tehtävät ovat vaatimusten priorisointi eli mitä vaatimuksia seuraavassa julkaisussa otetaan käyttöön, julkaisun suunnittelu eli koska julkaisu tehdään, jotta tiedetään, miten pitkään vaatimuksia voidaan kehittää, julkaisun sisältämien vaatimusten dokumentointi, julkaisun sisältämien ominaisuuksien tiedottaminen ja julkaisun laajuuden hallinta eli minkä verran vaatimuksia julkaisuun on mahdollista sisällyttää (Weerd ym. 2006.).

Markkinoiden tarpeiden ja niiden yhteensovittaminen yrityksen resursseihin on haastavaa, tämän vuoksi menestyvän tuotteen suunnittelu tarvitsee onnistunutta tuotteen hallintaa (Weerd ym. 2006.).

2.1.3 Tuotekehitystaso

Ohjelmiston ohjelmointi sisältää joukon eri työtehtäviä, jotka kaikki yhdessä hallitussa kokonaisuudessa tuottavat valmiin ohjelmiston (Weerd ym. 2006). Lui & Chanin (2006) mukaan ohjelmointi sisältää seuraavat eri tehtävät: vaatimusten ymmärtäminen, suunnittelu, ohjelmointi, testaus ja integrointi. Ohjelmoinnin työtehtäviä voi pitää kognitiivisina tehtävinä, jotka vaativat sekä opettelua, että ymmärtämistä. Ohjelmointi vaatii erilaisia taitoja kuten ongelmanratkaisukykyä, suunnitelmallisuutta, nopeaa ajattelua ja kausaalista päättelyä.

Vaatimusten ymmärtäminen tarkoittaa ohjelmistoon toteutettavien ominaisuuksien ymmärtämistä. Ohjelmoijan pitää ymmärtää miten asiakas tahtoo ohjelmiston toimivan, jotta se voidaan toteuttaa oikein. Ohjelmoijan pitää myös ohjelmoida ohjelmisto niin, että myös asiakas ymmärtää miten ohjelmisto toimii ja osaa käyttää sitä, eli että ohjelmisto on käytettävä. Vaatimusten ymmärtäminen on haastavaa, koska eri ohjelmoijat voivat helposti ymmärtää vaatimukset väärin. Tämä johtuu kolmesta eri syystä, joita ovat ohjelmoijan ymmärryksen puute, huonosti kirjoitettu ja kuvattu vaatimus ja ohjelmoijan omat ennakkoasenteet (Lui & Chan, 2006.).

Ohjelmoinnissa tulisi pyrkiä siihen, että kaikki toiminnot olisivat itsenäisiä ja niin, että muutos yhteen ohjelmiston koodiin ei aiheuta muutoksia toisissa. Ohjelmoinnin suunnittelu tarkoittaa miten nämä ohjelmiston eri aliohjelmat yms. jaetaan ja koko ohjelmiston rakenne toteutetaan, jotta se olisi mahdollisimman eheä kokonaisuus (Lui & Chan, 2006). Suunnittelun tuloksena syntyy ohjelmiston arkkitehtuuri, jolla tarkoitetaan rakennetta mistä ominaisuuksista ohjelmisto koostuu ja näiden väliset yhteydet ja riippuvuudet (Clements, Garlan, Little, Nord & Stafford, 2003).

Ohjelmoinnilla tarkoitetaan ohjelman koodin kirjoittamista eli toimintaohjeiden kirjoittamista tietokoneelle jonkin tietyn tehtävän suorittamiseksi. Ohjelmoitaessa koodi kirjoitetaan valitulla ohjelmointikielellä, joka sitten käännetään konekielelle, jonka tietokone suorittaa. Ohjelmointi sisältää koodin kirjoittamisen lisäksi usein vaatimusten analysointia ja algoritmien eli toimintaohjeiden havainnollistamista, jotka sitten kirjoitetaan koodiksi (Lui & Chan, 2006).

Testauksella tarkoitetaan kirjoitetun ohjelmiston testaamista, jotta se toimii kuten pitääkin ja se ei sisällä virheitä (Lui & Chan, 2006.). Testauksen ensisijainen tarkoitus on havaita virheet ohjelmistossa, jotta nämä löydetään ja voidaan korjata ennen ohjelmiston käyttöönottoa. Testauksella voidaan varmistaa ohjelmiston virheettömyys testauksen rajoissa. Ohjelmistojen koon, riippuvuudet ja yhteydet muihin ohjelmistoihin ja ominaisuuksiin tekevät testauksesta hidasta ja aikaa vievää. Tämän vuoksi testauksella ei voida tunnistaa kaikkia ohjelmiston virheitä (Falk, Kaner & Nguyen, 1999). Testauksella löydetyt virheet ovat usein virheitä itse ohjelmiston koodissa. Virheitä voivat aiheuttaa myös puutteelliset vaatimukset, jotka aiheuttavat ohjelmiston toimimattomuuden tietyissä tilanteissa, koska niitä ei osattu ottaa huomioon. Tämän vuoksi vaatimusten ymmärtäminen onkin erityisen tärkeää ohjelmoinnissa (Huizinga & Kolawa, 2007).

Internetin yhdistämässä maailmassa yhä useammat ohjelmistot on suunniteltu toimimaan yhdessä jo olemassa olevien ja uusien ohjelmistojen kanssa (Rao, Raghunathan & Vonderembse, 1997). Näin ohjelmisto vaatii toimiakseen liitännän johonkin toiseen ohjelmistoon. Integroinnin tarkoitus on saattaa kaksi tai useampi ohjelmisto toimimaan yhdessä niin, että ohjelmistot käyttävät samoja tietoja toimittamaan tarkoitetun toimenpiteen. Integraatio on prosessi, jossa linkitetään erilaisia sovelluksia toimimaan yhtenäisenä kokonaisuutena. Integrointi vastaa fyysisten tuotteiden kokoamista. Mikäli tuotteen osia ei koota oikein, ei tuote toimi kuten pitäisi (Lui & Chan, 2006.) Integroimalla olemassa olevia ohjelmistoja toimimaan yhtenäisenä voi yritys kasvattaa tehokkuutta sekä vähentää operatiivisia kustannuksia (Rao ym., 1997).

3 HUKKA OHJELMISTOJEN TUOTEKEHITYKSESSÄ

Yrityksen tehokkuus on tärkeää yrityksen toiminnan ja kilpailuedun kannalta. Tehokkuus syntyy monen eri toiminnon optimaalisesta toteutumisesta (Poppendieck ja Poppendieck, 2003; Womack ja Jones, 2003.). Yksi tehokkuutta haittaava tekijä tuotantoteollisuudessa sekä ohjelmistojen tuotekehityksessä on hukka. Hukka (eng. Waste, jap. Muda) käsitteenä hukka on alun perin johdettu Lean-ajattelutavasta. Lean-ajattelutavan tarkoitus on poistaa hukkaa yrityksen tuotekehitysprosesseista. (Womack ja Jones, 2003). Lean-ajattelun johtamisfilosofiaa käsitellään tarkemmin seuraavassa luvussa. Hukka on alun perin Toyotan käyttämä tuotantoteollisuuden käsite. Toyota kehitti Toyota Production Systemin (TPS), Toyotan johtajan Taiichi Ohno johdolla. TPS:n tarkoitus oli poistaa hukkaa. Hukkana nähtiin kaikki prosessit, resurssit ja työ, mitkä eivät luo arvoa asiakkaalle (Ohno, 1988). Ohnon mukaan tuotantoteollisuuden hukkia ovat:

1. **Inventaario** (Inventory) hukka tarkoittaa raaka-aineiden ja valmiiden tuotteiden ylimääräistä inventaariota. Ylimääräisellä inventaariolla tarkoitetaan tekeillä olevaa työtä (Work In Progress). Tällöin varastossa on materiaalia, joita odottaa jokin tehtävä tai prosessi. Ylimääräisen inventaarion takana on yleensä erilaisia ongelmia, jotka aiheuttavat sen. Sen ansiosta toimet inventaarion pienentämiseen saattavat hyödyttää myös muita yrityksen toimintoja tehokkuuden kannalta.
2. **Ylimääräinen käsittely** (Extra Processing) tarkoittaa tuotteeseen käytettyä ylimääräistä aikaa, -resursseja tai -ponnisteluja, jotka eivät ole tarpeellisia asiakkaan tarvitseman laadun toteuttamiseen. Ylimääräinen käsittely hukkana on vaikein tunnistaa tuotantoteollisuuden hukista.
3. **Ylituotanto** (Overproduction) on tilanne, jossa tuotettu tuote ei ole tarkoitettu käytettäväksi tai myytäväksi välittömästi. Ohno kertoi ylituotannon olevan useiden muiden hukkien syy, kuten inventaarion ja odottamisen.
4. **Kuljetus** (Transportation) on tilanne, missä raaka-ainetta tai tehtävää tuotetta joudutaan kuljettamaan pidemmälle, kuin on tarve. Tämä saattaa johtua tuotantolaitoksen huonosta pohjasta. Kuljettamista tapahtuu miltei kaikissa tuotteen tuotantoprosesseissa, joten kuljetettavan matkan minimoiminen on hukan minimoimiseksi tärkeää.
5. **Odottaminen** (Waiting) tapahtuu, kun toisen työntekijän täytyy odottaa jonkin työn vapautumista toiselta työntekijältä. Odottamisesta johtuva hukka hidastaa koko tuotteen tuotantoprosessia odottamisen verran, eikä odotettua aikaa voida palauttaa tuotantoprosessin myöhäisemmissä vaiheissa.
6. **Ylimääräinen liike** (Motion) tarkoittaa tuotantoteollisuudessa kaikkea työntekijän suorittamaa ylimääräistä liikettä tuotteen valmistamisen aikana. Joissain tapauksissa liikettä saadaan pienennettyä lisäämällä työpisteen ergonomiaa. Ylimääräinen liike hidastaa työpisteen nopeutta toteuttaa sille määrätty prosessi

7. **Viat** (Defects) ovat tuotantoteollisuudessa valmiin tuotteen vikoja, jotka täytyy korjata, jotta tuote voidaan myydä asiakkaalle tai luovuttaa seuraavaan vaiheeseen. Vika tarkoittaa aina tuotteen pois heittämistä, kierrättämistä uudelleen raaka-aineeksi tai vian korjaamista. (Ohno, 1988).

Poppendieckin ja Poppendieckin (2003, 2006) esittämä ohjelmistojen tuotekehityksen hukka on johdettu tuotantoteollisuuden hukista. He esittävät, että hukka on ohjelmistojen tuotekehityksen prosessissa jotain, joka ei luo arvoa prosessille, lopputuotteelle tai asiakkaalle (Poppendieck ja Poppendieck 2003; Poppendieck ja Poppendieck 2006). Poppendieckin ja Poppendieckin näkemys eri hukista on kuitenkin muuttunut heidän julkaisuissaan vuodesta 2003 vuoteen 2006, minkä takia voidaan olettaa, että ohjelmistojen tuotekehityksen hukkaa ei ole määritelty tarkasti ja sen määritelmän voidaan odottaa muuttuvan myös jatkossa. Myös muut tutkijat ovat todenneet saman ja tutkimusten kautta esittäneet lisää erilaisia ohjelmistojen tuotekehityksen hukkia. Hukka määritellään ohjelmistojen tuotekehityksessä miksi tahansa tehtäväksi tai prosessiksi, joka vie aikaa tai muita resursseja tuotekehitysprosessissa ilman, että se tuottaa arvoa lopputuotteelle, prosessille tai aliprosesseille (Mujtaba, Feldt ja Petersen. 2010; Poppendieck ja Poppendieck. 2006; Al Baik ja Miller. 2014). Kuten Ohno (1988) on määritellyt tuotantoteollisuudelle seitsemän hukkaa, ovat myös Poppendieck ja Poppendieck määritelleet ohjelmistojen tuotekehitykselle seitsemän hukkaa heidän 2003 julkaisussaan. Poppendieckin ja Poppendieckin 2003 määrittelemiä hukkia ei ole selitetty tarkemmin tässä pro gradussa, mutta ne ovat esitetty taulukossa 1. Poppendieckin ja Poppendieckin 2006 julkaisemassa julkaisussa seitsemän ohjelmistojen tuotekehityksen hukkaa ovat:

1. **Osittain tehty työ** (Partially done work)
2. **Ylimääräiset ominaisuudet** (Extra features)
3. **Uudelleenoppiminen** (Relearning)
4. **Luovutukset** (Handoffs)
5. **Viivästykset** (Delays)
6. **Tehtävän vaihto** (Tasks switching)
7. **Viat** (Defects) (Poppendieck ja Poppendieck 2006).

Osittain tehty työ tarkoittaa vain osittain tehtyä ohjelmiston ominaisuutta, jota ei voida sen keskeneräisyyden takia julkaista tuotannossa. Osittain tehty työ on johdettu tuotantoteollisuuden Inventaario-hukasta. Syitä miksi, osittain tehdyn työn hukkaa esiintyy ovat:

1. Ominaisuuden priorisointi ilman ominaisuuden tarkkaa määrittelyä,
2. Ominaisuuden teknistä analyysiä ei ole tehty tai sitä ei ole tehty tarpeeksi tarkasti sprintin suunnittelun aikana,
3. Tehtävien odotusaikoja ei ole arvioitu oikein,
4. Eri ominaisuuksien välisiä riippuvuuksia ei ole huomioitu
5. Keskenjääneen ominaisuuden poisto sprintistä toisen ominaisuuden toteutuksen takia ja

6. Tehtävät on arvioitu puutteellisesti. (Poppendieck ja Poppendieck 2006).

Ominaisuuden priorisointi ilman ominaisuuden tarkkaa määrittelyä tarkoittaa sitä, että ominaisuus on priorisoitu ennen kuin se on vastaanotettu tuotteen omistajalta. Tämä aiheuttaa sen, että ominaisuutta aletaan kehittää ilman, että sen koko laajuus on kehittäjien tiedossa. Tällöin ominaisuus jää kesken, koska sitä ei voida toteuttaa määrittelyjen mukaan. Jos ominaisuuden teknistä analyysiä ei ole tehty tai sitä ei ole määritelty tarpeeksi tarkasti sprintin suunnittelun aikana, voi se aiheuttaa sen, että ominaisuuden teknisen vaativuuden takia sitä ei ehditä toteuttaa sprintin aikana ja ominaisuus jää kesken. Mikäli ominaisuuden tehtävien odotusaikoja ei ole pystytty arvioimaan oikein, aiheuttaa se sen, että ominaisuuden tuotekehitys tulee kestämään kauemmin kokonaisuudessaan kuin siihen on alun perin arvioitu kuluvan. Tällöin ominaisuutta ei ehditä toteuttamaan sprintin aikana ja ominaisuus jää kesken. Ominaisuudelle tulee määritellä tarkasti sen riippuvuudet muihin ominaisuuksiin. Mikäli riippuvuuksia ei oteta huomioon tarpeeksi tarkasti, voi se aiheuttaa ongelmia tuotteen kehityksessä, mikä taas aiheuttaa ongelmia aikataulussa. Ikonen, Kettunen, Oza ja Abrahamsson (2003) havaitsivat myös tutkimuksessaan jonkin tehtävän ajallisen arvioinnin epäonnistumisen osittain tehdyn työn hukkana. Tällöin jotkin muut tehtävät odottivat edellisen tehtävän valmistumista (Ikonen, Kettunen, Oza & Abrahamsson, 2010). Osittain tehdyn työn hukaksi myös lasketaan kaikki julkaisusta pois jätettävät ominaisuudet, jotka mahdollisesti poistetaan toisen ominaisuuden korkeamman priorisoinnin takia. Tällöin ominaisuus jää kesken ja se saatetaan julkaista vasta myöhemmin. Mikäli ominaisuuden kehitykseen tarvittavat tehtävät on arvioitu väärin aiheuttaa se aikatauluongelmia ominaisuuden kehitykselle, milloin ominaisuus voi jäädä ulos sprintistä ja jää kesken. (Poppendieck & Poppendieck, 2006). Ikonen ym. havaitsivat tutkimuksessaan, että osittain tehdyksi työksi koettiin tilanteet, jossa tehtävä määrättiin toiselle henkilölle, joka pystyi jatkamaan tehtävän suorittamista paremmin (Ikonen, ym. 2010.) Dahlmanin, Olssonin ja Akillioglun (2014) mukaan osittain tehdyksi työksi voidaan myös laskea pitkäaikaiset tehtävät yrityksen ideatasolla eli backlogilla. Tehtävät ovat jo päätetty toteutettavaksi ja ne voivat olla osittain suunniteltuja (Dahlman, Olsson & Akillioglu. 2014.).

Ylimääräiset ominaisuudet tarkoittavat ominaisuuksia, joita ei ole määritelty, mutta ne ovat silti toteutettu ja ominaisuuksia, joita kukaan ei käytä ohjelmistossa. Ylimääräiset ominaisuudet hukka ohjelmistojen tuotekehityksessä on johdettu ylituotannosta. Ylimääräisiä ominaisuuksia syntyy, jos tuotteesta ei ole tarkkaa visiota tai sitä ei ole määritelty tarpeeksi tarkasti. Tällöin syntyy vääriä päätöksiä tuotteen ominaisuuksista ja siitä syntyy hukkaa. Mikäli ei ymmärretä tuotteen käyttäjiä tai tuotteen tilaajan määrittelyjä saattaa tuotteeseen joutua ylimääräisiä ominaisuuksia, joille tilaajalla tai käyttäjillä ei ole tarvitta. Tuotteen ominaisuuksia voidaan myös priorisoida tuotteeseen väärin. Tällöin ominaisuudet, joita ei vielä tarvita tuotteessa johtuen esimerkiksi sen riippuvuuksista muihin ominaisuuksiin, ovat ne turhia. Ylimääräisiä ja turhia ominaisuuksia saatetaan myös lisätä tuotteeseen tilaajan määrittelyjen tai käyttäjien tarpeiden ulkopuolelta, jotta tuote näyttää asiakkaalle tai käyttäjälle

paremmalta. Tätä kutsutaan "Gold Platingiksi" Tällaiset ominaisuudet ovat kuitenkin hukkaa. Gold Platingia käsitellään tarkemmin Al-Baik ja Millerin (2014) hukkien osuudessa. (Poppendieck & Poppendieck, 2006). Ikonen ym. (2010) tutkimuksessa ylimääräisiksi ominaisuuksiksi havaittiin tilanteet, joissa asiakkaan sitoutuminen projektiin oli heikkoa ja tuotekehitystiimi teki päätöksiä oletusten perusteella. Lisäksi "hypettäminen" eli tilanteet, joissa tuotekehitystiimi sai idean ja kehitti ominaisuuden, joka kuulosti ideatasolla hyvältä, mutta ei kuitenkaan vastannut asiakkaan ominaisuutta (Ikonen, ym. 2010). Wang, ym. (2012) tutkimuksessa kävi ilmi, että tehtävien prosessien ja tehtävien, jotka tuottavat ominaisuuksia, tulisi olla linjassa asiakkaan strategian kanssa. Näin vältytään ylimääräisiltä ominaisuuksilta tuotteessa (Wang, ym. 2012).

Uudelleenoppiminen ohjelmistojen tuotekehityksessä tarkoittaa tilannetta, jossa ohjelmiston tuotekehitystiimin osaamista tai tietämystä ei käytetä hyväksi tai sitä ei ole saatavilla tiimin yksilöiden kesken. Tämä aiheuttaa sen, että tiimin sisällä yksilöt tai pienemmät tiimit joutuvat oppimaan asioita, vaikka tämä osaaminen löytyisi jo joltain toiselta yksilöltä tai tiimiltä. Näitä voivat olla erilaiset menetelmät, prosessit tai työkalut. Uudelleenoppimisessa käytetään aikaa turhaan. Uudelleenoppimisen hukka saattaa johtua siitä, että osaamisen tai tietämyksen jaolle ei ole prosessia tai menetelmää. Mikäli osaamista ja tietämystä ei dokumentoida, on suuri todennäköisyys, että uudelleenoppimisen hukkaa syntyy. Uudelleenoppimista aiheuttaa myös kommunikaation puute tiimin sisällä tai tiimien välillä. Tällöin osaamisen- tai tiedon jakoa ei voi tapahtua. Lisäksi osaamisen- ja tiedon jaossa voi esiintyä ongelmia, jos tiimi on hajautettu työskentelemään eri toimipisteissä. Uudelleenoppiminen on johdettu ohjelmiston tuotekehityksen hukista tuotantoteollisuuden ylimääräisen käsittelyn hukasta. (Poppendieck & Poppendieck 2006). Uudelleenoppimiseen voidaan myös rinnastaa työntekijöiden tiedon hukkaa, eli tietouden häviämistä työntekijöiden eri kompetensseista tai tiedon käyttämättä jättämistä (Ikonen, ym. 2010.) Myös Wang, Conboy & Cawley (2012) kuvaavat työntekijöiden käyttämättömän luovuuden yhdeksi hukaksi, joka on rinnastettavissa uudelleenoppimiseen (Wang, Conboy & Cawley. 2012).

Luovutukset tarkoittavat ohjelmistojen tuotekehityksessä tilanteita, joissa jokin ominaisuuden toteuttaminen tai tehtävä luovutetaan seuraavalle henkilölle tai tiimille. Tällaisia luovutuksia voivat olla esimerkiksi ominaisuuden luovuttaminen analyysin tekevältä yksiköltä suunnittelijalle, siitä eteenpäin koodaajalle ja edelleen testaajalle. Luovutuksissa syntyvä hukka on tietämyksen katoamista luovutuksen aikana. Poppendieck ja Poppendieck (2006) väittävät, että jokaisessa luovutuksessa häviää 50% ominaisuuden tietämyksestä. Tällöin suuri määrä luovutuksia aiheuttaa erittäin suuren tietämyksen hukan määrän. Hukka, joka aiheutuu luovutuksista voi johtua kolmesta eri syystä 1) ominaisuuden toteuttaminen vaatii tietyn prosessin, jotta se voidaan toteuttaa, 2) ominaisuuden toteuttamiseen vaaditut henkilöt tai tiimit työskentelevät hajautetusti tai 3) ominaisuuden toteuttamiseen tarvittava informaatio ei ole läpinäkyvää ja kaikkien saatavilla. Luovutusten hukka on johdettu tuotantoteollisuuden kuljetuksen hukasta. (Poppendieck & Poppendieck, 2006). Mandic, ym. 2010 havaitsivat tutkimuksessaan, että päätöksenteon välttely

aiheutti hukkaa ohjelmistojen tuotekehityksessä. Päätöksenteko luovutettiin toiselle henkilölle, tiimille tai asiakkaalle (Mandic, ym. 2010).

Viivästykset tarkoittavat ohjelmistojen tuotekehityksessä tilanteita, joissa arvoa tuottavan aktiviteetin toteuttaminen tai aloittaminen viivästyy. Tällöin ominaisuudelle ei pystytä toteuttamaan sen valmistumiseksi tarpeellisia aktiviteetteja ja ominaisuuden toteuttamisen aikataulu venyy. Viivästykset aiheuttavat sen, että asiakas ei saa arvoa mahdollisimman nopeasti. Viivästykset myös aiheuttavat tuotekehitysprosessiin katkonaisuutta, joka aiheuttaa muita hukkia kuten tehtävien vaihtoa ja ylimääräisiä ominaisuuksia. Tehtävien vaihtoa tapahtuu, jos työntekijä ei saa odottamaansa tehtävää työn alle, vaan joutuu odottamaan ja siirtyy tekemään jotakin toista tehtävää ennen kuin saa odottamansa tehtävän työn alle. Ylimääräisiä ominaisuuksia syntyy, jos tuotekehitystiimi joutuu odottamaan päätöstä työn aloittamisesta ja toteuttaa sillä välin mahdollisesti muita ominaisuuksia, jotta tiimillä on työtä. Viivästyksiä aiheutuu, jos tiimissä ei ole työn tekemiseen tarvittavia työntekijöitä, tiimissä noudatetaan työn toteuttamiseen tarpeettomia prosesseja tai tiimillä on liian monta yhtäaikaista tehtävää, tehtävän toteuttaminen riippuu muista prosesseista tai tehtävistä. (Poppendieck & Poppendieck 2006).

Tehtävän vaihto tarkoittaa ohjelmistojen tuotekehityksessä sitä, jos tehtävän toteuttaja (yksilö tai tiimi) vaihtaa tehtävää ennen kuin edellinen tehtävä on suoritettu loppuun. Tehtävän vaihto aiheuttaa sen, että tehdylle työlle ei synny arvoa sen tekemisen aikana, koska se jää kesken. Arvo realisoituu vasta, kun tehtävä on suoritettu loppuun. Tehtävien vaihto myös aiheuttaa muita hukkia kuten viivästyksiä ja osittain tehtyä työtä. Mahdollisia syötä tehtävien vaihdolle ovat ulkopuolelta tulevat keskeytykset meneillään olevaan tehtävään, tiimi tekee montaa eri tehtävää yhdenaikaisesti ja, jos kehitystiimin ja tuotteen omistajan välinen kommunikaatio ei toimi. (Poppendieck & Poppendieck, 2006). Ikonen ym. (2010) havaitsivat tutkimuksessaan, että tehtävän vaihdosta aiheutui hukkaa, koska tehtävän tekijän täytyi orientoitua uuteen tehtävään ja se nähtiin hukkana (Ikonen, ym. 2010).

Viat ovat ohjelmiston ongelmia, jotka eivät toimi halutulla tavalla. Viat aiheuttavat hukkaa niiden olemassaololla, koska ne vaativat korjauksen. Poppendieck ja Poppendieck (2006) ovat määritelleet vian aiheuttamalle hukalle laskentakaavan: hukka = (vian aiheuttama vaikutus) x (aika, jolloin vikaa ei huomata). Tällöin vian aiheuttama vaikutus kasvaa, jos vika pysyy kauemmin huomaamatta. Vikoja syntyy ohjelmistoihin monesta eri syystä. Näitä syitä ovat mm. väärin ymmärretyt määritelmät, ohjelmiston tuotekehitystiimin osaamattomuus, testaajien myöhäinen osallistuminen, automaatiotestaamisen huomiotta jättäminen ja hyväksyntäperusteiden puuttuminen. Ikonen ym. (2010) havaitsivat tutkimuksessaan, että viat, jotka havaittiin myöhemmin, aiheuttivat eniten ylimääräistä työtä ja sitä myöten hukkaa tiimille (Ikonen, ym. 2010).

Al-Baik ja Miller (2014) tutkivat artikkelissaan Waste identification and elimination in information technology organizations tuotantoteollisuudesta johdettuja hukkia ja johtavat niistä ohjelmistojen tuotekehitykseen paremman soveltuvia hukan määritelmiä ja kuvauksia. Tutkimuksessaan Al-Baik ja Miller tekevät tutkimuksen ainoastaan yhdestä organisaatiosta, minkä takia tutkimus ei ole yleistettävissä. Al-Baik ja Miller (2014) esittävät, että ohjelmistojen

tuotekehityksen hukkia ei voi suoraan johtaa tuotantoteollisuuden hukista. (Al-Baik & Miller, 2014). Ohjelmistojen tuotekehityksen hukkaa on tutkittu vain vähän, ja näissäkin tutkimuksissa on käytetty pääasiallisina hukan määritelminä tuotantoteollisuuden hukan määritelmiä teoreettisena pohjana. Myös Al-Baik ja Miller (2014) käyttävät tuotantoteollisuudesta johdettuja hukkia teoreettisena pohjana itse määrittelemilleen hukille, mutta he ovat näiden lisäksi tunnistaneet tutkimuksessaan tuotantoteollisuudesta johdetuista hukista poikkeavia hukkia. (Al-Baik & Miller, 2014). Heidän tutkimuksensa on kuitenkin käytetty vain yhtä yritystä, minkä takia heidän tutkimus ei ole yleistettävissä muihin ohjelmistojen tuotekehitystä tekeville yrityksille. Al-Baikin ja Millerin (2014) mukaan hukkia ovat:

1. **Ylimääräinen käsittely, työkalut ja metodit** (Gold Plating)
2. **Liian tarkat määrittelyt** (Over specification)
3. **Asiakkaan osallistumisen puute ja väärät oletukset** (Lack of customer involvement and inappropriate assumptions)
4. **Kahdennettu käsittely ja kahdennetut prosessit** (Double handling / duplicate processes)
5. **Keskitetty päätöksenteko** (Centralized decision making)
6. **Odottaminen** (Waiting)
7. **Viivästynyt vahvistus ja kelpuutus** (Deferred verification and Validation)
8. **Viat** (Defects)
9. **Vanhentunut informaatio tai -versio tuotteesta** (Outdated information / obsolete working version)

Gold Plating Al-Baikin ja Millerin (2014) mukaan tarkoittaa tarpeettomien työvälineiden, järjestelmien tai metodien käyttöä ohjelmistojen tuotekehityksessä. Raporttien tuottamista ja jakamista, jotka eivät ole tarpeellisia, tai informaation ylituotantoa. Gold Platingilla tarkoitetaan myös tarpeettomien kosmeettisten ominaisuuksien toteuttamista ohjelmistoon sekä pyrkimystä täydellisten ohjelmistojen tuottamiseen sen sijaan, että tuotettaisiin ohjelmistoja, jotka ovat riittäviä. (Al-Baik & Miller, 2014).

Liian tarkat määrittelyt Al-Baikin ja Millerin (2014) mukaan tarkoittaa liian tarkkaa ohjelmiston analysointia ja määrittelyä, joka ei tue päätöksentekoa. Tätä ovat sellaisen tiedon etsiminen, jota on hankala löytää sekä ohjelmiston ylisuunnittelu. (Al-Baik & Miller, 2014).

Al-Baik ja Miller (2014) esittävät hukaksi poiketen Poppendieckin ja Poppendieckin (2003, 2006) määrittelystä **asiakkaan osallistumisen puutteen ja väärät oletukset.** Asiakkaan osallistumisen puute aiheuttaa väärinymmärryksiä asiakkaan tarpeista ja vaatimuksista. Tämän lisäksi se saattaa aiheuttaa pitkät hyväksyntäprosessit ohjelmiston tuotekehityksessä sekä aiheuttaa väliaikaisia toteutuksia ohjelmistoon. Asiakkaalta saatavan informaation puute aiheuttaa vääriä oletuksia ohjelmiston kehityksessä. (Al-Baik & Miller, 2014). Korkala ja Mauer (2014) havaitsivat tutkimuksessaan kolme erilaista hukkaa, jotka liittyivät asiakkaan ja toteuttajan väliseen kommunikaatioon; 1) osallistumisen puute, 2) yhteisen ymmärryksen puute, 3) rajoitettu pääsy informaatioon. Yhteisen

ymmärryksen puutteella tarkoitetaan sitä, että asiakas ja toteuttaja ovat ymmärtäneet saman asian eri tavalla. Tällöin syntyy hukkaa. Joissain tilanteissa toteuttaja tekee vääränlaisen ominaisuuden. Joissakin tilanteissa toteuttaja tekee vääriä oletuksia asiakkaan tarpeesta ja tekee silloin ylimääräistä työtä. Yhteisen ymmärryksen puute johtuu kommunikaation puuttumisesta tehtävää kohtaan. Rajoitettu pääsy informaatioon aiheuttaa tietotason puutteen, joka saattaa aiheuttaa vääriä tehtäviä tai oletuksia asiakkaan tarpeista. Lisäksi tieto saattaa olla hajautettu eri tietolähteisiin, joka aiheuttaa tiimin tai henkilön tiedon puutetta. (Korkala & Mauer. 2014). Wang, ym. (2012) kuvaavat ylimääräisen tehtävänä olevan työn yhdeksi hukaksi. Tehtävien tulisi olla vedettävänä, kun sille on vapaita resursseja, eikä työnnettäväksi ylemmältä tasolta työtä tekevälle tiimille (Wang, ym. 2012). Mujtaba ym. (2010) havaitsivat tutkimuksessaan, että asiakkaan osallistumisen puute kuten hidas kysymysten vastausaika aiheuttivat hukkaa. Hukka ilmeni muina hukkina kuten odottamisella, vioilla tai tehtävän vaihdolla (Mujtaba ym. 2010). Mandic ym. (2010) havaitsivat tutkimuksessaan, että epävarmuus suoritettavasta tehtävästä tai prosessista aiheutti hukkaa. Tehtävää tai prosessia suorittava tiimi joutui tehdä oletuksia asiakkaan puolesta. (Mandic ym. 2010).

Kahdennetulla käsittelyllä Al-Baik ja Miller (2014) tarkoittavat, että ohjelmistoa kehitetään tai ylläpidetään kahdessa eri tiimissä, vaikka sitä voitaisiin kehittää yhdellä tiimillä. Tiimit kehittävät samaa ominaisuutta ja tämä tapahtuu tiimien tietämättä. Kahdennetuilla prosesseilla voidaan tarkoittaa kehityksen multitaskingia, jossa priorisoinnin puute aiheuttaa sen, että ohjelmiston kehityksessä siirrytään ominaisuudesta toiseen, vaikka ominaisuutta ei ole saatu valmiiksi. Tämä aiheuttaa sen, että ajallisesti kaikkien ominaisuuksien valmistuminen keskimäärin kasvaa. (Al-Baik & Miller, 2014). Mujtaba ym. (2010) havaitsivat tutkimuksessaan, että kahdennettu testaus poisti hukkaa ohjelmistojen tuotekehityksessä. Joissain tapauksissa kahdennettu käsittely kannatti, mikäli prosessit ovat kyvykkäitä siihen. (Mujtaba ym. 2010).

Keskitetyllä päätöksenteolla Al-Baik ja Miller (2014) tarkoittavat tarpeettomia päätöksentekoprosesseja, epäselviä vastuita ja -auktoriteettia. Keskitetty päätöksenteko saattaa aiheuttaa ylimääräisiä raportointeja, jotka eivät lisää arvoa. Lisäksi, mikäli keskitetty päätöksenteko aiheuttaa resurssien päättämisen projektin alkuvaiheessa, haittaa se joustavuutta. Pitkät päätöksentekoprosessit haittaavat innovatiivisuutta ja luovuutta projektissa. Drury ym. (2012) tutkivat päätöksenteon vaikutusta ketterässä kehityksessä. Heidän tutkimus osoitti, että ketterässä kehityksessä päätöksenteko vaikeutuu, kun jokin toinen taho on vastuussa päätöksenteosta. Lisäksi keskitetty päätöksenteko saattaa aiheuttaa kehitystiimille konflikteja omien päätösten tekoon ja ne saattavat poiketa tiimin valitsemista päätöksistä. Drury ym. (2012) osoittavat keskitettyyn päätöksentekoon liittyviksi ongelmiksi seuraavat seikat:

1. Epätietoisuus vaikuttaa päätöksentekoon (Uncertainty affects decision maker)
2. Tietoa ei kerätä rationaalisesti (Information is not collected rationally)
3. Tiimi omaksuu toimintatapoja päätöksentekoon (Behaviors are adapted)

Epätietoisuus aiheuttaa sen, että kehitystiimi luottaa muiden päätöksentekoon, eikä ole valmis ottamaan vastuuta päätöksenteosta. Mikäli tietoa päätöksentekoa varten ei kerätä rationaalisesti, vaan päätöksentekoon vaikuttava tieto on kerätty vääristä lähteistä, tai sitä ei kerätä ollenkaan, kehitystiimin päätöksenteko on seurausta muiden tekemistä päätöksistä, koska tiimillä ei ole saatavilla tietoa päätöksentekoa varten. Mikäli tiimi on omaksunut toimintatapoja päätöksentekoa varten aikaisemmista päätöksistä, joiden teko on keskitetty, voi päätöksenteko jatkua keskitettynä, vaikka tiimillä olisi mahdollisuus tehdä omia päätöksiä. (Drury ym. 2012.)

Odottamisella Al-Baikin ja Millerin (2014) mukaan tarkoitetaan kaikkea odottamista, jota ohjelmistojen tuotekehityksessä saattaa syntyä. Näitä ovat esimerkiksi katselmuksen tai hyväksynnän odottaminen, informaation odottaminen, mitä tarvitaan työtehtävän suorittamiseen ja päätöksenteon odottaminen. (Al-Baik & Miller, 2014). Ikonen ym. (2010) havaitsivat tutkimuksessaan, että odottamista voivat olla tiedon odottaminen tehtävän suorittamista varten, vapaan koodin katselmoijan odottaminen tai tuotantoonviennin odottaminen (Ikonen, ym. 2010)

Viivästynyt vahvistus ja kelpuutus. Al-Baikin ja Millerin (2014) mukaan se tarkoittaa sitä, että asiakkaalta ei saada vahvistusta tai kuittausta työlle. Se on myös standardien tarkkaa seuraamista, joka aiheuttaa testauksen puuttumisen. Sillä tarkoitetaan myös testauksen koulutuksen vähyyttä, testauksen tärkeyden tietoisuuden puutetta, testauksen ajan puutetta ja testauksen resurssien puuttumista. (Al-Baik ja Miller, 2014).

Vioilla Al-Baik ja Miller (2014) sekä sekä Poppendieck ja Poppendieck (2003, 2006) tarkoittavat vikoja ohjelmistossa, joka aiheutuu väärinymmärretystä tai epäselvästä asiakkaan tarpeesta (Al-Baik & Miller, 2014; Poppendieck & Poppendieck, 2003; (Poppendieck ja Poppendieck, 2006).

Vanhentuneella informaatiolla tai versiolla Al-Baik ja Miller (2014) tarkoittavat integraatiosyklin hitautta, puuttuvia tai virheellisiä dokumentaatioita monimutkaisille ohjelmistoille sekä raporttien jakamista liian aikaisessa vaiheessa, joka aiheuttaa tiedon nopean vanhenemisen (Al-Baik & Miller, 2014). Korkala ja Mauer havaitsivat tutkimuksessaan, että puuttuva informaatio johtui kommunikaation puutteesta eri tiimien tai henkilöiden välillä. Havaittiin, että etenkin tuoteomistajan eli tuotetason henkilön vanhentunut informaatio aiheutti hukkaa hänen tehtävilleen. Hän ei pystynyt tehdä päätöksiä asioista vanhentuneen informaation takia (Korkala & Mauer. 2014). Mandic, ym. (2010) havaitsivat tutkimuksessaan, että rajoitettu pääsy informaatioon aiheutti hukkaa sitä tarvitsevalle tiimille tai henkilölle. Informaatiota tarvittiin tehtävän tai prosessin suorittamiseksi laadukkaasti. He havaitsivat myös, että vääristynyt informaatio aiheutti hukkaa. Tiimi tai henkilö joutui hankkimaan uutta tietoa tai suoritti tehtävän vanhentuneella informaatiolla. (Mandic ym. 2010)

Alla on esitetty eri julkaisijoiden määritelmät hukasta. Määritelmät ovat jaettu samankaltaisuuden mukaan riveittäin. Lisäksi taulukkoon on merkattu, millä tasoilla aikaisempien tutkimusten mukaan hukkaa esiintyy. Esiintyvyys on merkattu, jos yksi tai useampi julkaisu on esittänyt hukan esiintyvän sillä

ohjelmistojen tuotekehityksen tasolla tai kuvannut esimerkin hukasta jollakin ohjelmistojen tuotekehityksen tasolla.

TAULUKKO 1 Hukat ja niiden esiintyvyys eri ohjelmistojen tuotekehityksen tasoilla.

Poppendieck ja Poppendieck (2003)	**Poppendieck ja Poppendieck (2006)**	**Al-Baik ja Miller (2014)**	**Wang, ym. (2012)**	**Mujtaba, ym. (2008)**	**Ikonen, ym. (2010)**	**Mandic, ym. (2010)**	**Esiintyy portfolio-tasolla**	**Esiintyy tuote-tasolla**	**Esiintyy tuoteke-hitysta-solla**
Osittain tehty työ	Osittain tehty työ				Osittain tehty työ		X	X	X
Ylimääräiset prosessit		Ylimääräiset työkalut ja metodit	Liian suuri määrä yhdenaikaista työtä	Ylimääräiset prosessit	Ylimääräiset prosessit				X
Ylimääräiset ominaisuudet	Ylimääräiset ominaisuudet	Liian tarkat määrittelyt			Ylimääräiset ominaisuudet			X	X
Tehtävän vaihto	Tehtävän vaihto				Tehtävän vaihto			X	X
Odottaminen	Viivästykset	Odottaminen		Odottaminen	Odottaminen		X	X	X
Liike	Luovutukset	Keskitetty päätöksenteko		Liike	Liike	Päätöksenteon välttely	X	X	X
Viat	Viat	Viat			Viat				X
	Uudelleenop-piminen		Työntekijöiden käyttämätön luovuus	Asiantuntijoiden heikko saatavuus		Rajoitettu pääsy tarpeelliseen tietoon		X	X
		Kahdennettu käsittely ja prosessit							X
		Asiakkaan osallistumisen puute ja väärät oletukset	Asiakkaan osallistumisen puute	Asiakkaan sitoutumisen puute		Epävarmuus		X	X
		Vanhentunut informaatio tai versio tuotteesta				Tiedon vääristyminen	X	X	X

Käytämme tässä tutkimuksessa hukan määritelminä Poppendieckin ja Poppendieckin (2006) esittämiä ohjelmistojen tuotekehityksen hukkien määritelmiä sekä niiden tukena Al-Baikin ja Millerin (2014) määrittelemiä hukkia.

4 HUKAN HALLINTA JA MINIMOINTI OHJELMISTOJEN TUOTEKEHITYKSESSÄ

Hukan minimoinnin tavoite on pysyvästi kehittää ohjelmistojen tuotekehityksen tehokkuutta (Al-Baik & Miller, 2014). Jotta hukkaa voidaan minimoida, täytyy se ensin tunnistaa. Hukan tunnistamiseen, hallintaan ja minimoimiseen voidaan käyttää erilaisia menetelmiä. Tässä kappaleessa kerromme kirjallisuuskatsauksessa eniten ilmenneistä menetelmistä: 1) Leanista, joka on ajattelutapa, jonka yksi prinsiippi on hukan eliminointi (Wang, ym. 2012; Womack & Jones. 2003; Mujtaba ym. 2010), 2) Kanbanista, joka on Leanin operatiiviseen käyttöön tarkoitettu visuaalinen työkalu (Ikonen ym. 2010; Ahmad, Markkula & Oivo. 2013), 3) Value Stream Mappingista, joka visualisoi koko tuotekehitysprosessin ja siihen sisältyvät arvoa tuottavat ja ei arvoa tuottavat tekijät (Lehtonen, Kilamo, Suonsyrjä & Mikkonen, 2016; Dahlman ym., 2014), 4) Statistical Process Controlista, jota käytetään ohjelmistojen tuotekehityksessä tuotekehitysprosessin tilastolliseen mittaukseen (Vashisht. 2014; Womack & Jones. 2003.), 5) Six Sigmasta, jonka laajennettu versio Lean Six Sigma on joukko menetelmiä yrityksen tuotekehityksen prosessien kehittämiseen (Raffo, Mehta, Anderson & Harmon, 2010.)

4.1 Lean ohjelmistojen tuotekehityksessä

Lean ohjelmistojen tuotekehityksessä on samoin kuin hukka johdettu tuotantoteollisuudesta (Wang, ym. 2012). Lean on alunperin Toyota Production Systemissä käytetty termi, jota ei ole tarkkaan määritelty. Lean on joukko menetelmiä, joilla on tarkoitus optimoida koko tuotantoprosessi (Ohno, 1988). Mujtaban, Feldtin ja Petersenin (2008) mukaan lean on lähestymistapa ohjelmistokehitykseen, jonka tarkoituksena on kehittää prosessien tehokkuutta. Leanin mukaisessa tuotekehityksessä systemaattisesti pyritään vähentämään hukkaa. Leanin tarkoitus on lyhentää aikaa, jolla tuote saadaan markkinalle ja parantaa asiakastyytyväisyyttä. (Mujtaba, ym. 2010). Womackin ja Jonesin (2003) mukaan lean on ajattelutapa, jonka tarkoitus on määritellä arvo ja luoda mahdollisimman tehokas arvoa luova prosessi tuotekehitykseen. 1) Arvo on asiakkaan määrittelemä asia, ja se tulee tarkkaan ymmärtää. 2) Arvovirta on prosessi, joka luo asiakkaan määrittelemän arvon. 3) Arvovirtakuvaus on dokumentti, joka tunnistaa kaikki prosessin vaiheet ja kaikkien vaiheiden arvon prosessille. 4) Asiakas luo tarpeen tuotteelle ja varmistutaan siitä, että mitään ei tehdä ennen sitä tarvitaan. 5) Prosessille tavoitellaan täydellisyyttä jatkuvasti ja hukkaa pyritään tunnistamaan ja poistamaan. (Womack ja Jones, 2003)

Leanin käytänteet tuotantoteollisuudessa ja ohjelmistojen tuotekehityksessä poikkeavat jonkin verran johtuen siitä, että tuotantoteollisuudessa tuotetaan fyysisiä tuotteita, kun taas ohjelmistojen tuotekehityksessä luodaan pääasiassa ohjelmistoja, jotka eivät ole fyysisiä (Middleton & Joyce, 2010). Lean termin ohjelmistojen tuotekehitykseen esitti

Poppendieck ja Poppendieck (2003). Heidän näkemyksensä mukaan ohjelmistojen tuotekehityksessä Leanin mukaan on 7 periaatetta, joiden mukaan tulee toimia, jotta noudatetaan Leania ohjelmistojen tuotekehitystä. Poppendieckin ja Poppendieckin (2003) näkemys Leanin ohjelmistojen tuotekehityksen periaatteista on muuttunut heidän 2006 teokseen nähden, käytämme 2006 teoksessa esitettyjä Leanin periaatteita:

1. Eliminoi hukka (Eliminate Waste)
2. Rakenna laadukasta (Build Quality in)
3. Luo tietämystä (Create Knowledge)
4. Päätä myöhään (Defer Commitment)
5. Toimita nopeasti (Deliver Fast)
6. Kunnioita ihmisiä (Respect People)
7. Optimoi täysin (Optimize the Whole)

Jotta 1) hukka voidaan eliminoida, on tärkeää, että se tunnistetaan (Poppendieck & Poppendieck 2006; Al-Baik & Miller, 2014). Se mitä hukkia ohjelmistojen tuotekehitysprosessissa voi ilmetä, riippuu yleensä siitä, mitä prosesseja ja menetelmiä ohjelmistojen tuotekehityksessä käytetään (Al-Baik & Miller 2014). Ohjelmistojen tuotekehityksessä voidaan käyttää erilaisia menetelmiä hukan tunnistamiseen kuten arvovirtakuvantamista (Value Stream Mapping) ja statistista prosessin kontrolloimista (Statistical Process Control) (Wang ym. 2012). Niistä kerrotaan lisää tässä pro-gradussa myöhemmin. Hukan tunnistamisen jälkeen hukan syy tunnistetaan ja se eliminoidaan tarvittavilla keinoilla (Al-Baik & Miller, 2014). Leanin periaatteita noudatetaan eri määrin organisaatioissa. Wangin ym. (2012) tutkimuksen mukaan hukan eliminointi oli yksi eniten noudatetuista Leanin periaatteista.

Poppendieck ja Poppendieck (2006) esittävät, että 2) rakentamalla laadukasta ohjelmistoa, testaamalla heti rakennusvaiheessa, on yksi keino vähentää vikojen hukkaa. Testaamisen tarkoitus ei ole etsiä vikoja, vaan varmistaa, että ohjelmisto toimii asiakkaan määrittelemällä tavalla. Viat ohjelmistossa pyritään tunnistamaan mahdollisimman nopeasti ja niiden julkaisemista välttämään (Wang ym. 2012).

3) Luomalla tietämystä koko ohjelmistojen tuotekehitysprosessin ajan voidaan minimoida uudelleenoppimisen hukkaa. Koko ohjelmistojen tuotekehityksen prosessi on tietämystä luova prosessi, joka vaatii tietämyksenjakojärjestelmän, jotta siinä syntyvä tietämys voidaan dokumentoida, jakaa ja käyttää prosessin aikana tai myöhemmin. (Poppendieck & Poppendieck, 2006). Middleton & Joyce (2010) havaitsivat tutkimuksessaan, että nopeat kehityssprintit, kanban-taulun käyttö ja pieni yhtäaikaisten toteutettavien tehtävien määrä lisäsi tietämystä kehitystiimin sisällä ja kehitti kehitystiimin nopeutta suorittaa tehtäviä. Mujtaba, ym. (2008) havaitsivat tutkimuksessaan, että prosessin tai tehtävän asiantuntijoiden puuttuminen aiheutti sen, että prosessi tai tehtävä saatiin hitaasti valmiiksi (Mujtaba ym. 2008). Tietämystä voidaan käyttää päätöksentekoon prosessin aikana tai myöhemmissä ohjelmistojen tuotekehitysprojekteissa. Tämä vähentää tarvetta keskitetylle

päätöksenteolle, ja tiimi voidaan valtuuttaa päätöksentekoon. Poppendieckin ja Poppendieckin (2006) mukaan ohjelmistojen tuotekehityksessä on tärkeää, että

4) päätökset tehdään niin myöhään kuin mahdollista. Jos päätökset tehdään prosessin alkuvaiheessa, on tapahtuviin ongelmiin vaikeampaa vaikuttaa ja ratkaista. Myöhäinen päätöksenteko pienentää riskiä ongelmille ja antaa mahdollisuuden siirtää tai poistaa ongelma kokonaan prosessista.

5) Toimittamalla nopeasti Poppendieck ja Poppendieck (2006) tarkoittavat mahdollisimman nopeaa sykliä, jossa toimitetaan asiakkaan määrittelemää ohjelmistoa ja kehitetään ohjelmistojen tuotekehityksen prosessia. Toimittamalla asiakkaalle nopeasti, voidaan saada asiakkaalta palautetta ohjelmistosta nopeissa sykleissä, jolloin ohjelmistoa voidaan korjata palautteen mukaan mahdollisimman nopeasti seuraavaan sykliin ja korjauksesta pyytää palautetta uudelleen.

6) Ihmisten kunnioittamisella Poppendieck ja Poppendieck (2006) tarkoittavat koko organisaation ja tiimien ihmisten kunnioittamista ja heidän voimavarojensa hyödyntämisestä. Organisaation tulisi ottaa työntekijöiden tietämys ja osaaminen huomioon ja käyttää niitä oikein, mikäli ihmisten tietous tulee tietoon. Näin voidaan jakaa tietämystä organisaation läpi ja vähentää uudelleenoppimista. Lean organisaatio pyrkii toimittamaan arvoa ensimmäisen periaatteen, hukan minimoimisen mukaisesti.

7) Optimoimalla koko prosessi tarkoitetaan hukan minimoimista, koko prosessin jatkuvaa kehittämistä ja mahdollisimman nopeaa arvon tuottamista asiakkaalle. (Poppendieck & Poppendieck, 2006). Leanin mukaiseen ohjelmistojen tuotekehitykseen on luotu erilaisia menetelmiä ja työkaluja, joilla voidaan toteuttaa Leanin periaatteiden mukaista ohjelmistojen tuotekehitystä.

Petersen (2012) kuvaa tutkimuksessaan neljä indikaattoria hukan havaitsemiseksi Leanin mukaisessa ohjelmistojen tuotekehityksessä; 1) ylläpidon pyyntöjen sisäänvirtaus, 2) visualisointi kumulatiivisten virtausdiagrammien avulla, 3) tehtävän aloittamisen ja käynnistämisen välinen aika, 4) työmäärä. Ylläpidon pyynnöillä tarkoitetaan asiakkaan pyyntöjä tuotteen ylläpitoa varten. Sisäänvirtauksen määrä voidaan laskea pyyntöjen määrällä tietyllä ajanjaksolla. Pyyntöjen määrä kuvaa tehtävän tilaa; onko tehtävä kontrollissa vai ei. Visualisoidun kumulatiivisen virtausdiagrammin avulla nähdään ylläpitopyyntöjen määrä visuaalisesti ajanjaksojen edetessä. Sen avulla voidaan myös määritellä tehtävien eri vaiheiden pyynnöt erikseen ja nähdä, mitkä tehtävät saavat eniten pyyntöjä. Tehtävän aloittamisen ja käynnistämisen välisellä ajalla tarkoitetaan aikaa, jolloin tehtävän on ollut tarkoitus alkaa, mutta milloin se on todellisesti käynnistynyt. Sen tarkastelun avulla voidaan ennustaa aika, joka kuluu, ennen kuin tehtävä oikeasti käynnistetään. Resursseja voidaan tarpeen mukaan jakaa sen avulla ja näin vähentää hukkaa. Työmäärän arvioinnilla tarkoitetaan statistista arviointia eri tehtävien työmäärästä. Sen tarkoitus on ennustaa tulevien työmäärien arviointia ja välttää liian suuren työmäärän vastaanottamista, joka aiheuttaa odottamisen hukkaa. Petersenin kuvaamien indikaattoreiden tarkoitus on ennustaa ja rajoittaa menossa olevien työn määrää, jota voidaan soveltaa esimerkiksi Kanban taulussa. (Petersen. 2012).

Wang, ym. (2012) tutkimuksessa havaittiin, että monen eri ketterän kehittämisen menetelmien yhdistäminen lean ajattelutavan kohtiin kuten hukan

minimointiin oli tehokas keino tuotekehityksen prosessien ja tehtävien optimointiin. Niiden yhdistämisen avulla oli helpompi tunnistaa hukkaa ja kehittää ohjelmistojen tuotekehityksen prosesseja ja tuotteen laatua. (Wang, ym. 2012).

4.1.1 Hukan hallinta ja minimointi Leanin eri menetelmien avulla

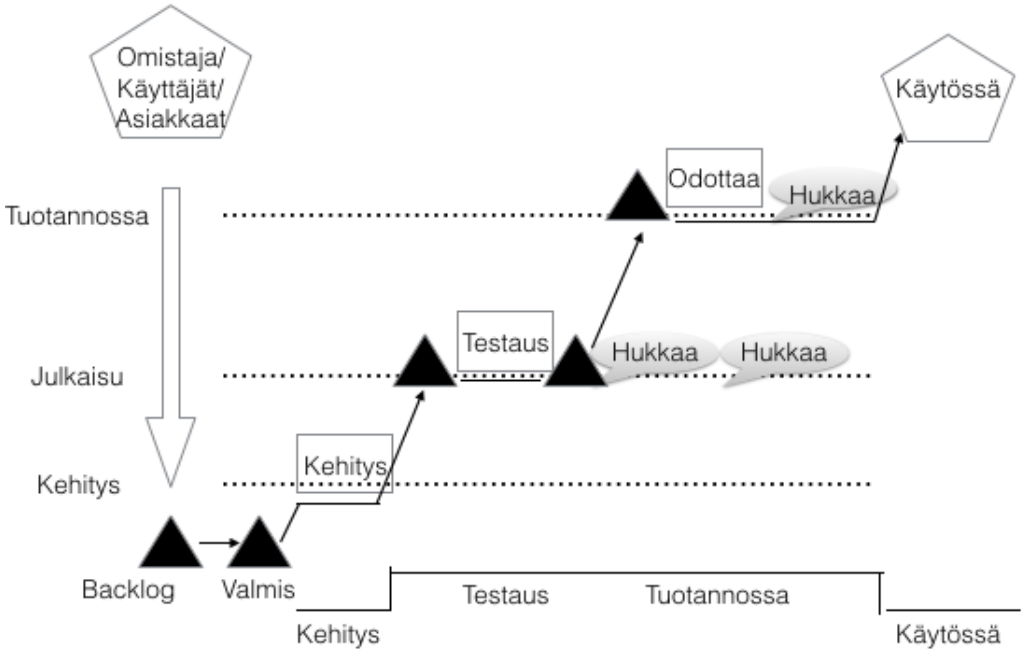
Leanin kaksi käytettyä työkalua hukan hallintaan ja minimointiin ovat Value stream mapping ja Statistical process control. Molemmille työkaluille on yhteistä se, että näissä pyritään visualisoimalla tunnistamaan tuotekehityksessä esiintyvähukka. Value stream mapping visualisoi koko tuotekehitysprosessin ja siihen sisältyvät arvoa tuottavat ja ei arvoa tuottavat tekijät (Lehtonen ym. 2016; Dahlman ym., 2014.) Statistical process control visualisoi joukon yksittäisiä vastaavia tuotekehitysprosesseja ja esittää mitkä ovat menneet suunnitellusti ja mitkä eivät (Vashisht, 2014.)

Value stream mapping (VSM) on yksi Leanin tuotekehityksen työkaluista, jonka tarkoitus on optimoida koko tuotekehitysprosessia. VSM on työkalu, joka mahdollistaa yritysten ymmärtää oman tuotekehityksen työn prosessit. VSM avulla voidaan tunnistaa mahdollisuudet prosessien kehittämiseksi. VSM auttaa tunnistamaan prosessien pullonkaulat, jotta niitä voidaan optimoida tehokkaammiksi. (Mujtaba ym. 2010.).

VSM auttaa yritystä tarkastelemaan tuotekehityksen tekijöitä, jotka tuottavat arvoa tuotekehitys putkelle. Tämä “value stream” eli arvoketju on kokoelma toimenpiteitä, joita tarvitaan tuotteen kehittämiseksi alkupisteestä asiakkaalle käyttöön. Arvoketjun toimenpiteet voidaan jakaa kahteen kategoriaan: arvoa tuottaviin ja ei arvoa tuottaviin (Lehtonen ym. 2016.). Toimenpiteet sisältävät sekä tuotekehitysprosessiin arvoa lisäävää ja arvoa vähentäviä toimenpiteitä, arvoa vähentäviä toimenpiteitä voidaan pitää hukkana. Yrityksen tuotekehityksen arvoketju koostuu näistä molemmista toimenpiteistä, jotta tuote voidaan viedä koko tuotekehitysprosessin läpi asiakkaalle käyttöön. Tuotekehityksen arvoketjun ymmärtäminen mahdollistaa tuotekehityksestä vastaavien ymmärtää ja ajatella koko tuotekehitysprosessi uudelleen arvon tuotannon näkökulmasta (Mujtaba ym. 2010.).

VSM optimoi tuotekehitystä tehokkaammaksi tarkastelemalla tuotekehityksen arvoketjun flowta. Flowlla tarkoitetaan tuotekehityksen prosessia, joka päättyy tuotteen valmistumiseen. Prosessiin liittyy tyypillisesti materiaali, tieto ja ihmiset. Tuote, jota toteutetaan, voi olla esimerkiksi uusi ominaisuus, virheen korjaus tai teknisesti uudelleen tehty tuote. Tuotteen valmistus flowlla tarkoitetaan tuotteen valmistamista raaka-aineista asiakkaalle käytettäväksi. Tässä vaiheessa tuotteen määrittelyt muutetaan arvoa tuottaviksi ominaisuuksiksi ohjelmiston käyttäjille. Tuotteen suunnittelu flowlla tarkoitetaan tuotteen valmistamista ideoinnista käyttöönottoon. Flowssa käytettävät resurssit lisäävät arvoa tuotekehitysprosessiin. Resursseja ovat mm. tuotekehitystiimi, asiakas ja ohjelmiston käyttäjät. Flown tehokkuudella tarkoitetaan, kuinka paljon flowssa kehitettävää tuotetta tehdään tietyssä ajassa. Flown tehokkuutta mitataan usein tuotteen toimitusajalla tai läpimenoajalla eli

miten nopeasti tuote saadaan prosessin läpi asiakkaalle käyttöön (Lehtonen ym. 2016.).



KUVIO 2 Value Stream Mapping

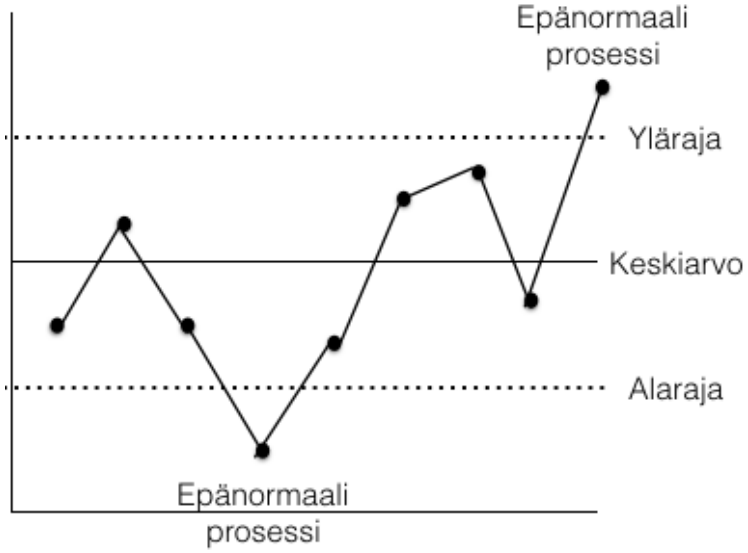
Kuviossa on kuvitteellinen yrityksen tuotekehitysputki. Vaakasuora akseli kuvaa aikaa ja siihen on lisätty aikaväli kuvaamaan, miten pitkään ominaisuus on kehityksessä, testauksessa ja tuotannossa ja käyttäjällä. Pystysuora akseli kuvaa kehitystä ideasta loppukäyttäjälle ja tähän on merkitty tasot, milloin ominaisuus on kehitettävänä, julkaisut ja tuotannossa. "Backlog" laatikko sisältää yrityksen kehitettävät ominaisuudet, "Valmis" laatikko sisältää valitut kehitettävät ominaisuudet. "Kehitys" laatikko sisältää kehityksessä olevat ominaisuudet, "Testaus" laatikko testauksessa olevat ominaisuudet. "Odottaa" laatikko sisältää ominaisuudet, jotka on kehitetty ja testattu ja jotka odottavat julkaisua asiakkaalle, "Käytössä" kuvaa, kun ominaisuus on käyttäjällä käytössä. Puhekuplat kuvaavat prosesseja yrityksen tuotekehityksessä, joissa syntyy hukkaa. Kuvitteellisessa tapauksessa hukkaa syntyy testausvaiheessa, kun testaus tapahtuu vasta julkaisun jälkeen ja kun testausprosessi kestää pitkään. Lisäksi hukkaa syntyy, kun ominaisuus on valmis ja tuotannossa mutta ei vielä käyttäjälle saatavilla.

Toinen työkalu nimeltä Statistical process control (SPC) on yksi Leanin työkaluista, jonka tarkoitus on optimoida koko tuotekehitysprosessia. SPC on muiden aikaisemmin mainittujen Leanin työkalujen mukaan johdettu tuotantoteollisuudesta (Womack & Jones, 1990). SPC on alun perin ollut Toyotan käyttämä tuotantoteollisuuden työkalu prosessien kehittämiseksi. Siinä kerätään dataa prosessin eri pisteistä ja pyritään kehittämään prosessia datan perusteella (Womack & Jones, 1990; Vasisht, 2014). Kun prosessia kehitetään, tulee datasta saatavien arvojen kehittyä, jotta prosessin kehityksessä on onnistuttu. SPC:tä voidaan käyttää hukan minimoimiseen (Womack & Jones, 1990).

SPC:tä käytetään ohjelmistojen tuotekehityksessä tuotekehitysprosessin tilastolliseen mittaukseen. Sen avulla voidaan mitata Vashishtin (2014) mukaan uudelleen tehtävää työtä (Rework), työn tuottavuutta (Productivity) ja vikojen esiintymistä (Defect density). Vashisht (2014) esittää kontrollikaaviot, joiden avulla voidaan visuaalisesti esittää eri datapisteistä saatava data määrällisesti ja niitä voidaan seurata halutuin aikavälein. Saatava data analysoidaan ja sen perusteella muutetaan ohjelmistojen tuotekehitysprosessia, jonka jälkeen mittaus suoritetaan uudelleen ja mittaukset analysoidaan. Tästä nähdään, onko tuotekehityksen prosessi kehittynyt. (Vashisht, 2014.)

SPC:n käyttäminen edellyttää, että ymmärretään kehitysprosessi ja tiedetään sen rajat. Vashishtin (2014) mukaan tuotekehitysprosessi kannattaa jakaa pienempiin prosesseihin SPC:tä käytettäessä. Näitä prosesseja ovat esimerkiksi aikaisemmin kuvatut vaiheet kuten vaatimusten analysointi, ominaisuuksien määrittely, ohjelmistoarkkitehtuurin muodostaminen, ohjelmiston toteutus, ohjelmiston testaus jne.

SPC esittää datan pisteinä kuviossa, jotka esittävät prosessien variaatiota. Kuvioon on sijoitettu ylä- ja alarajat, joiden sisäpuolelle sijoittuva data ovat normaalia variaatiota ja ulkopuolelle sijoittuva data epänormaalia. Rajojen sisällä tapahtuva variaatio on normaalia ja se kuvaa ennustettavaa prosessia, rajojen ulkopuolella tapahtuva variaatio on epänormaalia ja se kuvaa ei ennustettavaa prosessia. Rajojen ulkopuolelle sijoittuvia prosesseja yrityksen on syytä analysoida tarkemmin. Kuvion keskellä oleva viiva on datan keskiarvo. Mitä lähempänä keskiarvoa data on, sitä tehokkaampi tämä osa prosessia koko tuotekehityksestä on. (Vashisht, 2014.)

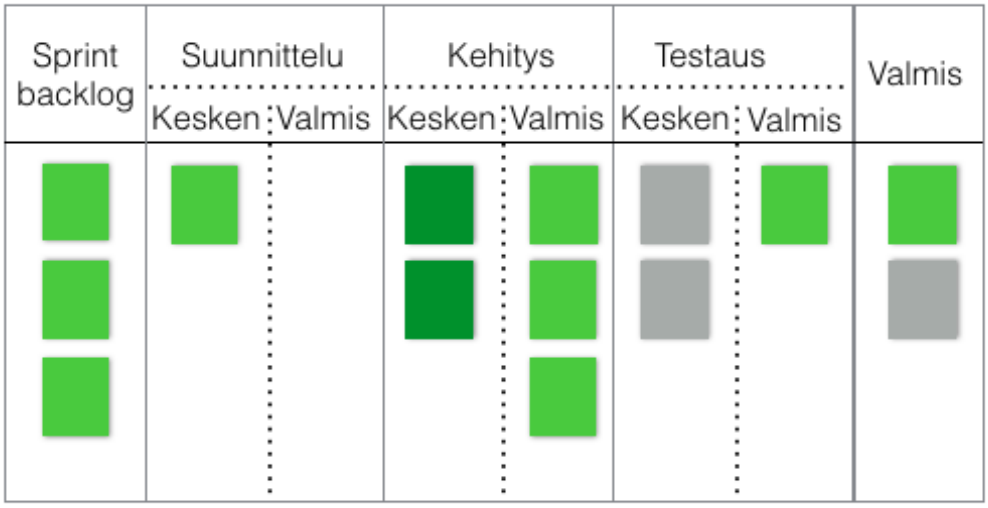


KUVIO 3 Statistical process control

SPC-kuvion keskellä on viiva, joka on datasta muodostettu keskiarvo. Katkonaiset viivat keskiarvon ylä-ja alapuolella ovat rajoja, jonka sisälle sijoittuva data on normaalia ja ulkopuolelle sijoittuva data epänormaalia ja nämä prosessit vaativat syvempää analyysia.

4.2 Kanban ohjelmistojen tuotekehityksessä

Kanbanin mukainen tuotekehityksen prosessi malli on yksi Leanin tuotekehityksen operatiivisen työn hallintaan tarkoitettu työkalu (Ikonen ym. 2010; Ahmad ym. 2013). Kanban on alunperin teollisuudesta autonvalmistaja Toyotan kehittämä tuotantojärjestelmän ohjaustyökalu. Myöhemmin Kanban on omaksuttu myös ohjelmistotuote - ja palvelukehitykseen. Kanban nimi viittaa mallissa käytettävään tauluun, johon pyritään sisällyttämään kaikki projektiin kuuluva. Taulun avulla pystytään visuaalisesti hahmoittamaan nykyiset ja tulevat tehtävät sekä niihin kuluva aika. Kanban taulun tarkoitus on vähentää yritysten tuotekehityksessä yhtaikaa tehtävien tuotteiden määrää ja tätä kautta lyhentää yrityksen valmistamien tuotteiden läpimenoaikaa (Kniberg & Skarin, 2010; Ahmad ym. 2013.)



KUVIO 4 Kanban-taulu

Kanban taulu sisältää tehtäviä, joita kuvataan post-it lapuilla. Lisäksi Kanban taulu jaetaan vaiheisiin kuvaamaan tehtävien etenemistä tuotekehitysputkessa. Kuvion kanban taulu sisältää viisi eri vaihetta, joista kolme on vielä jaettu kahteen, jotta ymmärretään mitä on työn alla ja mitkä ovat valmiita ja ne voidaan siirtää seuraavaan vaiheeseen, kun resurssit sen sallivat.
Andersonin (2010) mukaan Kanbanin perusajatus ohjelmistokehityksen kehittämiseen perustuu seuraaviin tuotekehityksen sääntöihin:

1. Työn tekemisen ja määrän visualisointi
2. Keskeneräisen työn minimointi
3. Mittaa ja hallitse työn läpimenoaikaa
4. Tee työn tekemisen prosessit selviksi kaikille
5. Kehitä yhteistyötä mallien ja tieteellisten keinojen avulla

Kniberg & Skarin (2010) toteavat, että yleisesti Kanbanissa on kolme sääntöä: visualisoi tehtävä työ, minimoi keskeneräinen tehtävä työ ja mittaa työn tekemisen nopeutta eli miten nopeasti esimerkiksi uusi ominaisuus saadaan toteutettua.

Kanban on työn tekemisen kontrolloimiseen tarkoitettu mekanismi (Ahmad ym. 2013.) Kanbanin periaatteen mukaan uuden työn aloittaminen pitäisi tapahtua vasta, kun meneillään tehtävä työ on valmis tai lopetettu niin, että uusi tuote tai ominaisuus kehitetään käyttäjille juuri silloin, kun sitä tarvitaan. Kanban pyrkii ohjaamaan yritysten tuotekehitystä vain yhteen asiaan kerralla ja jakamaan kehityksen vaiheisiin, jotta tuotteet saadaan valmistettua nopeammin ja tuote valmistuu vaiheittain ja ajallaan. Kanban on enemmän työtä ohjaava prosessi ohjelmistokehitykseen kuin työkalu. Prosessin tukena käytetään työn määrän ja keston visualisointia Kanban taulun avulla (Kniberg & Skarin, 2010).

Kanban visualisoi työn seuratakseen miten kehitys etenee eri kehitysvaiheiden läpi. Työmäärän, vaiheiden ja keston hahmottamiseksi voidaan käyttää valkotaulua ja perinteisiä tarralappuja. Lisäksi nykypäivänä on tyypillistä, että käytetään myös tähän tarkoitettuja projektinhallintaohjelmia. Nykypäivänä on tavallista, että valkotaululle kuvataan yrityksen tuotekehityssuunnitelma ja ohjelmistojen avulla määritetään tuotteiden kehityksen sisältämät vaiheet ja tehtävät yksityiskohtaisemmin. Valkotaulusta koko yritys hahmottaa oman tuotekehityksen suunnitelmat ja aikataulut. Projektinhallinta ohjelmien avulla yrityksen tuotekehitys voi suunnitella ja aikatauluttaa kehityksen tarkemmin. Lisäksi työkalujen avulla yrityksen tuotekehityksen läpimenoaikaa voidaan optimoida paremmin, kun tehtäviin kuluvaa aikaa voidaan paremmin mitata (Kniberg & Skarin, 2010.).

4.2.1 Hukan hallinta ja minimointi Kanban-taulun avulla

Ikonen ym. (2010) sanovat, että edut, joita Kanbanista saadaan ovat keskeneräisen työn vähentyminen ja työn määrä osataan mitoittaa omiin resursseihin paremmin. Kniberg & Skarin (2010) määrittävät kolme Kanbanista saatavaa hyötyä ketterään ohjelmistokehitykseen. Ensimmäiseksi Kanbanilla voidaan visualisoimalla osittaa työ vaiheisiin ja vaiheet tehtäviksi sekä arvioida tekemiseen kuluva aika paremmin. Toiseksi Kanbanilla on helpompi minimoida samanaikaisesti tehtävä keskeneräinen työ, kun ymmärretään omat resurssit paremmin. Tätä kautta voidaan rajoittaa kunkin vaiheen sisältämät tehtävät kohtaamaan resurssit paremmin ja julkaista uusia ominaisuuksia nopeammin. Kolmanneksi Kanbanin avulla voidaan mitata ja optimoida tehtäviin kuluva aika tarkemmin. Kanbanin kautta yritys voi optimoida toteutettavien tehtävien koon ja määrän, jotta tehtävän toteutuksen kesto olisi lyhyt ja ennustettavissa.

Käytettäessä Kanbania tuotekehityksen organisoimiseksi voidaan, sillä vähentää ja hallita monissa tuotekehityksen vaiheissa ja toiminnoista aiheutuvaa ja esiintyvä hukkaa. Ahmad ym. (2013) listaavat seuraavat Kanbanista saatavat hyödyt:

- Kehityksen läpimenoaika lyhenee
- Resurssien määritys helpompaa
- Tehottomien menetelmien ja prosessien poistaminen
- Parempi tiedonkulku asiakkaan tarpeiden tunnistamiseksi
- Selkeys yrityksen tuotekehityksessä

Kanbanin avulla voidaan erityisesti lyhentää tuotteen tai ominaisuuden kehityksen läpimenoaikaa eli miten nopeasti se saadaan julkaistua. Tämä saavutetaan, kun keskitytään kehittämään vain tärkeimpiä tuotteita tai ominaisuuksia. Keskittymällä vain tärkeimpiin asioihin yritys pystyy julkaisemaan tasaisella, nopeammalla tahdilla valmiita tuotteita tai ominaisuuksia käyttöön (Ahmad ym. 2013.).

Kanbanin avulla yritys onnistuu myös paremmin suhteuttamaan resurssit tehtävän työn määrään nähden. Tämä tasoittaa yrityksen kehitystiimien kehittämien tuotteiden ja ominaisuuksien kehitykseen kuluvaa aikaa paremmin, kun ennen toteutusta ymmärretään suhteuttaa resurssit työn laajuuden ja tärkeyden mukaisesti. Näin saadaan kehitettyä laadukkaampia tuotteita ja parannettua kehityksen tehokkuutta (Ahmad ym. 2013.). Middleton ja Joyce (2010) artikkelissaan saavat tulokseksi, että resurssien hyödyntäminen Kanbanin avulla niin, että resursseja käytetään silloin, kun niitä on vapaana, voidaan kehittää ohjelmistoa myös tehokkaammin. (Middleton & Joyce, 2010.).

Kanbania hyödyntämällä yritys pääsee eroon menetelmistä ja prosesseista, jotka eivät tuo arvoa koko tuotekehitysprosessiin. Käytettäessä Kanbania ei synny tilannetta, että ei tiedettäisi mitä kehitettäisiin seuraavaksi. Kanban helpottaa yritystä suunnittelemaan työn tekemisen paremmin (Ahmad ym. 2013.). Kanban kasvattaa ja tekee yhteistyöstä asiakkaan kanssa parempaa, kun asiakkaan tarpeet ymmärretään paremmin. Käytettäessä Kanbania asiakas ymmärtää tuotteenkehityksen elinkaaren paremmin, kun tuotekehitys on helposti hahmotettavissa ja miten tuotteen kehitys etenee. Asiakkaan on myös helpompi nostaa omat tarpeet paremmin esille ominaisuuksien priorisoinnissa yms. (Ahmad ym. 2013.).

Kanban parantaa ja selkeyttää informaation kulkua. Visualisoimalla kehitettävät tuotteet ja ominaisuudet, sekä kehitykseen kuluvat resurssit ja aika, on informaatio helpommin saavutettavissa ja ymmärrettävissä kaikille sidosryhmille (Ahmad ym. 2013.).

4.3 Six-Sigma ohjelmistojen tuotekehityksessä

Six-Sigma ja siitä laajennettu Lean Six-Sigma on joukko menetelmiä yrityksen tuotekehityksen prosessien kehittämiseen, joita on omaksuttu myös ohjelmistokehityksen kehittämiseen. Six-Sigma on alunperin puhelinvalmistaja Motorolan kehittämä malli, jolla pyrittiin parantamaan matkapuhelimien tuotekehitystä. Six-Sigma pyrkii tuotekehityksessä esiintyvän hukan vähentämisen lisäksi kasvattamaan koko yrityksen tehokkuutta kehittämällä ja eliminoimalla prosesseja

ja aktiviteetteja, jotka eivät kasvata tuotekehityksen arvoa. Se tarjoaa systemaattisen lähestymistavan yrityksen tuotekehityksen operaatioiden uudelleen suunnittelemiseksi ja kehittämiseksi. Six-Sigman perimmäinen ajatus on, mitä tarkemmin tuotekehityksen prosesseja hallitaan, sitä parempia tuloksia tuotekehitys saavuttaa (Raffo ym. 2010.).

Nykypäivän Lean Six-Sigman suuntaus on enemmän asiakaskeskeinen. Prosessien tehokkuuteen ja laatuun liittyvät periaatteet huomioidaan siinä, miten yrityksen ja tiimien pitäisi organisoitua, jotta asiakkaalle voidaan toimittaa paras tuote ilman, että resursseja hukataan mihinkään, mistä asiakas ei ole valmis maksamaan. Leanin Six-Sigman asiakaskeskeisyyteen liittyvä suuntaus muodostuu kahdesta yrityksen omaksumasta tekijästä. Ensimmäinen on ymmärtää käyttää niin vähän resursseja kuin mahdollista tarvittavan arvon tuottamiseksi asiakkaalle. Toinen on tunnistaa ja ymmärtää, mitkä prosessit oikeasti lisäävät tuotteen arvoa asiakkaalle. Jotta koko yritys voidaan organisoida niin, että keskiössä on asiakkaalle tuotava arvo, on ymmärrettävä yrityksen ydinosaaminen josta asiakas maksaa. Six-Sigma ajattelutavan perimmäinen ajatus on, että asiakas ei tahdo maksaa mistään muusta kuin itse tuotteesta tai palvelusta. Jotta koko organisaatio omaksuu asiakaskeskeisyyden, on yrityksen tiedostettava oma ydinosaaminen, josta myös asiakas on valmis maksamaan. (Raffo ym. 2010.).

Raffo ym. (2010) mukaan ensimmäinen askel tuotteen suunnittelussa on ymmärtää, mitä asiakas haluaa. Lean Six-sigma-ajattelussa tätä kutsutaan ”asiakkaan ääneksi”, jota käytetään kuvaamaan asiakkaan tiedostetut ja tiedostamattomat vaatimukset. Nämä voidaan kerätä lukuisilla eri keinoilla kuten keskusteluilla tai haastatteluilla, tutkimuksista, määrittelyistä, havainnoista, asiakaspalautteesta jne.

"Asiakkaan ääni" on prosessi, jota käytetään asiakkaan vaatimusten keräämiseen. Tämän prosessin on tarkoitus olla aktiivinen ja jatkuvasti innovatiivinen jotta yritys havaitsee asiakkaiden muuttuvat vaatimukset (Raffo ym. 2010.).

4.3.1 Hukan hallinta ja minimointi Six-Sigman avulla

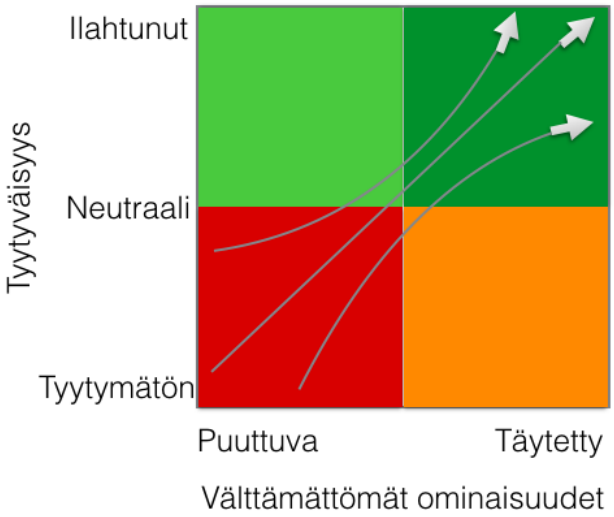
Lean Six-Sigma sisältää työkaluja tuotekehityksen prosessien tehostamiseksi ja asiakkaalle tuotettavan arvon kasvattamiseksi, näitä työkaluja ovat mm. Kano analyysi ja Quality Function Deployment.
Kano analyysi on tapa ymmärtää asiakkaan tarpeet ja mieltymykset ja priorisoida ne tehokkaammin markkinoiden tarpeiden mukaan (Raffo ym. 2010.). Kano analyysissa asiakkaan vaatimukset analysoidaan neljän eri tarpeen tyypin mukaisesti:

1. Dissatisfiers eli vaatimukset, joiden vuoksi asiakas on tyytymätön tuotteeseen tai palveluun
2. Must be´s eli välttämättömät vaatimukset, jotka asiakas huomaa ja joiden puute aiheuttaa tyytymättömyyttä tuotteeseen
3. Satisfiers eli vaatimukset, joihin asiakas on tyytyväinen ja jotka kasvattavat tyytyväisyyttä tuottetta kohtaan

4. Delighters eli piilossa olevat ominaisuudet, joita asiakas ei osannut odottaa mutta jotka kasvattavat syitä minkä vuoksia asiakas ostaa tuotteen (Raffo ym. 2010.).

Raffo ym. (2010) mukaan yhdistelmä, joka tarjoaa suurimman mahdollisen kilpailuedun yritykselle on sisällyttää tuotteeseen välttämättömät vaatimukset sekä mahdollisimman monta vaatimusta, jotka kasvattavat tyytyväisyyttä. Lisäksi tuotteeseen on sisällytettävä yksi tai useampi piilossa oleva ominaisuus, jotka kasvattavat syitä ostaa tuote.

Kano-mallia voidaan käyttää tunnistamaan tuotteen ominaisuudet, jotta yritys voi varmistaa, että tuotteessa on tarpeeksi ominaisuuksia kilpailijoihin verrattuna ja houkutellakseen tuotteelle uusia asiakkaita (Raffo ym. 2010.).



KUVIO 5 Kano-analyysi

Kano-analyysin alin käyrä heijastaa asiakkaiden perustarpeita. Nämä ovat välttämättömiä vaatimuksia, joita asiakkaat yleensä odottavat tuotteesta. Näiden ominaisuuksien läsnäolo tuotteessa ei johda korkeaan asiakastyytyväisyyteen mutta niiden puute aiheuttaa paljon tyytymättömyyttä, joka voi johtaa palautteeseen ja menetettyyn liiketoimintaan. (Raffo ym. 2010.).

Keskimmäinen käyrä heijastaa asiakkaan tyydyttäviä vaatimuksia, joiden määrä kasvattaa asiakastyytyväisyyttä. Näiden lähtökohta on ”mitä enemmän, sitä parempi”, mikä tarkoittaa, että mitä suurempi näiden toimintojen tai ominaisuuksien saatavuus on, sitä korkeampi asiakastyytyväisyys. (Raffo ym. 2010.).

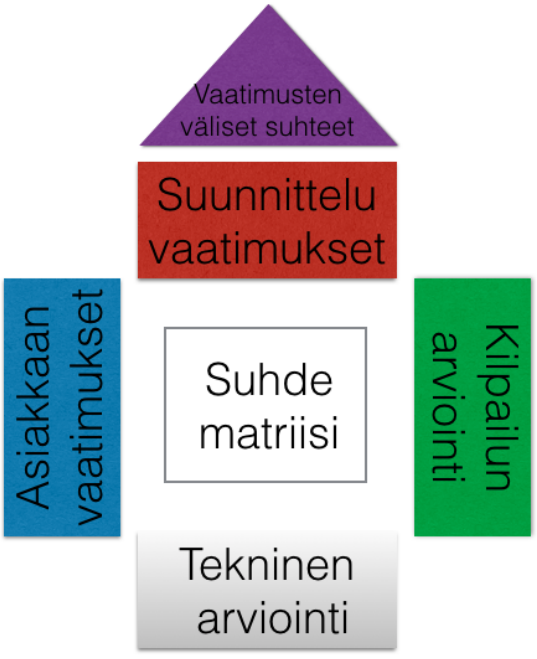
Ylin käyrä heijastaa niitä ominaisuuksia, jotka ilahduttavat ja innostavat asiakkaita. Ne voivat olla asiakkaan lausumattomia tai odottamattomia tarpeita, jotka johtavat korkeaan asiakastyytyväisyyteen (Raffo ym. 2010.).

Toinen työkalu nimeltä Quality Function Deployment (QFD) on työkalu asiakkaan vaatimusten kääntämiseksi tuotteen ominaisuuksiksi tuotteen myöhemmissä vaiheissa kehitysprosessia. Työkalulla voidaan asettaa

ominaisuudet tärkeysjärjestykseen, joka mahdollistaa tuotteen tärkeimpien ominaisuuksien määrittämisen jo alkuvaiheessa ennen kuin tuotteen kehitykseen on tehty suuria investointeja. Lisäksi sen avulla on mahdollista dokumentoida asiakkaan oikeat tarpeet sen sijaan, että tehdään päätöksiä muiden ei tuotteen käyttäjien mielipiteisiin perustuen. QFD vähentää tuotekehityksen kustannuksia, kun tietoa saadaan asiakkaalta. Lisäksi se vähentää tuotekehitykseen kuluvaa aikaa ja markkinoille saadaan nopeammin uusia tuotteita markkinoiden paremman ymmärryksen ja ymmärtämällä asiakkaan oikeat tarpeet. QFD mahdollistaa paremman asiakastyytyväisyyden parempien tuotteiden, pienempien kustannusten ja nopeamman reagointikyvyn kautta. Lisäksi se vähentää tarvittavien muutosten määrää tuotteen elinkaarella (Raffo ym. 2010.).

QFD työkalua käytetään Raffo ym. (2010) mukaan muodostamalla House of Quality. Tämä toimii työkaluna tuotekehityksessä ja tehostaa tuotekehitystä seuraavilla tavoilla:

1. Mahdollistaa tuotteiden keskeisten ominaisuuksien varhaisen määrittämisen asiakasvaatimusten perusteella ennen kuin tuotteeseen ja tuotantoprosessiin tehdään huomattavia investointeja
2. Mahdollistaa dokumentoida asiakkaan tarpeet sen sijaan, että tehdään päätöksiä muiden kuin asiakkaan mielipiteen perusteella
3. Vähentää tuotekehityksen kustannuksia, kun asiakas saadaan osallistumaan tuotekehitykseen
4. Nopeuttaa uusien tuotteiden markkinoille saattamiseen kuluvaa aikaa, kun ymmärretään paremmin markkinoita ja asiakasta
5. Mahdollistaa paremman asiakastyytyväisyyden parantamalla tuotteita, alentamalla kustannuksia ja nopeammalla reagointiajalla
6. Vähentää tuotteelle tehtävien muutosten määrää tuotteen elinkaarella



KUVIO 6 House of quality

1. Asiakkaan vaatimukset
2. Kilpailun arviointi
3. Suunnitteluvaatimukset
4. Vaatimusten väliset suhteet
5. Suhdematriisi
6. Tekninen arviointi

Asiakkaan vaatimukset: Talon rakennus alkaa Raffon ym. (2010) mukaan tunnistamalla ja kirjaamalla asiakkaan vaatimukset. Vaatimusten tärkeys pisteytetään arvoilla 1-9.

Kilpailun arviointi: Raffon ym. (2010) mukaan on tarkasteltavana olevan tuotteen kilpailukyvyn vertaamista kilpailijoiden vastaavien tuotteiden kanssa. Arviointiasteikkoa käyttämällä luodaan matriisi osoittamaan, kuinka tarkasteltavana oleva tuote ja kilpailevat tuotteet täyttävät vaaditut asiakkaan tarpeet.

Suunnitteluvaatimukset ovat Raffo ym. (2010) mukaan luettelo suunnitteluvaatimuksista, jotka on sisällytettävä tuotteeseen asiakkaiden tarpeiden tyydyttämiseksi.

Vaatimusten väliset suhteet: Raffon ym. (2010) mukaan on analyysi, jossa suunnitteluvaatimusten välisten suhteiden vaikutukset tunnistetaan ja ymmärretään. Jos suunnitteluvaatimus täyttyy, se voi asettaa jonkin muun suunnitteluvaatimuksen vastaiseksi. ”Positiivista merkkiä” käytetään osoittamaan suoraan suhteellinen suhde ja ”negatiivista merkkiä” käytetään kääntämään suhteellisesti suhteellinen suhde.

Suhdematriisi on Raffo ym. (2010) mukaan tunnistettujen asiakkaan tarpeiden ja suunnitteluvaatimusten välinen suhde. Tässä muodostetaan luokitusasteikkoon perustuva arviointi tuotteen ominaisuuksista ja asiakkaan tarpeista ja tunnistetaan, kuinka vahvasti vaatimus tyydyttää asiakkaan tarpeen.

Tekninen arviointi on Raffo ym. (2010) mukaan arviointi kuinka monimutkainen on tekniikka, jota on tarkoitus käyttää vaatimusten toteuttamiseksi. Vaatimukset asetetaan arviointiasteikolle, jolla luodaan tekninen arviointimatriisi. Tämän avulla tehdään vertailu siitä, missä analysoitava tuote ja paras kilpaileva tuote edustavat suunnitteluvaatimuksia, jotka on täytettävä.

4.4 Hukan hallinnan vaikutukset ohjelmistojen tuotekehitykseen

Mikäli ohjelmistojen tuotekehitystä tekevä yritys pystyy tunnistamaan hukan ja eliminoimaan sen ja sen riskit, on yrityksen ohjelmistojen tuotekehitys tehokkaampaa (Poppendieck ja Poppendieck 2006). Tämän takia hukan

tutkiminen puhtaasti ohjelmistojen tuotekehityksen näkökulmasta on tärkeää tieteellisesti sekä käytännöllisesti. Tässä seminaari raportissa tuloksiksi saatiin, että hukat on mahdollista jakaa eri ohjelmistojen kehityksen tasoille. Jotkin hukat ovat olemassa useammalla kuin yhdellä ohjelmistojen tuotekehityksen tasolla. Kirjallisuuden mukaan jokin prosessi voi olla myös hukkaa yhdellä tasolla, kun toisella se kasvattaa tuotekehityksen tehokkuutta.

Hukan tunnistamiseen ja eliminointiin on useita eri tieteellisiä menetelmiä, jotka tarjoavat tähän erilaisia työkaluja. Menetelmistä esitetty Lean on enemmän ajattelutapa, jonka omaksumalla ja periaatteita noudattaen yritys voi tunnistaa ja eliminoida hukkaa tuotekehityksessä (Mujtaba ym. 2010.) Lisäksi Lean tarjoaa kaksi erilaista työkalua, joilla yritys voi tarkastella visuaalisesti tuotekehitysprosessia ja siinä esiintyvää hukkaa. Statistical Process Control visualisoi joukon yksittäisiä vastaavia tuotekehitysprosesseja ja esittää mitkä ovat menneet suunnitellusti ja mitkä eivät (Womack & Jones, 1990). Value Stream Mapping visualisoi koko tuotekehitysprosessin ja siihen sisältyvät arvoa tuottavat ja ei arvoa tuottavat tekijät (Mujtaba ym. 2010.).

Toinen työn visualisointiin perustuva menetelmä ja työkalu nimeltä Kanban. Sen keskiössä on taulu, jonka avulla pystytään visuaalisesti hahmoittamaan nykyiset ja tulevat tehtävät sekä niihin kuluva aika. Kanban tarjoaa yritykselle keinon tunnistaa ja eliminoida hukkaa yrityksen tuotekehityksessä visualisoimalla työn eri vaiheet ja tätä kautta ymmärtää tuotekehityksen etenemisen paremmin (Kniberg & Skarin, 2010; Ahmad ym. 2013.).

Kolmas asiakaskeskeinen menetelmä nimeltä Six-Sigma ja siitä johdettu Lean Six-Sigma muodostuu kahdesta yrityksen omaksumasta tekijästä. Ensimmäinen on ymmärtää käyttää niin vähän resursseja kuin mahdollista tarvittavan arvon tuottamiseksi asiakkaalle. Toinen on tunnistaa ja ymmärtää, mitkä prosessit oikeasti lisäävät tuotteen arvoa asiakkaalle. Lean Six-Sigma tarjoaa kaksi eri työkalua, Kano-analyysin asiakkaiden tarpeiden ymmärtämiseksi ja näiden priorisoimiseksi. Toinen työkalu nimeltään Quality Function Deployment on työkalu asiakkaan vaatimusten kääntämiseksi tuotteen ominaisuuksiksi tuotteen myöhemmissä vaiheissa kehitysprosessia (Raffo ym. 2000.).

5 TUTKIMUKSEN TOTEUTUS

Tässä luvussa esitellään, miten tutkimus toteutettiin, mitä vaiheita tutkimus sisälsi sekä perustellaan valitut menetelmät. Ennen menetelmien esittelyä kerrataan tutkimuksen tarkoitus ja tavoite sekä tutkimuskysymys. Lisäksi esitellään myös apututkimuskysymykset, jotka auttavat vastaamaan tutkimuskysymykseen. Tutkimuskysymykseen vastaamiseksi esitellään valitut tutkimusmenetelmät ja miten itse tutkimus toteutettiin sekä perustellaan, miksi tutkijat päätyivät valittuihin menetelmiin. Tämän jälkeen esitellään ja perustellaan valittu tiedonkeruu menetelmä. Lopuksi kuvataan tutkimustulosten analysointimenetelmä ja miten johtopäätökset muodostettiin. Viimeisenä arvioidaan tutkimuksen luotettavuutta.

Tutkimuksen tarkoituksena on selvittää mitä hukalla tarkoitetaan ohjelmistojen tuotekehityksessä, mistä se aiheutuu ja miten sitä voidaan vähentää. Tutkimuksessa hukka on jaettu tuotekehityksen kolmelle eri tasolle, portfoliotaso, tuotetaso ja tuotekehitystaso. Tutkimuksen tutkimuskysymys on: miten hukkaa voidaan vähentää portfoliotasolla, tuotetasolla ja tuotekehitystasolla. Tutkimuskysymykseen vastaaminen edellyttää, että ymmärrämme mitä on hukka, mitä tarkoitetaan ohjelmistojen tuotekehityksessä portfoliotasolla, tuotetasolla ja tuotekehitystasolla sekä mitä hukka on portfoliotasolla, tuotetasolla ja tuotekehitystasolla ja mikä aiheuttaa hukkaa jokaisella tasolla sekä mitä siitä seuraa. Tutkimusongelmaksi muodostui seuraava

- Miten voidaan tunnistaa ja minimoida hukkaa eri ohjelmistojen tuotekehityksen tasoilla.

Tutkimusongelman ratkaisemiseksi määrittyi seuraavat apukysymykset:

1. Mitä on portfolionhallinta
2. Mitä on tuotehallinta
3. Mitä on tuotekehitys
4. Mitä on hukka ohjelmistojen tuotekehityksen eri tasoilla
5. Mitä hukkaa eri ohjelmistojen tuotekehityksen tasoilla esiintyy ja syntyy
6. Mitä hukasta aiheutuu
7. Miten hukkaa eri ohjelmistojen tuotekehityksen tasoilla pyritään tunnistamaan
8. Miten hukkaa ja sen riskiä hallitaan ja minimoidaan eri ohjelmistojen tuotekehityksen tasoilla
    a. Mitä operatiivisia menetelmiä käytetään
    b. Mitä johtamismenetelmiä käytetään

5.1 Systemaattinen kirjallisuuskatsaus

Systemaattinen kirjallisuuskatsaus tehtiin perustuen Okolin ja Schabramin (2010) systemaattisen kirjallisuuskatsauksen menetelmään. Kirjallisuuskatsauksen tarkoitus on luoda tieteellinen teoreettinen perusta tutkimukselle. Kirjallisuuskatsauksessa käytiin läpi suuri määrä aiheeseen liittyviä artikkeleita ja teoksia, jotka löydettiin kirjallisuuskatsauksen protokollassa määriteltyjen hakusanojen mukaan. Löydetyt artikkelit valittiin protokollassa määritellyillä valinta perusteilla ja niistä tehtiin analyysi. Protokollalla tarkoitetaan suunnitelmaa, jonka mukaan systemaattinen kirjallisuuskatsaus tehdään. Protokollan dokumentointi on tärkeää, sen perusteella toteutetaan artikkeleiden haku ja valinta. (Okoli & Schabram, 2010).

Protokolla testattiin ja validoitiin ennen kirjallisuuskatsauksen toteuttamista. Protokolla koulutettiin molemmille kirjallisuuskatsauksen tekijöille ja ensimmäinen hakuvaihe toteutettiin omatoimisesti, jonka jälkeen omista löydöksistä tehtiin valinnat analyysivaiheeseen. Tutkimuksessa tehtävän kirjallisuuskatsauksen protokolla on esitetty alla.

Hakusanat, joilla etsittiin kirjallisuutta tietokannoista ovat: waste, software development, software product development, portfolio management, software product, product management, information system development, information system product, identification, information technology, elimination, management, method, software engineering ja näiden yhdistelmiä. E-kirjastot, joita käytetään kirjallisuuden etsimiseen ovat: AIS Electronic Library, ACM, IEEE. Kirjastot valittiin niiden alustavan tarkastelun perusteella

- Rajaukset, joilla artikkelit valittiin:
    - Etsimme noin 30 lähdettä kysymykseen vastaamiseksi, joista valittiin sopivimmat
        - Sopivimmat lähteet valittiin seuraavin perustein:
            - lähde käsittelee hukkaa johtamismenetelmien näkökulmasta
            - lähde käsittelee hukkaa ohjelmistojen kehityksessä
            - lähde julkaistu vuoden 2000 jälkeen
                - pl. tieteenalan klassikkoteokset
            - mielellään jonkin verran viittauksia
            - kieli on englanti tai suomi
        - Mikäli 30 lähdettä ei riitä, etsimme lisää
            - Lähteiden määrän tuli kattaa tutkittava ilmiö kokonaisvaltaisesti
- Hakeminen tapahtui:
    - Etsimällä jokaisesta käytettävästä tietokannasta aineistoa esitellyillä hakusanoilla
    - Valitsemalla hakutuloksista aiheeseen sopivat artikkelit ja teokset
    - Mikäli artikkeli julkaistu jossain yllämainitussa julkaisussa ja artikkelin kuvaus osuu aiheeseen, tutustutaan siihen tarkemmin

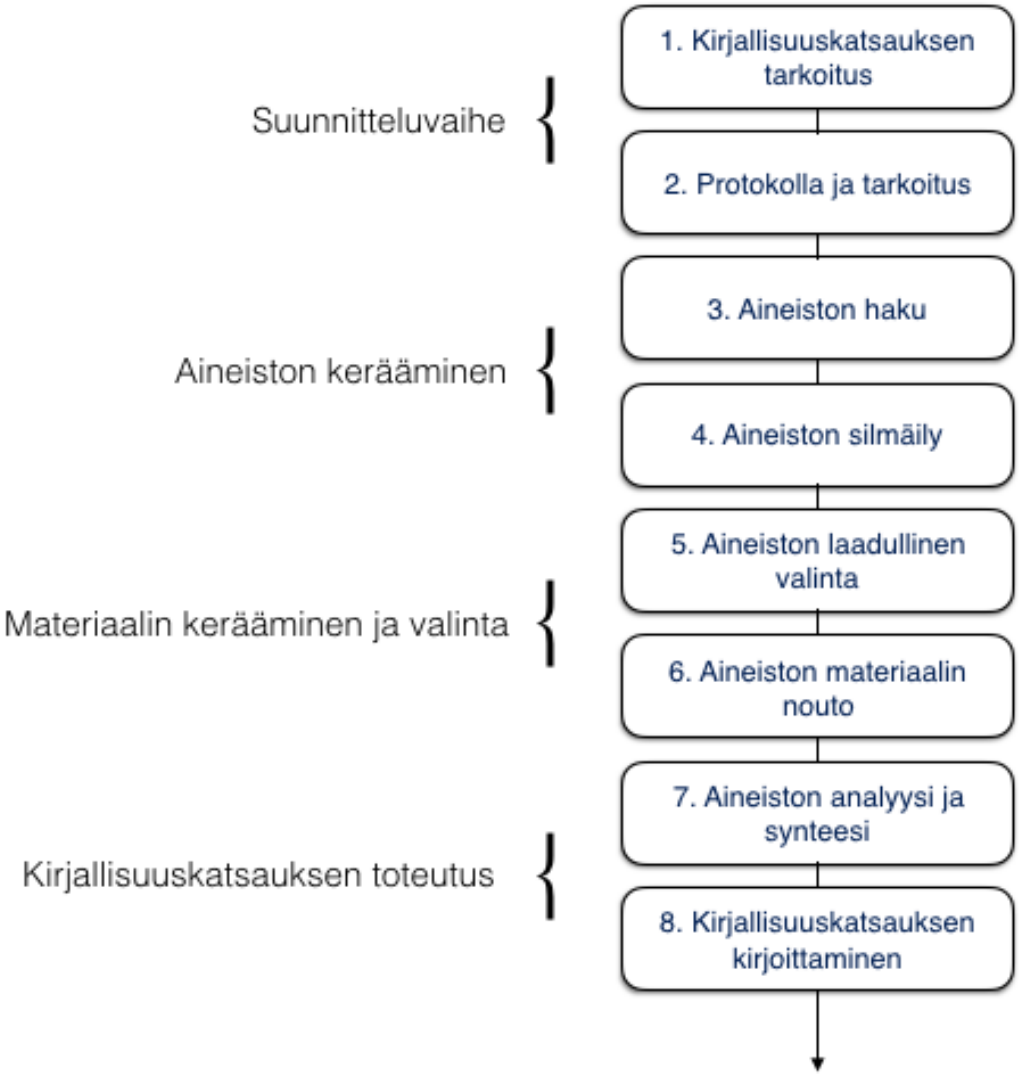
- Mikäli teosta pidetään merkittävänä aiheen tieteen alalla, ja teoksen kuvaus osuu aiheeseen, tutustutaan siihen tarkemmin
- Molemmat kirjoittajat hakevat aineistoa yllä mainituilla kriteereillä ja valitsevat relevantit artikkelit ja teokset
    - Ennen aineistoon tutustumista kirjoittajat vertaavat valitsemiaan aineistoja ja yhdessä päättävät, joihin tutustutaan tarkemmin ja käytetään lähteinä
    - Valitut aineistot jaetaan kahtia aiheen mukaan ja tämän perusteella syntyy myös työnjako kirjallisuuskatsauksen kirjoittamiselle

Seuraavassa vaiheessa tehtiin aineiston haku protokollassa määritellyin hakusanoin. Okolin ja Schabramin (2010) systemaattisen kirjallisuuskatsauksen aineiston haun kirjaamisessa tulee käyttää taulukkoa tai tietokantaa, johon kerätään kaikki hakuvaiheessa löydetty aineisto. Jokaisella hakusanalla tai hakusanayhdistelmällä pyrittiin löytämään 30 tieteellistä artikkelia.
Neljännessä vaiheessa taulukkoon kerättävä tietokanta jaetaan kirjoittajille ja ne käytiin silmämääräisesti läpi otsikko ja johdantotasolla. On hyvin todennäköistä, että hakusanoilla ilmenneet artikkelit eivät kaikki sovi tutkimukseen, ja ne karsitaan pois taulukosta. (Okoli & Schabram, 2010.)

Viidennessä vaiheessa käytiin läpi kaikki jäljelle jäänyt aineisto tarkasti. Aineisto valittiin, mikäli se tuo arvoa kirjallisuuskatsaukselle. Mikäli artikkeli ei liittynyt hukkaan tai ohjelmistojen tuotekehitykseen tavalla, joka ei ole hyödyllistä, jätettiin se pois tutkimuksesta.

Aineisto, joka ei ylittänyt pisterajaa, poistettiin taulukosta ja tietokannasta. Kuudennessa vaiheessa taulukkoon jäljelle jääneistä aineistoista kerättiin kirjallisuuskatsaukseen relevantin materiaalin nouto. Siinä jokainen tutkimukselle relevantti kohta lisättiin taulukkoon lainauksena. Jokainen aineisto käytiin yksitellen läpi kirjoittajien toimesta. Lainausten tulee vastata tutkimuskysymykseen, jotta materiaali on relevanttia. (Okoli & Schabram, 2010.)

Seitsemännessä vaiheessa tehtiin synteesi materiaalista, joka jäi jäljelle materiaalin noutovaiheesta. Jokainen lainaus värikoodattiin niiden yhteneväisyyden mukaan ja niistä tehtiin synteesi kirjallisuuskatsaukseen. Materiaalista kirjoitettiin kokonaisvaltainen selkeä kirjallisuuskatsaus, josta käy ilmi kirjallisuuden synteesi tutkimuksen aihealueeseen. Näiden lisäksi synteesi tehtiin olemassa olevista tutkimuksista ja niitä tarkasteltiin niiden yhteneväisyyksien mukaan.
Kahdeksannessa vaiheessa synteesin perusteella kirjoitettiin kirjallisuuskatsaus. Kirjallisuuskatsauksen osa-alueet jaettiinn kirjoittajille. Kirjoittamisen jälkeen toinen kirjoittaja arvioi ja kommentoi kirjoitettua osuutta, ja tarvittaessa siihen tehtiin muutoksia. (Okoli & Schabram, 2010.).



KUVIO 7 Systemaattisen kirjallisuuskatsauksen protokolla (Okoli ja Schabram, 2010.)

5.2 Tutkimusmalli

Empiirisen tutkimuksen tutkimusmenetelmä on kvalitatiivinen tutkimus. Tämä tutkimusmenetelmä valittiin sen vuoksi, koska tutkimuksen tarkoitus on ymmärtää tiettyä ilmiötä. Laadullisessa tutkimuksessa pyritään ymmärtämään ja kuvaamaan tutkittavaa ilmiötä (Hirsjärvi, Remes & Sajavaara, 2009). Tässä tutkimuksessa ilmiö on hukka ohjelmistojen tuotekehityksen kolmella eri tasolla. Tutkimuksen tulokset ovat perusteltuja tutkimuksen kontekstissa. Tutkimusongelman ratkaisemiseksi kvalitatiivinen sopii hyvin, koska tässä tutkimuksessa ei esitetä hypoteeseja tai esioletuksia perustuen muihin aikaisemmin toteutettuihin tutkimuksiin tai teorioihin.

Tutkimus toteutettiin usealla kvalitatiivisella tapaustutkimuksella. Tämä menetelmä valittiin sen perusteella, koska tutkimuksen tarkoitus on tutkia ilmiötä oikeassa ympäristössä. Valitulla menetelmällä ymmärretään paremmin kehittävien tuotteiden ja niiden valmistajien välisiä vuorovaikutuksia.

Tutkimuksen tapaus on ymmärtää ilmiötä/ ilmiöitä, jotka aiheuttavat hukkaa ohjelmistojen tuotekehityksessä. Tutkimuksen tavoite on saada läpileikkaava ymmärrys ilmiöstä ja tämän vuoksi valitsimme tutkimuksen toteutettavaksi usealla tapaustutkimuksella. Myersin ja Newmanin (2007) mukaan kvalitatiivinen tutkimusote on parempi tutkimustapa silloin, jos kirjallisuudessa ei ole esitetty aiemmin selkeää teoriamallia, jota voi mitata kvantitatiivisella tutkimuksella.

Kanasen (2008) mukaan kvalitatiivisen tutkimuksen kautta pyritään luomaan yleistyksiä ja päätelmiä kerätyn aineiston kautta. Määrällisessä tutkimuksessa ei tehdä päätelmiä, vaan tutkimustulokset perustuvat mitattavien asioiden tulosten analysointiin ja tilastollisiin tuloksiin (Kananen, 2008). Tässä tutkimuksessa pyrittiin kuvaamaan tutkittavia asioita ja ilmiöitä kokonaisvaltaisesti, jotta niistä saatiin mahdollisimman todenmukainen kuvaus. Kvalitatiivinen tutkimus perustuu luonnollisessa ympäristössä tehtyyn laadulliseen analyysiin, jossa informaatio on kerätty havainnoitavien ihmisten kertomista tai kirjoitetuista sanoista (Denzin & Lincoln, 2011). Tällöin kvalitatiivinen tutkimusote oli oikea lähestymistapa tutkimusta kohtaan, koska tutkimuksessa ei ollut tarkoitus mitata kvantitatiivisilla tutkimusmenetelmillä mitattavaa dataa, vaan kuvata tutkimuksen ilmiötä kokonaisvaltaisesti tutkimusongelman näkökulmasta (Myers & Newman, 2007.)

Kvalitatiivisen tutkimuksen tiedonkeruumenetelmiä ovat esimerkiksi haastattelut, havainnointi, dokumentit ja kyselyt (Hirsjärvi, ym. 2009). Tämän tutkimuksen tiedonkeruumenetelmäksi valittiin haastattelut. Vaihtoehtoinen tiedonkeruumenetelmä olisi voinut olla havainnointi, mutta tämän tutkimuksen kontekstissa se ei ollut mahdollinen, koska kohdeyritysten tuotteen toimiala, toimintaympäristöt ja työskentelytavat olivat erilaisia.

Haastatteluissa haastateltiin kolmesta eri yrityksestä kolmella eri ohjelmistojen tuotekehityksen tasolla työskenteleviä henkilöitä. Henkilöiden tittelit ja tehtävät olivat esimerkiksi tuotekehitysjohtaja, tuotepäällikkö ja ohjelmistokehittäjä. Haastateltaviksi valittiin henkilöitä, jotka liittyivät vahvasti tutkittavaan ilmiöön. Seuraavaksi kerromme tarkemmin eri haastattelumuodoista, jonka tarkoitus on perustella valittua haastattelumenetelmää.

Henkilöitä haastateltiin yksitellen. Ryhmähaastattelu ei ollut mahdollinen, koska yritysten toimialat ja työskentelymenetelmät olivat erilaisia toisiinsa verrattuna. Sen lisäksi ryhmähaastatteluiden järjestäminen olisi ollut haastavaa, koska yritykset sijaitsivat eri paikkakunnilla.

Strukturoitu haastattelu on haastattelun kannalta tiukin. Siinä haastattelukysymykset on asetettu etukäteen ja haastattelun sanamuotoja ei voi muuttaa. Lisäksi Strukturoidussa haastattelussa ei voi vaihtaa esimerkiksi kysymysten paikkaa kesken haastattelun. Strukturoitu haastattelu vastaa menetelmänä lomakekyselyä (Eskola & Suoranta, 2000.) Strukturoitu haastattelu olisi ollut liian jäykkä, eikä sen aikana olisi mahdollisesti päästy pureutumaan nouseviin asioihin tarpeeksi, jotta ilmiön selittäminen olisi ollut mahdollista.

Puolistrukturoitu haastattelu on strukturoitua haastattelua vapaampi haastattelumuoto. Esimerkiksi kysymysten järjestelyä, sanamuotoja ja rakennetta voidaan vaihtaa haastattelusta toiseen. Yleensä puolistrukturoidussa

haastattelussa haastateltava on miettinyt ennakkoon aihealuetta ennen haastattelua. Hirsijärven ja Hurmeen (2001) mukaan Puolistrukturoidulle haastattelulle ei ole tarkkaa teoreettista muotoa, vaan haastattelu voi tapauksesta tai tilanteesta johtuen olla hyvinkin erilainen. Kuitenkin haastattelun tarkoitus on selvittää haastateltavien kokemuksia, näkemyksiä ja mielipiteitä tutkittavasta ilmiöstä. Puolistrukturoitu haastattelu kulkee ennalta määriteltyjen teemojen ja suuripiirteisesti asetettujen kysymysten asettamissa rajoissa. (Hirsjärvi, ym. 2009; Hirsjärvi & Hurme, 2001)

Teemahaastattelu on keskustelumuotoinen tilanne, jossa haastattelu ei etene ennalta määriteltyjen kysymysten kautta vaan haastattelun ennalta määriteltyjen teemojen mukaan. Teemojen ja niihin liittyvien kysymysten järjestys haastattelussa on vapaa, eikä haastatteluissa välttämättä keskitytä tiettyihin teemoihin samalla laajuudella riippuen haastattelusta. Teemahaastattelu antaa mahdollisuuden haastateltavalle tulkita kysymyksiä ja teemoja mahdollisimman luontevasti oman osaamisen ja tietämyksen mukaan. (Hirsjärvi, ym. 2009; Hirsjärvi & Hurme, 2001)

Haastattelumuodoksi valittiin puolistrukturoidun ja teemahaastattelun yhdistelmä. Kirjallisuuden mukaan teemahaastattelu voidaan nähdä joissain tapauksissa puolistrukturoituna haastatteluna, koska ne ovat keskenään niin samankaltaisia. Puhuttaessa toisen haastattelumuodon elementeistä, voidaan ne nähdä myös toisen haastattelun elementteinä. (Hirsjärvi, ym. 2009) Tämän tutkielman haastattelumuodon tekee puolistrukturoiduksi se, että haastattelukysymykset olivat ennakkoon määritellyt, mutta niiden sanamuotoja ja järjestystä voitiin muuttaa haastattelukohtaisesti. Haastattelun teemat pohjautuivat tutkimuksen kirjallisuuskatsauksen teemoittaisiin aihealueisiin, joihin pyrittiin saamaan haastattelussa vastauksia mahdollisesti tarkentavien kysymysten kautta. Kaikkia teemojen tarkentavia kysymyksiä ei käyty tarvittaessa läpi jokaisessa haastattelussa, mikäli teemaan saatiin täsmällinen ja tarkka vastaus aikaisempien kysymysten avulla. Teemahaastattelua pidetään luotettavana tutkimusmenetelmänä, koska sen avulla pystytään havainnoimaan syvällisesti miksi ja miten haastateltava ilmiö ilmenee haastateltavan mielestä (Myers & Newman, 2007).

5.2.1 **Haastatteluiden teemat**

Haastatteluissa käytettiin luvun 5.2 tutkimusmallia. Haastatteluissa oli eri kysymykset portfolio-, tuote- ja ohjelmistojen tuotekehitystason henkilöille. Jokainen haastattelu oli kuitenkin jaettu samankaltaisiin teemoihin: 1a) portfoliotason menetelmät, 1b) tuotekehitystason menetelmät 1c) ohjelmistojen tuotekehitystason menetelmät 2) hukka käsitteenä, 3) hukan tunnistaminen 4) hukan riskin hallinta 5) hukan minimointi 6) hukka muilla kuin omalla tasolla.

Haastattelukysymykset luotiin tutkimusmallissa tunnistettujen teemojen perusteella. Kysymysten tarkoitus oli saada vastaukset tutkimusongelmaan ja tutkimuskysymyksiin. Haastattelun teemojen aihealueisiin luotiin tukikysymyksiä, joiden tarkoitus oli auttaa haastattelijaa havainnoimaan ja tunnistamaan

kokonaisvaltaisesti haastateltavan näkemystä ja mielipiteitä havainnoitavasta ilmiöstä. Tukikysymyksiä ei tarvittu, mikäli haastattelun aikana kysymykseen oli vastattu tarpeeksi tarkalla tasolla. Teemoittaisesti jaetun haastattelun tarkoitus oli selvittää jokaista teemaa koskeva ilmiö mahdollisimman tarkalla tasolla. Tutkimuksen tuloksissa on tarkoitus vastata siihen, miten tutkimuksesta ilmenneet asiat vertaantuvat kirjallisuuskatsauksessa esiintyneisiin vastaavien teemojen teorioihin.

5.3 Tiedonkeruu ja otanta

Tutkimuksen otantamenetelmänä käytettiin harkinnanvaraista otantaa. Tutkimuksessa haastateltavien henkilöiden valinta tehtiin tutkijoiden asettamien ja toimeksiantajayrityksen kriteerien perusteella. Tutkittavat yritykset rajattiin 10-100 hengen ohjelmistojen tuotekehitystä tekeviin yrityksiin, jossa tehdään ohjelmistojen tuotekehitystä sekä yrityksessä käytetään portfolion hallintaa, tuotehallintaa ja tehdään ohjelmistojen tuotekehitystä. Kvalitatiivisessa tutkimuksessa on perusteltua rajata joukko, jolta tiedonkeruu tehdään. Kvalitatiivisessa tutkimuksessa ei tarvitse tutkia koko joukkoa, tai mahdollisimman laajaa otantaa tutkimusjoukosta. Kokonaistutkimus olisi tarkoittanut, että ensin olisi kartoitettu kaikki suomalaiset ohjelmistojen tuotekehitystä tekevät yritykset, joissa on 10-100 henkeä ja sen jälkeen alustavasti tarkasteltu, missä kaikissa yrityksissä hallitaan ohjelmistojen tuotekehitystä portfolio-, tuote- ja ohjelmistojen tuotekehitystasolla. Tämän jälkeen jokaisesta yrityksestä olisi haastateltu kolmea henkeä. Otanta olisi ollut liian suuri tämän tutkielman rajoissa. Lisäksi voitiin todeta, että haastattelut harkinnanvaraisella otannassa perustuen toimeksiantajan suosituksiin olivat riittävät ilmiön ymmärtämiseen ja havainnointiin (Saaranen-Kauppinen & Puusniekka 2006).

Haastatteluun osallistui 9 haastateltavaa kolmesta eri suomalaisesta ohjelmistojen tuotekehitystä tekevästä yrityksestä. Haastateltavat henkilöt haluttiin sellaisista yrityksistä, joista toimeksiantajalla oli kokemusta ja jokainen haastateltava yritys teki ohjelmistojen tuotekehityksen hallintaa kolmella tutkimuksessa käsiteltävällä tasolla. Tavoitteena oli, että haastateltava yritys tekee aktiivisesti oman ohjelmiston tuotekehitystä, eikä esimerkiksi projekteina ohjelmistoja ulkopuolisille yrityksille. Näin voitiin varmistua, että haastateltava yritys tekee portfoliohallintaa, tuotehallintaa ja ohjelmistojen tuotekehitystä aina samalle tuotteelle, eikä menetelmät vaihdu asiakkaittain.

Haastattelukutsuja lähetettiin yhteensä kuudelle yritykselle. Muut yritykset eivät voineet osallistua haastatteluihin. Haastatteluihin kutsuttiin kutsukirjeellä, joka on nähtävissä Liitteessä 1. Haastateltavia ei pyydetty valmistautumaan erityisillä tavoilla, vaan ainoastaan miettimään haastattelun teemaa ennen haastattelua. Portfoliotason haastatteluun kutsuttiin tuotekehitysjohtaja tai muu henkilö, joka vastaa koko tuotekehityksestä ja sen portfoliosta. Nimikkeitä haastatteluissa oli Software Development Director, Digital Transformation lead ja Business Development Director. Tämän tason

henkilöt pystyivät vastaamaan portfoliotason haastattelukysymyksiin ja vastausten perusteella voitiin havainnoida ja ymmärtää portfoliotasolla tapahtuvia ilmiöitä. Tuotetason haastatteluun kutsuttiin tuotteesta vastaavia henkilöitä. Nimikkeitä haastateltavissa oli Tuoteomistaja, Product Owner ja IT-Specialist. Tuotetason haastatteluissa pyrittiin selvittämään tuotetasolla tapahtuvia ilmiöitä hukkaan liittyen. Haastattelun määrittelyiden perusteella tuoteomistajat pystyivät vastaamaan näihin haastattelukysymyksiin parhaiten ilmiön ymmärtämisen kannalta. Ohjelmistotason haastatteluihin kutsuttiin ohjelmistojen tuotekehittäjiä. Nimikkeitä haastatteluissa oli Senior Software Developer, Test Specialist ja Software Architecht. Ohjelmistojen tuotekehitystason haastatteluissa tavoitteena oli ymmärtää ohjelmistojen tuotekehitystason ilmiöitä hukasta. Määrittelyiden ja toimeksiantajan suositusten perusteella ohjelmistojen tuotekehitystä tekevä henkilö oli oikea haastateltava.

Haastatteluihin kutsuttavien henkilöiden yhteystiedot saatiin toimeksiantajalta tai selvittämällä yhteystiedot yrityksen ohjelmistojen tuotekehityksen esimieheltä. Potentiaaliset haastateltavat henkilöt määritti ohjelmistojen tuotekehityksen esimies. Kutsussa kerrottiin, mistä haastateltavilla henkilöillä tulee olla tietämystä, jotta he voivat vastata haastattelukysymyksiin. Haastateltavia pyrittiin motivoimaan esittämällä, että toimitettavan tutkimuksen perusteella yritys voi reagoida heillä jo nyt aiheutuvaan hukkaan sekä nähdä mitä hukkaa muissa heidän kokoluokan ohjelmistojen tuotekehitystä tekevissä yrityksissä syntyy ja miten niitä hallitaan.

Haastattelut toteutettiin joko paikan päällä tai Skype for Business verkkotapaamisohjelmiston kautta. Kaikki haastattelut nauhoitettiin. Lisäksi haastattelija teki muistiinpanoja haastattelun aikana tutkimukselle tärkeimmistä kohdista. Haastattelut litteroitiin tekstitiedostoiksi. Litteroinnin tarkoituksena oli löytää jokaisesta haastattelusta muissa haastatteluissa toistuvat asiasanat ja fraasit. Haastatteluiden pituus vaihteli 1h – 1,5h välillä.

Haastatteluiden jälkeen aineisto tuotiin yhteen. Tärkeimpiä haastatteluiden tuloksia litteroinnin ja muistiinpanojen perusteella verrattiin toisiinsa ja niistä tehtiin havaintoja. Haastatteluista voitiin havaita eri hukkiin ja niihin liittyviä teemoja. Lisäksi voitiin havaita hukan esiintymiseen ja hallintaan liittyviä teemoja. Lisäksi voitiin havaita eri tasojen erot ohjelmistojen tuotekehityksen hukkaan liittyen. Tässä vaiheessa voitiin todeta, että lisää haastatteluja ei tarvitse tehdä, koska haastatteluissa selkeästi toistui samat teemat ja asiat ilmiöön liittyen. Tällöin voitiin lopettaa uuden aineiston kerääminen. Kun aineistossa alkaa toistua samat tulokset, sitä kutsutaan aineiston kyllääntymiseksi tai saturaatioksi (Hirsjärvi, ym. 2009).

5.4 Aineiston analyysi

Aineiston analysoinnin tarkoitus on puhdistaa ja järjestää aineisto niin, että löydökset tulevat havaittaviksi. Niin, että aineistosta voi tehdä päätelmiä ja tutkimuksen merkitys tulee esiin (Hirsjärvi, ym. 2009). Aineisto analysoitiin

järjestämällä haastattelut ja vastaukset loogisiin ryhmiin teemoittain. Tämän jälkeen molemmat kirjoittajat koodasivat vastaukset ja poimivat sieltä merkittävimmät löydökset tutkimuksen kannalta. Molempien kirjoittajien löydöksiä verrattiin toisiinsa ja tätä kautta varmistettiin, että haastatteluista on löydetty kaikki relevantti informaatio, josta voi tehdä johtopäätöksiä. Saaranen-Kauppisen ja Puusniekan (2006) mukaan teemoittelu on hyvä analyysimenetelmä teemahaastatteluihin perustuvassa tutkimuksessa. Aineisto järjestellään teemoihin ja alateemoihin perustuen niiden esiintyvyyteen haastatteluissa. Teemojen muodostamiseen voidaan käyttää koodaystekniikoita tai teemojen kvantifiointia eli määrällistä tutkimista. (Saaranen-Kauppinen & Puusniekka, 2006.) Teemoittelu toteutettiin tässä tutkimuksessa käyttäen haastattelujen taulukointia ja taulukkolaskentaohjelmiston eri luokitteluominaisuuksia.

Tutkimuksen kaikki haastattelut litteroitiin. Litteroitu aineisto vietiin Microsoft Excel taulukkolaskentaohjelmaan ja litteroinnista tehtiin tietokanta. Ensimmäiseen sarakkeeseen sijoitettiin haastatteluiden tutkimusteemat, joiden pohjalta haastattelukysymykset oli asetettu. Haastattelun teemoja oli yhteensä 6. Tämän jälkeen haastattelut lisättiin Excel-tiedostoon. Haastattelun lauseet jaettiin teemoittain eri teemoihin. Tämän jälkeen jokaisesta lauseesta etsittiin avainsanoja. Avainsanat jaettiin sarakkeisiin ja niiden kappalemäärät kirjattiin ylös teemoittain. Avainsanat valittiin perustuen kirjallisuuden avainsanoihin.

Avainsanojen mukaan haastatteluista pystyttiin luomaan taulukot hukan esiintyvyydestä eri ohjelmistojen tuotekehityksen tasoilla. Esiintyvyyttä verrattiin muihin tasoihin ja siitä saatiin kokonaiskuva hukasta ohjelmistojen tuotekehityksestä. Avainsanojen perusteella pystyttiin havainnoimaan hukan minimoimiseen ja hallintaan liittyvät seikat. Nämä avainsanat kirjattiin taulukkoon.

Viimeiseksi litteroinnista muodostettiin suorat lainaukset, joiden perusteella tutkimustuloksissa voitiin antaa esimerkkejä ja todisteita siitä, että kyseisen aineiston esittäminen on perusteltua tutkimuksen mukaan. Esimerkkien avulla pystytään myös ymmärtämään ja esittämään tärkeitä tutkimuslöydöksiä tutkimuksen perusteella. Lainauksia olivat esimerkiksi seuraavat:

> "Hukkaa on portfoliotasolla muun muassa, kun asioihin kiinnitetään liian varhain huomiota ja näin ollen niitä pyöritellään turhaan"

> "Portfoliotason prosessi pitäisi mallintaa, jotta tällä tasolla syntyvää hukkaa voitaisiin tunnistaa ja minimoida paremmin"

> "Tuotekehityksessä tuotekehityksessä työ tehostuisi, jos voisi keskittyä yhteen asiaan kerralla"

5.5 Tiedonkeräyksen luotettavuuden arviointi

Seuraavaksi arvioidaan, onko tiedonkeräys toteutettu luotettavasti. Koko tutkielman luotettavuutta ja yleistettävyyttä tarkastellaan luvussa 7.3 reliabiliteetin ja validiteetin avulla. Reliabiliteetti tarkoittaa tutkimuksen toistettavuutta ja

validiteetti kertoo, miten hyvin tutkimuksen avulla voidaan mitata sitä ilmiötä, jota tutkimuksella pyritään mittaamaan.

Tiedonkeräys ja haastattelujen otanta pyrittiin toteuttamaan niin, että ne ovat mahdollisimman toistettavissa ja niiden perusteella voidaan tehdä yleistyksiä. Haastattelurunko, teemat ja kysymykset rakennettiin aikaisempien tutkimusten perusteella. Avainsanat perustuivat kirjallisuudessa esiintyviin avainsanoihin ja aikaisempien tutkimusten havaintoihin (Hirsjärvi & Hurme, 2001).

Tutkimuksen yritykset valittiin toimeksiantajan kanssa suunnitellun luokittelun perusteella, koska haluttiin varmistaa, että kyseiset yritykset tekevät ohjelmistojen tuotekehitystä tietyillä toimeksiantajan tietämillä ja tutkimuksen vaatimilla menetelmillä. Lisäksi tällä tavalla voitiin varmistaa, että kohdejoukon yritykset olivat sellaisia, joilla oli tarpeeksi korkea substanssi ja osaaminen vastata haastattelukysymyksiin ja näin mahdollistaa ilmiön havainnointi. Otanta tutkimuksessa oli homogeeninen, mutta tutkimusongelma vaati, että yritykset ovat tietyn kaltaisia. Laadullinen tutkimus vaati, että haastattelujoukko oli tarkasti määritelty, jotta haastattelukysymyksiin vastaaminen oli mahdollista. Ohjelmistojen tuotekehitystä tekevien yritysten sisältä valikoitiin tietyillä nimikkeillä toimivia henkilöitä, jotka olivat sellaisia, joiden voitiin katsoa olevan kokonaisvaltainen näkemys yrityksen toimintoihin, jotka liittyivät havainnoitavaan ilmiöön.

Tutkimukselle oli myös tärkeää, että haastateltavat olivat kiinnostuneita osallistumaan tutkimukseen. Osallistumisesta ei tarjottu korvausta, vaan haastateltaville luvattiin ainoastaan lähettää lopullinen tutkimus, jota he voivat hyödyntää jatkossa organisaation sisällä hukan tunnistamiseen ja minimointiin. Liitteessä 1 esitellään haastattelukutsu, jonka perusteella haastateltavat kutsuttiin haastatteluun. Kutsussa motivoivana tekijänä nähtiin mahdollisuus osallistua tieteelliseen tutkimukseen ja tutkimuksen toimittaminen yritykselle sen valmistuttua. Haastattelukutsuun myönteisesti vastanneet henkilöt pitivät tutkimusta todennäköisesti mielenkiintoisena näistä seikoista johtuen.

Hirsjärven ja Hurmeen (2001) mukaan tutkimuksen onnistuminen perustuu myös tutkijoiden riittävään asiantuntemukseen ja perehtyneisyyteen tutkimuksen aiheisiin liittyen. Tutkijoiden tulee olla haastattelun aikana mahdollisimman objektiivisia ja olla johdattelematta haastattelua haluamaansa suuntaan (Hirsjärvi & Hurme, 2001). Tutkijoiden osaaminen nähtiin saavutetuksi kirjallisuuskatsauksen teon aikana. Tutkijoiden oma kiinnostus aihetta kohtaan ja aiempi osaaminen aiheeseen liittyen auttoivat tutkimuksen prosessissa.

Haastatteluiden laatu pyrittiin varmistamaan ja kasvattamaan molemmille osapuolille sopivan haastatteluajankohdan sopimisella sekä nauhoittamalla haastattelu testatulla nauhoitusjärjestelmällä. Teknisesti haastatteluiden nauhoittamisessa tai litteroinnissa ei esiintynyt ongelmia. Haastattelut pystyttiin litteroimaan täysin nauhoitetulla aineistolla.

Haastattelujen luotettavuus pyrittiin varmistamaan kertomalla haastateltavalle haastattelun aluksi lyhyesti tutkimuksen tavoitteesta, tarkoituksesta, aiheesta ja etenemisestä. Lisäksi haastattelukutsussa ja haastattelujen alussa kerrottiin, että haastattelutuloksia käsitellään ja ne esitetään tutkimuksen tuloksissa luotettavasti niin, että haastateltavaa yritystä ei voi tunnistaa niiden perusteella

6 TUTKIMUSTULOKSET

Tutkimustulosten läpikäynti aloitetaan esittelemällä mitä haastatteluista selvisi, miten tulokset jaoteltiin ja mikä merkitys tuloksilla on tälle tutkimukselle. Lisäksi arvioimme miten tutkimustuloksia voi hyödyntää hukan tunnistamiseen ja hukan hallintaan missä tahansa ohjelmistojen tuotekehitystä harjoittavassa yrityksessä. Tämän jälkeen käydään läpi itse haastattelut jokaiselta tasolta. Ennen jokaisen tason haastatteluiden läpikäyntiä esitellään vielä lyhyesti, miten kyseinen taso liittyy ohjelmistojen tuotekehitykseen ja mikä merkitys sillä on yrityksen tuotekehitykselle.

6.1 Haastatteluiden tulokset

Tutkimuksen tuloksina saatiin yrityksessä havaittuja erilaisia hukkia, jotka perustuvat täysin tai osittain kirjallisuuskatsauksen hukan määritelmiin. Niiden pohjalta jaoimme eri ohjelmistojen tuotekehityksen hukat osa-alueittain teoreettiseen viitekehykseen. Merkitys on tieteellisesti ja käytännöllisesti suuri, koska niitä voidaan hyödyntää käytännössä ohjelmistojen tuotekehityksessä.

Tutkimustuloksia voidaan hyödyntää hukan tunnistamiseen ja hukan sekä hukan riskin minimoimiseen. Niiden avulla ohjelmisto tuotekehitystä tekevä yritys voi ymmärtää, minkälaista hukkaa tuotekehitysprosessin aikana voi syntyä, miksi sitä syntyy ja millä menetelmillä hukkaa on pyritty minimoimaan haastateltavissa organisaatioissa ohjelmistojen tuotekehityksen kolmella eri tasolla. Koska tutkimus tehdään tapaustutkimuksena monessa eri yrityksessä, ovat tulokset yleistettävissä tieteellisesti sekä myös muihin yrityksiin.

6.2 Portfoliotaso

Portfolio eli tuotekehitys salkku kertoo yrityksen nykyiset ja tulevaisuuden kehityshankkeet (Vähäniitty & Rautiainen, 2005.). Ohjelmistojen tuotekehityksessä portfolio on työkalu, jonka avulla yritys allokoi omia resursseja tuottavuuden kasvattamiseksi, strategian saavuttamiseksi ja riskin hallintaan (Rautiainen, Schantz & Vähäniitty, 2011). Ideoita portfolioon yrityksillä on usein enemmän kuin resursseja näiden toteuttamiseksi. Jatkuvaa ideoiden virtaa täytyy osata hallita, jotta yritys voi jo aikaisessa vaiheessa osata valita oikeat kehityskohteet yrityksen tuotekehitys salkkuun ja tietää mihin kohdistaa resurssit ja milloin (Vähäniitty & Rautiainen, 2005.). Portfolion hallinnalla tarkoitetaan yrityksen tuoteportfolioon liittyvien päätösten tekemistä. Portfolion hallintaan kuuluu päätökset, jotka koskevat yrityksen olemassa olevia tuotteita, niiden elinkaaren hallinta. Portfolion hallintaa on myös mahdollisten uusien tuotteiden arviointi ja päätökset tuotteiden kehittämisestä tai hylkäämisestä. Lisäksi portfolion hallintaan

kuuluu myös ulkopuolisten kumppaneiden ja ulkoistetun kehityksen hallinta (Weerd ym. 2006.).

Tutkimukseen haastateltiin henkilöitä keiden rooli yrityksissä, on ohjelmistojen tuotekehityksessä osallistua yrityksen portfolion hallintaan. Henkilöt ovat yritysten tuotekehityksen vetäjiä, esimiehiä ja vastaavat yritysten tuotekehityksen suunnittelusta, ohjauksesta ja toimeenpanosta.

Seuraavissa luvuissa käydään läpi portfoliotason haastatteluiden vastaukset. Haastateltavilta etsimme vastauksia kysymyksiin: onko yrityksillä käytössä jotain menetelmää portfolionhallintaan, onko yritys tunnistanut hukkaa portfoliotasolla ja onko heillä keinoja hallita ja vähentää hukkaa.

6.2.1 Portfolion hallintamenetelmä

Tässä kappaleessa käydään läpi portfoliotason haastatteluiden vastauksia portfolionhallinnasta. Kysymyksillä yritämme ymmärtää haastateltavilta, onko yrityksillä käytössä jotain menetelmää portfolionhallintaan.

Kaikkien yritysten tuotekehityksen suunnitelmat tulevat suoraan yritysten strategiasta. Yritysten johtoryhmät ja ohjausryhmät hallitsevat tuotekehityksen suunnitelmia ja tekevät päätökset näihin liittyen. Useimmissa yrityksissä päätöksiä ohjaa tuottavuusajatus eli tehdään niitä asioita, joiden tuotto tai hyöty on korkein. Yhdessä yrityksessä päätöksiä ohjaa yrityksen digitalisaation tavoitetila.

Kaikissa yrityksissä ymmärretään käsite tuoteportfolio tai tuotekehityssalkku, joka sisältää kehitettävät ja ylläpidettävät tuotteet. Useimmissa yrityksistä käsitettä ei kuitenkaan käytetä sellaisenaan päivittäisessä työssä. Useimmissa yrityksissä tuoteportfolio käsittää tuotekehitys backlogin ja roadmapin. Backlogilla tarkoitetaan kehitettäviä tuotteita/ ominaisuuksia salkussa, sisältöä ja roadmapilla sisältöä kalenterissa eli koska tuotteet/ ominaisuudet tehdään.
Haastateltujen yritysten roadmap eli tuotekehitys kalenterissa on suunniteltu yrityksestä riippuen karkealla tasolla noin vuoden päähän, yhdessä viiden vuoden päähän.

Kaikissa yrityksissä tuotekehityksen suunnitelmien hallintaan on käytössä järjestelmä nimeltä Jira. Yritykset mallintavat tuotekehityksen suunnitelmat tähän järjestelmään. Portfoliotason suunnitelmia kutsutaan järjestelmässä Epiceiksi. Nämä pilkkoutuvat käyttötapauksiin ja tiketeiksi tuote- ja kehitystasolla.

Kaikissa yrityksissä tuotekehitystä priorisoidaan jatkuvasti liiketoiminnan muuttuvien prioriteettien mukaisesti. Yritysten johtoryhmät ja ohjausryhmät ovat vastuussa tuotekehityksen backlogin priorisoinnista. Osassa yrityksistä priorisointia tapahtuu viikoittain ja yhdessä harvemmin kerran kvartaalissa.

Suurin osa yrityksistä ei käytä portfolionhallintaan mitään tiettyä tunnistettua menetelmää tai mallia. Yhdessä yrityksistä portfolionhallintaa toteutetaan SAFE:n periaatteiden mukaisesti.

Kaikissa yrityksissä Roadmap on visualisoitu Kanbantaululle ja sitä käytetään aktiiviseksi päätösten teon tukena. Yhdessä yrityksistä on toteutettu niin kutsuttu julkaisukerros, josta kaikki päätöksenteossa mukana olevat ja

asianosaiset voivat tutustua yrityksen tuotekehitysflowhun eli mitä yrityksen tuotekehityksessä on tällä hetkellä työn alla ja mitä on viety jo tuotantoon.

Kaikki yritykset pyrkivät portfolionhallinnalla paremmin ymmärtämään ja hyödyntämään omat resurssit. Mihin ja milloin yritys käyttää tuotekehityksen resurssit ja mihin ei, eli pyrkiä tekemään oikeita asioita oikeaan aikaan. Lisäksi portfolinhallinnalla pyritään läpinäkyvyyteen ja avoimuuteen yrityksen sisällä, jotta kaikki tietää mihin ja milloin yritys käyttää tuotekehitys resurssit ja mihin ei. Yksi yrityksistä sanoo, että "portfolionhallinta on yritykselle viestinnän- ja priorisoinnin apuväline." Toisessa "portfolionhallinnalla pyritään parantamaan tuotekehityksen läpinäkyvyyttä, avoimuutta ja kehittämään koko yhteisön luottamusta."

6.2.2 Hukan tunnistaminen portfoliotasolla

Tässä kappaleessa käydään läpi portfoliotason haastatteluiden vastauksia hukan tunnistamisesta. Vastauksista yritämme ymmärtää haastateltavilta, esiintyykö yrityksillä hukkaa portfoliotasolla. Haastateltavilta kysyttiin tunnistavatko he jotain kirjallisuudesta johdettua hukkaa portfoliotasolla. Taulukkoon on koottu kirjallisuudesta johdetut hukat, esiintyykö hukkaa yrityksissä ja yksi esimerkki mikä sen aiheuttaa. Aluksi haastateltavilta kysyttiin, mikä on heidän mielestänsä hukkaa portfoliotasolla. Lopuksi haastateltavilta kysyttiin, mikä on heidän mielestänsä hukkaa muilla tasoilla.

Yritysten mielestä portfoliotasolla heillä hukkaa on, kun:

> "Asioihin kiinnitetään liian varhain huomiota ja näin ollen niitä pyöritellään turhaan."
>
> "Yhteistyökumppaneiden kanssa, pidetään näin ollen liikaa turhia neuvotteluja ja harrastetaan turhia aktiviteetteja."
>
> "Lisäksi portfoliotasolla yritetään edistää montaa eri asiaa yhtä aikaa, asiat etenevät hitaammin ja tarpeet muuttuvat ajan kuluessa."
>
> "Päätökset pitäisi pystyä tekemään niin myöhään kuin mahdollista jotta näin ei kävisi."
>
> "Portfoliotasolla myös usein kiinnitetään huomiota liikaa yksittäisten ihmisten tavoitteisiin ja tarpeisiin ja näin tätä asiaa edistetään liikaa, kun joku asia on jollekin henkilökohtaisella tasolla tärkeä, vaikka se ei olisi linjassa strategian kanssa."
>
> "Portfoliotasolla ei myöskään pidä sortua liian yksityiskohtaiseen suunnitteluun."
>
> "Portfolionhallinnan tehtävä on huolehtia suuntaviivoista ja antaa asiantuntijoiden huolehtia toteutuksesta."

TAULUKKO 2 Eri hukkien esiintyvyys portfoliotasolla

Hukka	**Hukan esiintyvyys**	**Esimerkki**
Osittain tehty työ	Esiintyy kaikissa yrityksiä	Prioriteettien vaihtuessa tehtävät jäävät kesken
Uudelleen oppiminen	Esiintyy osassa yrityksiä	Pitäisi retrota myös portfoliotasolla
Luovutukset	Ei esiinny	
Viivästykset	Esiintyy osassa yrityksiä	Henkilöt suhtautuvat työn vauhtiin eri tavalla. Tämän takia jotkut henkilöt joutuvat odottamaan muita
Tehtävien vaihto	Esiintyy kaikissa yrityksissä	Prioriteettien vaihtuminen aiheuttaa tehtävien vaihtoa
Ylimääräinen käsittely	Esiintyy osassa yrityksiä	Erilaiset henkilöt käsittelevät tehtävät eri tavalla, osa liian tarkasti
Liian tarkat määrittelyt	Ei esiinny	
Päällekkäiset tehtävät	Esiintyy osassa yrityksiä	Läpinäkyvyyden puute aiheuttaa päällekkäisiä tehtäviä
Keskitetty päätöksenteko	Ei esiinny	
Odottaminen	Esiintyy kaikissa yrityksissä	Aikataulujen muutokset aiheuttavat odottamista
Vanhentunut informaatio	Esiintyy osassa yrityksiä	Ajantasaisen tieto ei ollut kaikkien saatavilla

Yritysten mielestä tuotekehitystasolla hukkaa on, kun:

> ”Testaus kuormittuu paljon. Tämä johtuu siitä, että sovelluksilla ja niiden ominaisuuksilla ei ole laatutasoja vaan kaikki testataan samalla tarkkuudella, vaikka kaikki ominaisuudet eivät ole yhtä kriittisiä. Lisäksi testauksen pitäisi olla automatisoidumpaa.”

> ”Työssä esiintyy viivettä, joka johtuu muiden tasojen viiveestä ja odottamattomista töistä. “

6.2.3 Hukan minimointi portfoliotasolla

Tässä kappaleessa käydään läpi portfoliotason haastatteluiden vastauksia hukan minimoinnista. Vastauksista yritämme ymmärtää haastateltavilta pyrkivätkö yritykset minimoimaan hukkaa portfoliotasolla. Aluksi haastateltavilta kysyttiin,

miten heidän mielestään portfolionhallintaa, tuotehallintaa ja tuotekehitystä voisi kehittää.

Yksi yritys toivoisi, että heillä olisi keino mitata tuotteiden potentiaalia ja rahallista arvoa etukäteen portfoliotasolla. "Näin ymmärrettäisiin tuotteiden arvo resurssien ja tuoton näkökulmasta ja osattaisiin kannattavammin suunnitella tuotekehitystä." Tuote- ja tuotekehitystason työtä yhdessä yrityksessä kehitettäisiin niin, että "pitäisi keskittyä optimaalisen flown löytämiseen ja riippuvuuksien poistamiseen ja tätä kautta myös tuotekehitystason työ olisi tehokkaampaa, kun etenemisen esteet on poistettu. "

Osassa yrityksistä tuotekehitys on jaettu tiimeihin, niin, että kaikilla on oma toisistaan poikkeava rooli tuotekehityksessä. Yhdessä yrityksistä yksi tiimi, kehittää uusia tuotteita ja ominaisuuksia, kun toinen keskittyy ylläpitämään nykyisiä. Toisessa yrityksessä tiimit on muodostettu erilaisten isompien kokonaisuuksien ympärille. Molemmissa yrityksissä tiimit tekevät kuitenkin yhteistyötä muun muassa luovutusten helpottamiseksi ja sovellusten riippuvuuksien ymmärtämiseksi. Yhdessä yrityksessä tuotekehitys ei ole jaettu vaan kaikki tekevät kaikkea.

Kaikissa yrityksissä on kehitystä odottavalle työlle backlog. Odottavan työn määrää ei rajoiteta yritysten backlogeissa. Kaikissa yrityksissä samat henkilöt päättävät otetaanko työ kehityksen alle vai ei.

Yrityksissä analysoidaan tuotekehityksen hyötyjä ja kustannuksia keskenään. Yritysten tuotekehityksen tarpeellisuutta analysoitaessa tästä puuttuu tieteellisyys ja tuotekehityksen rahallista potentiaalia ei mitata etukäteen, koska se on usein niin haastavaa tai siihen ei ole keinoa.

Yrityksissä tuotekehityksen ideat ja tarpeet syntyvät sisäisistä- ja ulkoisista sidosryhmistä. Tuotekehityksen tarpeellisuuden arviointia tehdään pääsääntöisesti kustannus ja tuotto näkökulmasta. Yhdessä yrityksistä arviointia tapahtuu myös siitä näkökulmasta, tahdotaanko näyttäytyä edelläkävijänä alalla. Tarpeen analysointi tapahtuu yrityksissä johtoryhmässä tai ohjausryhmässä. Yrityksissä tuotteilla on pääsääntöisesti aina tuoteomistaja, kuka osallistuu suunnitteluun ja analysointiin ja johtoryhmä luottaa omistajan arvioon tuotteen tai siihen tulevan uuden ominaisuuden tärkeydestä.

Portfoliotasolla osalla yrityksistä tuotekehityksen eteenpäin vienti viivästyy useimmiten sisäisten ja ulkoisten sidosryhmien päätösten odottamisesta. Yhdellä yrityksistä epävarmuus resursseista aiheuttaa myös viivettä tuotekehityksen eteenpäinviennissä.

Yritykset eivät portfoliotasolla pyri käyttämään tuotteiden ja ominaisuuksien tekniseen arviointiin ja analysointiin aikaa. Portfoliotasolla yritykset pyrkivät keskittymään päätösten tekemiseen. Yrityksissä luotetaan asiantuntijoihin, joiden arvioiden perusteella päätös eteenpäin viennistä tehdään.

Yrityksissä uusista tuotteista ja ominaisuuksista kerättävä tieto kerätään portfoliotasolla liiketoiminnan ja tekniikan yhteistyöllä, liiketoiminta pystyy tuomaan "asiakkaan äänen" kuuluviin, kun tekniikka pystyy arvioimaan kehitykseen tarvittavia resursseja. Portfoliotasolla asiakkaiden kanssa ei suoraan kommunikoida. Tuoteomistaja ja muut asiantuntijat ovat alusta alkaen mukana uusien tuotteiden ja ominaisuuksien analysoinnista ja tuovat tiedon johtoryhmiin ja ohjausryhmiin päätöksenteon tueksi.

Yrityksissä ei mitata portfolionhallinnan onnistumista juurikaan. Yksi yritys mittaa satunnaisesti tuotekehityksen läpimenoaikaa, asiakastyytyväisyyttä ja budjetissa pysymistä mutta mittaamista on kuitenkin liian vähän, jotta tuloksia voisi analysoida ja näiden avulla kehittää portfoliohallintaa.

Yrityksissä ei pääsääntöisesti portfoliotasolla pyritä tunnistamaan hukkaa. Yhdessä yrityksistä tuotekehitystä ja sen koko ketjua pyritään kehittämään keskustelemalla mutta käsitteenä hukasta ei puhuta. Yhdessä yrityksessä on tunnistettu, että portfoliotasolla syntyy päällekkäistä ja ylimääräistä työtä ja tätä on yritetty minimoida.

Yrityksillä ei ole systemaattisia työkaluja tunnistaa hukkaa portfoliotason työskentelyssä, osassa hukkaa pyritään tunnistamaan vaistolla ja maalaisjärjellä. Yhdessä kerätään jatkuvasti palautetta ja tätä kautta hukkaa pyritään tunnistamaan.

Yritykset eivät ole keskittyneet hukan hallintaan portfoliotasolla. Yhdessä yrityksistä todetaan, että "Portfoliotason prosessi pitäisi mallintaa, jotta tällä tasolla syntyvää hukkaa voitaisiin tunnistaa ja minimoida paremmin". Haastatteluista selviää kuitenkin, että osa yrityksistä pyrkii minimoimaan portfoliotason hukkaa keskustelemalla ja kehittämällä asianosaisten työtapoja. Yksi yrityksistä pyrkii minimoimaan hukkaa sitouttamalla asianosaiset portfoliotyöskentelyyn ja tätä kautta parantamalla tuotekehityksen läpinäkyvyyttä ja avoimuutta.

6.3 Tuotetaso

Ohjelmiston tuotehallinta on liiketoiminnallinen prosessi, joka ohjaa tuotetta koko tuotteen elinkaaren ajan saavuttaakseen tuotteelle suurimman mahdollisen liiketoiminnallisen arvon (Maglyas ym. 2011.).

Tuotehallinta sisältää erilaisia tuotteen elinkaaren hallinnan tehtäviä, jotka Weerd ym. (2006) jakaa hierarkian mukaan neljään eri prosessiin. Portfolion hallinta käsittelee tuotteita yrityksen koko tuote portfoliossa. Tuotteen "roadmapping" eli millä aikataululla tuotteita kehitetään ja milloin uusia ominaisuuksia julkaistaan. Tuotteen julkaisun suunnittelu tarkoittaa mitä tuotteen vaatimuksia tuleviin julkaisuihin toteutetaan ja sisällytetään. Vaatimusten hallinta tarkoittaa tuotteen jokaisen yksittäisen vaatimuksen toiminta tarkoitusta ja miten ja miksi se on tärkeä.

Tutkimukseen haastateltiin henkilöitä, keiden rooli yritysten ohjelmistojen tuotekehityksessä on vastata tuotteenhallinnasta. Henkilöt ovat yritysten tuotepäälliköitä ja palveluvastaavia.

Seuraavissa luvuissa käydään läpi tuotetason haastatteluiden vastaukset. Haastateltavilta etsimme vastauksia kysymyksiin: onko yrityksellä käytössä jotain menetelmää tuotehallintaan, onko yritys tunnistanut hukkaa tuotetasolla ja onko heillä keinoja hallita ja vähentää hukkaa.

6.3.1 Tuotehallintamenetelmät

Tässä kappaleessa käydään läpi tuotetason haastatteluiden vastauksia yritysten tuotteen hallinnasta. Kysymyksillä yritämme ymmärtää haastateltavilta, onko yrityksillä käytössä jotain menetelmää tuotehallintaan.

Haastateltavat työskentelevät yrityksissä tuote- ja palveluvastaavina. Haastateltavien työtehtäviä ovat kehitystiimin veto, toimia rajapintana asiakkaan ja tuotekehityksen välissä sekä uusien ominaisuuksien kartoittaminen, analysointi ja suunnittelu yhdessä kehitystiimin kanssa

Yritykset näkevät tuotehallinnan ominaisuuksien ja tuotteiden suunnitteluna, kehittämisenä ja priorisointina, mitä kehitetään ja miten.

Yritykset näkevät elinkaaren hallinnan tuotteen ylläpitona, jatkokehityksenä ja tuotteen lopetuksena. Osassa yrityksistä elinkaaren hallintaan kiinnitetään huomiota alusta alkaen mutta ei kaikissa. Yhdessä kiinnitetään vain suurempien tuotteiden kohdalla. Elinkaaren hallinnasta yrityksissä vastaa tuotepäälliköt ja tuoteomistajat, päätökset jatkokehityksestä ja lopetuksesta tehdään portfoliotasolla, ylläpitoa tapahtuu jatkuvasti.

Yritysten aikataulu kehitettäville ominaisuuksille ja tuotteille vaihtelee muutamasta kuukaudesta yhteen vuoteen. Priorisointi kuitenkin muuttaa aikataulua jatkuvasti kaikissa yrityksissä. Julkaisun sisältämät ominaisuudet tiedetään noin yhdestä kolmeen kuukauteen ennen julkaisua riippuen yrityksestä.

Yrityksissä tuotekehityksen priorisointi tapahtuu portfoliotasolla, pääsääntöisesti kvartaalin tarkkuudella. Tarkempi priorisointi on tuoteomistajan tai päällikön ja muun kehitystiimin vastuulla. Tuottavimmat tuotteet ja ominaisuudet priorisoidaan pääsääntöisesti korkeimmalla mutta usein myös "pienet" asiat pyritään hoitamaan nopeasti alta pois.

Yritykset toteuttavat tuotehallintaa Kanbanin, Scrumin tai näiden sekoituksen mukaisesti. Yritykset käyttävät tuotehallinnan työkaluna Jira nimistä ohjelmistoa. Lisäksi yrityksillä on käytössä visuaalisia Kanban tauluja kuvaamaan tuotekehitystä, aikataulua yms.

Tuotehallinnan tavoitteet ovat yrityksille erilaiset. Yksi sanoo, että he pyrkivät tuotehallinnalla "tarjoamaan asiakkaille parempaa ja toimivampaa palvelua sekä kehittämään omaa tuotekehitystä ja arkkitehtuuria." Toinen taas sanoo, että sillä pyritään "toimimaan organisoidusti ja parantamaan tuotekehityksen läpinäkyvyyttä ja priorisointia." Kolmannelle tuotehallinta on, että "Tehdään vain niitä asioita, joilla on merkitystä eikä niitä, joista pidetään eniten ääntä. Tuotehallinnalla varmistutaan, että toteutetaan tarpeelliset ominaisuudet ja että niille on käyttäjiä."

6.3.2 Hukan tunnistaminen tuotetasolla

Tässä kappaleessa käydään läpi tuotetason haastatteluiden vastauksia hukan tunnistamisesta. Vastauksista yritämme ymmärtää haastateltavilta, esiintyykö yrityksillä hukkaa tuotetasolla. Haastateltavilta kysyttiin tunnistavatko he jotain kirjallisuudesta johdettua hukkaa tuotetasolla. Taulukkoon on koottu kirjallisuudesta johdetut hukat, esiintyykö hukkaa yrityksissä ja yksi esimerkki mikä sen

aiheuttaa. Aluksi haastateltavilta kysyttiin, mikä on heidän mielestään hukkaa tuotetasolla. Lopuksi haastateltavilta kysyttiin, mikä on heidän mielestään hukkaa muilla tasoilla.

Haastateltavien mielestä tuotetasolla heillä hukkaa on, kun:

> "Jira:ssa on paljon tavaraa, jotka ovat siellä vaikka niille ei mitään tehtäisi."
>
> "Ei suunniteltaisi etukäteen niin suunnitteluun ei kuluisi aikaa ja ei syntyisi niin paljoa turhaa, jos jotain ei tehdäkään."
>
> "Välillä viestintä ei pelaa, josta aiheutuu ylimääräistä työtä. Lisäksi tarpeet muuttuvat joskus työtä tehdessä"

TAULUKKO 3 Eri hukkien esiintyvyys tuotetasolla

Hukka	**Hukan esiintyvyys**	**Esimerkki**
Osittain tehty työ	Esiintyy kaikissa yrityksissä	Priorisoinnin muuttuminen ja ylimääräinen suunnittelu
Uudelleen oppiminen	Esiintyy kaikissa yrityksissä	Henkilöiden välisen kommunikaation puute
Ylimääräiset ominaisuudet	Esiintyy kaikissa yrityksissä	Suuremman kokonaisuuden turhat ominaisuudet ja liian tarkka vaatimusmäärittely
Luovutukset	Ei esiinny	
Viivästykset	Esiintyy kaikissa yrityksissä	Puutteellinen suunnittelu, ominaisuus on määritelty huonosti ja siihen halutaan lisää ominaisuuksia
Tehtävien vaihto	Esiintyy osassa yrityksiä	Henkilöstömuutokset, jossa vastuu vaihtuu
Viat	Esiintyy kaikissa yrityksissä	Suunnitteluun ei osallistu liiketoiminnalliset tai tekniset henkilö
Ylimääräinen käsittely	Esiintyy kaikissa yrityksissä	Liian suuri joukko osallistuu suunnitteluun
Liian tarkat määrittelyt	Esiintyy osassa yrityksiä	Ominaisuuteen määritellään lisää ominaisuuksia kehityksen aikana
Asiakkaan osallistumisen puute	Esiintyy kaikissa yrityksissä	Asiakasta ei osallisteta suunnitteluun tai asiakas ei ole sitoutunut
Päällekkäiset tehtävät	Ei esiinny	

Taulukko jatkuu seuraavalla sivulla.

Keskitetty päätöksentekо	Esiintyy osassa yrityksiä	Päätökset tehdään ylemmällä tasolla
Odottaminen	Esiintyy kaikissa yrityksissä	Päätöksiä joudutaan odottamaan
Vanhentunut informaatio	Esiintyy osassa yrityksiä	Suunnittelu tehdään liian aikaisin

Haastateltavien mielestä portfoliotasolla hukkaa on, kun suunta muuttuu usein nopeasti, priorisointi on puutteellista, ei ajatella tuotteiden elinkaarta, tehdään suuria julkaisuja pienten sijaan, tuotteiden laajuutta ei rajata alussa, jonka vuoksi tuotteen sisältö paisuu ja julkaisu viivästyy. Tuotekehityksessä hukkaa on, kun tuotteissa esiintyy paljon vikoja, tehtävät vaihtuvat, testaus on liian tarkkaa ja manuaalista sekä julkaisusykli hidas.

6.3.3 Hukan minimointi tuotetasolla

Tässä kappaleessa käydään läpi tuotetason haastatteluiden vastauksia hukan minimoinnista. Vastauksista yritämme ymmärtää haastateltavilta pyrkivätkö yritykset minimoida hukkaa tuotetasolla. Aluksi haastateltavilta kysyttiin miten heidän mielestään tuotehallintaa voisi kehittää.
Haastateltavat kehittäisivät tuotehallintaa eri tavoin. Yksi yritys sanoo, että "tuotehallintaa voisi kehittää työmääräarvioiden osalta, usein ne eivät pidä paikkaansa." Toisessa "tuotehallintaa voisi kehittää vaatimusmäärittelyn ja tähän käytetyn ajan osalta. Usein nämä tehdään aivan liian tarkasti etukäteen, kun olennaisimmat asiat saattaa tulla mieleen vasta kehitettäessä." Lisäksi "Toinen kehitettävä kohde on meidän ja asiakkaan työnteon synkronointi niin, että asiakas tietää milloin on heidän vuoronsa esimerkiksi testata, näin kehitys etenisi tehokkaammin." Tällä hetkellä hukkaa pyritään hallitsemaan pitämällä kehitysjono mahdollisimman lyhyenä sekä julkaisemalla niin usein kuin mahdollista.

Yrityksillä on kehitettäville tuotteille ja ominaisuuksille backlog jota käydään jatkuvasti läpi ja jota priorisoidaan ja josta valitaan kehitettävät tuotteet ja ominaisuudet. Useimmilla yrityksillä backlog on usein liian täynnä ja sinne unohtuu asioita, joka aiheuttaa turhaa työtä.

Kaikissa yrityksissä tuotekehityksen tarve analysoidaan johtoryhmässä tai ohjausryhmässä ja mikäli se otetaan kehitykseen, priorisoidaan se karkeasti tällä tasolla. Tämän jälkeen tuotepäällikkö tai tuoteomistaja yhdessä kehitystiimin kanssa tekee sille teknisen analyysin ja tarvittaessa se palautetaan vielä portfoliotasolle käsiteltäväksi tai sen toteutus aikataulutetaan.

Kaikki yritykset varmistuvat ominaisuuden tai tuotteen tarpeesta niin, että he pyrkivät keräämään tietoa tästä mahdollisimman paljon ja monipuolisesti, usein kuitenkin asioita on tehtävä liiketoiminnan jatkuvuuden takia, kun tarpeet kehitykseen tulevat ulkopuolisilta sidosryhmiltä. Täysin uusien innovaatioiden tarpeellisuuden analysointiin ei kuitenkaan ole kenelläkään keinoa.

Kaikki yritykset pyrkivät kehittämään ja julkaisemaan uudet tuotteet ja ominaisuudet pienissä osissa.

Kaikissa yrityksissä tuotteen ja ominaisuuden sisällöstä vastaa sen alueen liiketoiminnallinen asiantuntija. Analysointi ja tekninen suunnittelu tapahtuu yhdessä kehitystiimin kanssa ja usein tässä vaiheessa sisältö vielä muuttuu.

Osassa yrityksiä tuotekehityksen aloitus viivästyy useimmiten ulkoisista sidosryhmistä johtuvista syistä kuten informaation odottelusta tai vastaavasta viivästyksestä. Yhdessä yrityksessä resurssien puute on pääsääntöinen syy, minkä vuoksi kehityksen aloittaminen usein viivästyy.

Yritysten tuoteomistajat ja tuotepäälliköt eivät ole kiinnittäneet omaan ajankäyttöön huomiota. Yksi jakaa oman ajankäytön niin, että korkeimmalle priorisoidut ominaisuudet ja seuraavaksi kehitykseen menevät saavat eniten huomiota. Muut jakavat ajan töiden kesken parhaalla mahdollisella tavalla tilanteen mukaan.

Kaikissa yrityksessä samat henkilöt päättävät kehitettävät ominaisuudet ja tuotteet. Yrityksissä on pääsääntöisesti yksi tuotepäällikkö per tuote mutta osalla yrityksistä tuotteella voi olla enemmänkin kuin yksi tuotepäällikkö, jos tuote on sen verran suuri.

6.4 Tuotekehitystaso

Ohjelmiston ohjelmointi sisältää joukon eri työtehtäviä, jotka kaikki yhdessä hallitussa kokonaisuudessa tuottavat valmiin ohjelmiston (Weerd ym. 2006). Luin & Chanin (2006) mukaan ohjelmointi sisältää seuraavat eri tehtävät: vaatimusten ymmärtäminen, suunnittelu, ohjelmointi, testaus ja integrointi.

Tutkimukseen haastateltiin henkilöitä, keiden rooli yrityksissä on ohjelmistojen tuotekehityksessä osallistua tuotteiden ja ominaisuuksien ohjelmointiin, suunnitteluun ja testaukseen. Henkilöt ovat yritysten ohjelmistoarkkitehteja, tuotekehittäjiä, testaajia ja vastaavat yritysten tuotekehityksen suunnittelusta, toteutuksesta ja testauksesta.

Seuraavissa luvuissa käydään läpi tuotekehitystason haastatteluiden vastaukset. Haastateltavilta etsimme vastauksia kysymyksiin: onko yrityksellä käytössä jotain menetelmää tuotekehitykseen, onko yritys tunnistanut hukkaa tuotekehityksessä ja onko heillä keinoja hallita ja vähentää hukkaa.

6.4.1 Tuotekehitysmenetelmät

Tässä kappaleessa käydään läpi tuotekehitystason haastatteluiden vastauksia yritysten tuotekehitysmenetelmistä. Kysymyksillä yritämme ymmärtää haastateltavilta, onko yrityksellä käytössä jotain menetelmää tuotekehitykseen.
Haastateltavat työskentelevät yrityksissä sovelluskehittäjinä, arkkitehteina ja testaajina. Haastateltavien työtehtäviä ovat tuotteiden ja ominaisuuksien

suunnittelua, määrittelyä sisäisesti ja asiakkaan kanssa sekä ohjelmointia ja testausta.

Yritysten tuotekehitysprosessit noudattavat Kanbanin ja Scrumin malleja sekä näiden yhdistelmää.

Yrityksissä kehitystiimi, product owner ja joskus myös sen alueen asiantuntijat ovat mukana suunnittelemassa tuotteita ja ominaisuuksia. Näin ymmärretään paremmin työn tarkoitus ja ongelma mitä sillä ratkaistaan. Itse kehitys alkaa groomauksella jossa kokonaisuus pilkotaan tiketeiksi. Valmis tuote tai ominaisuus pyritään kehittämään mahdollisimman pienissä osissa, jotta saadaan julkaistua nopeasti jotain valmista.

Yritykset julkaisevat uusia ominaisuuksia ja tuotteita yhdestä kolmeen viikkoon. Jokaisessa julkaisussa pyritään julkaisemaan uusia toimivia ominaisuuksia. Tuotekehitystä yrityksissä priorisoi tuoteomistaja tai tuotepäällikkö.

Yritykset käyttävät tuotekehityksen hallintaan ja seurantaan Jira ohjelmistoa. Tänne on mallinnettu isommat kokonaisuudet, epicit jotka on hajoitettu yksittäisiin käyttötapauksiin, tiketteihin. Tuotekehitys ja testaus toteutetaan eri sovelluksilla. Lisäksi yrityksissä on tuotekehityksen visualisoimiseksi käytössä myös Kanban-taulu kokonaisuuksille ja tiketeille.

Yritykset tehostaisivat tuotekehityksessä muun muassa. tuotekehityksen priorisointia “Kun ominaisuus kehitetään liian aikaisin tai ei ole tarpeellinen syntyy inventaariota, joka on hukkaa.” Lisäksi käyttäjäpalautettaan analysointia pitäisi hyödyntää enemmän “Tämä helpottaisi tuotekehitystä ideoimaan ja tekemään käytettävämpiä ominaisuuksia.”

6.4.2 Hukan tunnistaminen tuotekehitystasolla

Tässä kappaleessa käydään läpi tuotekehitystason haastatteluiden vastauksia hukan tunnistamisesta. Vastauksista yritämme ymmärtää haastateltavilta, esiintyykö yrityksillä hukkaa tuotekehitystasolla. Haastateltavilta kysyttiin tunnistavatko he jotain kirjallisuudesta johdettua hukkaa tuotekehitystasolla. Taulukkoon on koottu kirjallisuudesta johdetut hukat, esiintyykö hukkaa yrityksissä ja yksi esimerkki mikä sen aiheuttaa. Aluksi haastateltavilta kysyttiin, mikä on heidän mielestään hukkaa tuotekehitystasolla. Lopuksi haastateltavilta kysyttiin, mikä on heidän mielestään hukkaa muilla tasoilla

Tuotekehitystason haastateltavat näkevät tuotekehitystasolla hukkana sen, kun toteutetaan turhia ominaisuuksia, ominaisuuden laajuus kasvaa liikaa ja testauksen synnyttämää odottamista. Lisäksi väärä priorisointi aiheuttaa tehtävien vaihtumista ja keskeneräistä työtä, liian tarkat määrittelyt eivät jätä kehittäjälle aina tarpeeksi tilaa ja asiakas ei aina osallistu tarpeeksi määrittelyyn ja testaukseen.

TAULUKKO 4 Eri hukkien esiintyvyys tuotekehitystasolla

Hukka	**Hukan esiintyvyys**	**Kommentit**
Osittain tehty työ	Esiintyy kaikissa yrityksissä	Priorisoinnin muuttuessa tehtävä jää kesken tai turhaksi
Uudelleen oppiminen	Esiintyy kaikissa yrityksissä	Retroja ei järjestetä tarpeeksi
Ylimääräiset ominaisuudet	Esiintyy kaikissa yrityksissä	Tuotteessa on vanhentuneita ominaisuuksia, joita ei käytetä
Luovutukset	Ei esiinny	
Viivästykset	Esiintyy kaikissa yrityksissä	Priorisoinnin muutokset aiheuttavat viivästyksiä
Tehtävien vaihto	Esiintyy kaikissa yrityksissä	Priorisoinnin muutokset aiheuttavat tehtävien vaihtoja
Viat	Esiintyy kaikissa yrityksissä	Johtuvat usein vääristä tuotekehitystyökaluista
Ylimääräinen käsittely	Esiintyy kaikissa yrityksissä	Liian tarkka määrittely ja tuotekehitysprosessin malli
Liian tarkat määrittelyt	Esiintyy kaikissa yrityksissä	Tuotteeseen suunnitellaan alkuvaiheessa liikaa ominaisuuksia
Päällekkäiset tehtävät	Ei esiinny	
Asiakkaan osallistumisen puute	Esiintyy osassa yrityksiä	Asiakkaan toiveet tulevat välikäsien kautta, minkä takia tuotteeseen tehdään vääriä ominaisuuksia
Keskitetty päätöksenteko	Ei esiinny	
Odottaminen	Esiintyy kaikissa yrityksissä	Odotetaan informaatiota asiakkaalta tai kumppanilta
Vanhentunut informaatio	Esiintyy kaikissa yrityksissä	Suunnittelu tehdään liian ajoissa
Viivästynyt vahvistus	Esiintyy osassa yrityksiä	Kehittäjä ei osaa itse testata ominaisuutta

Haastateltavien mielestä kehittäjien olisi hyvä olla tuotetasolla enemmän ja näin ymmärtää asiakkaiden tarpeet paremmin. Nyt tarpeet tulee heille muita kanavia pitkin ja on vaarana, että ei ymmärretä vaatimuksia oikein. Portfoliotasolla on liikaa ideoita, joiden etenemistä tuote- ja portfoliotasolla murehditaan. Tuotekehitystä ei pitäisi rasittaa portfoliotason ajatuksilla liikaa, jotta itse tekeminen ei häiriinny. Myös käyttäjien palautetta pitäisi analysoida paremmin,

jotta ymmärrettäisiin asiakkaiden tarpeet paremmin. Lisäksi tuoteomistaja sekaantuu välillä liian teknisiin asioihin, joka on pois hänen muista tehtävistä.

6.4.3 Hukan minimointi tuotekehitystasolla

Tässä kappaleessa käydään läpi tuotekehitystason haastatteluiden vastauksia hukan minimoinnista. Vastauksista yritämme ymmärtää haastateltavilta pyrkivätkö yritykset minimoimaan hukkaa tuotekehitystasolla. Aluksi haastateltavilta kysyttiin, miten heidän mielestään tuotekehitystä, tuotehallintaa ja portfoliohallintaa voisi kehittää.

Yksi haastateltava sanoo, että tuotekehitys ja tuotetasolla "ei pidä suunnitella liikaa valmiiksi ja suunnittelun pitäisi tapahtua mahdollisimman myöhään, kuitenkin sen verran etukäteen, ettei synny pullonkauloja. Oikea-aikaisuus on tärkeää, ettei ideat kerkeä muuttua ennen toteutusta." Lisäksi "tuotekehityksessä työ tehostuisi, jos voisi keskittyä yhteen asiaan kerralla. Tuotekehityksessä on myös tärkeää, että oikeat henkilöt osallistuvat suunnitteluun ja tuotetasolla myös teknisten henkilöiden pitäisi olla enemmän mukana asiakasrajapinnassa, jotta ymmärrettäisiin tarpeet paremmin." Portfoliotason hukkaa yksi haastateltava vähentäisi "unohtamalla vanhat ideat roskiin ja luottamalla, että tärkeät tulevat esiin asiakkailta ja ei itse yritetä luoda uusia ideoita." Lisäksi tällä tasolla ei pitäisi keskittyä liikaa yksityiskohtiin.
Yrityksissä kehittäjillä on normaalisti yhdestä kahteen asiaa kerrallaan kehitettävänä, osassa yrityksiä kehittäjät saavat itse valita ottavatko toisen tiketin työstettäväksi samanaikaisesti.

Yhdessä yrityksessä henkilöt on jaettu tehtävien luonteen (ylläpito ja uuskehitys) mukaisesti eri tiimeihin. Muissa yrityksissä kaikki henkilöt tekevät molempia tuotteen parissa.

Yrityksissä kehittäjät saavat itse päättää mitä ottavat työn alle mutta pääsääntöisesti prioriteetin mukaisesti. Yhdessä yrityksessä lähes koko kehitystiimi on aina mukana ominaisuuden suunnittelussa alusta alkaen. Muissa yrityksissä kehittäjät ovat mahdollisesti mukana isompien ominaisuuksien suunnittelussa mutta ei kaikkien.

Haastateltavien mielestä ominaisuuden ja tuotteen valmistumisen nopeuteen voidaan vaikuttaa parhaiten mitä pienempiin kokonaisuuksiin se voidaan pilkkoa. Valmistumisnopeuteen vaikuttaa myös, miten monta työtä kehittäjillä on samanaikaisesti kesken ja työrauha.

Suurin viivästymistä aiheuttava tekijä haastateltavien mielestä on huono suunnittelu, keskeytykset aiheuttavat vähemmän viivästymistä. Osassa yrityksiä määrittelyt ovat toteutettu riittävällä tarkkuudella. Tuotekehitystasolla on ymmärrys mikä käyttötarkoitus on, ketkä ovat sen asiakkaita, mitä sen tulee tehdä ja mitä käy, jos sitä ei toteuteta. Kehittäjät voivat vielä haastaa ja sanoa, mikä olisi heidän mielestään järkevin tapa toteuttaa asia. Yhdessä yrityksessä määrittelyt ovat välillä jopa liian yksityiskohtaisia.

Osassa yrityksiä tuotteenomistaja on asiakkaan ääni ja asiakas ei osallistu tuotteen kehittämiseen kuin hyvin harvoin ja silloinkin vain testaukseen. Yhdessä yrityksessä tuotteenomistaja käy jatkuvasti keskustelua asiakkaan kanssa

kehityksen aikana ja joskus kehittäjät kommunikoivat suoraan asiakkaan kanssa, asiakas on mukana tuotekehityksessä koko ajan.

Osassa yrityksiä kehitysvaiheessa asiakkaalta ei saada palautetta vasta, kun ominaisuus on tuotannossa ja mikäli siitä löytyy virheitä. Yhdessä yrityksessä asiakas antaa palautetta jatkuvasti tai vähintään hyväksynnän.

Yrityksissä julkaistaan uusi versio yhdestä kolmen viikon välein, jotta saadaan jatkuvasti uusia ominaisuuksia tuotantoon.

Yrityksissä testaus tehdään huolellisesti. Yhdessä yrityksessä kehittäjät luottavat testaajaan liikaakin, joka aiheuttaa sen, että testaus on usein hidasta.

Haastateltavien mielestä olisi tehokkaampaa, jos "jokainen ominaisuus viedään heti tuotteeseen ja viimeisin versio koko ajan käytössä. Näin saataisiin nopeammin testattua ja tehtyä korjaukset ja julkaistua." Lisäksi kehittäjät keskustelevat ominaisuuksista liian vähän keskenään ja tekevät liian monimutkaisia toteutuksia, kun asiat voisi tehdä usein yksinkertaisemmin.

Yrityksissä mitataan tuotekehitystason hukkaa vikojen- ja asiakaspalautteen määrällä. Myös Jira:sta saatavaa dataa olisi mutta se ei ole joko tarpeeksi kehittynyttä tai sitä ei analysoida tarpeeksi, jotta siitä olisi hyötyä.

Hukkaa pyritään hallitsemaan ja vähentämään yrityksissä järjestämällä säännöllisesti retroja ja puuttumalla esiin tulleisiin ongelmiin ja estämään nämä.

Yritykset varmistuvat työn laadusta katselmoinnilla, testauksella ja asiakkaiden palautteen perusteella ja negatiivisen palautteen vähentymisenä. Lisäksi, kun toiminta tehostuu ja liikevaihto kasvaa niin tehdään oikeita asioita.

6.5 Tulosten yhteenveto

Portfoliotaso

Haastateltavat ovat yritysten johtoa ja osallistuvat aktiivisesti portfolion hallintaan yrityksissä. Yritysten tuotekehityksen suunnitelmat tulevat yritysten strategiasta ja päätökset tekevät yritysten korkein johto. Yrityksissä on tuotekehityksen suunnitelmille oma tuotekehitysportfolio, joskin yrityksillä on tälle hieman eri termejä ja käsitettä ei käytetä päivittäisessä työssä. Enemmän yrityksissä puhutaan backlogista ja roadmapista joilla tarkoitetaan kehitettäviä asioita ja, koska kehitys tapahtuu. Yritysten tuotekehityksen suunnitelmat on kuvattu yrityksestä riippuen 1-5 vuoden aikajaksolle mutta suunnitelmat, sisältö ja aikataulu elää jatkuvasti. Kaikki yritykset käyttävät tuotekehitysten suunnitelmien mallintamiseen Jiraa ja portfoliotasolla yksittäisistä tuotekehityssunnitelmista puhutaan Jirassa Epiceinä. Yrityksissä tuotekehityksen priorisointi tapahtuu johtoryhmissä tai ohjausryhmissä ja se on jatkuvaa, johon vaikuttaa liiketoiminnan muuttuvat prioriteetit. Osassa yrityksiä priorisointia tapahtuu jatkuvasti, viikoittain ja yhdessä sitä tehdään harvemmin. Ainoastaan yksi yritys toteuttaa portfolionhallintaa tunnistetun menetelmän SAFE:n periaatteiden mukaisesti, muilla ei ole käytössä mitään tiettyä menetelmää tai mallia. Yrityksissä portfolionhallinnan tueksi backlog ja roadmap on visualisoitu Kanban taululle suunnittelun tueksi ja lisäksi kokouksista pidetään pöytäkirjaa. Yksi yritys on luonut ns. julkaisukerroksen

johon yrityksen tuotekehitysflow on mallinnettu kaikille asianosaisille nähtäväksi. Portfolionhallinta on kaikille yrityksille sisäinen viestintäväline, portfolionhallinnalla pyritään tekemään tuotekehityksestä läpinäkyvämpää ja tätä kautta poistamaan epätietoisuutta ja tehostamaan tuotekehitystä. Yhdelle yritykselle portfolionhallinta on myös priorisoinnin ja resursoinnin apuväline.

Yritysten tuotekehitys on jaettu tiimeihin ja tiimeillä on erilaiset roolit tuotekehityksessä. Tiimien työssä on kuitenkin riippuvuuksia toisten työhön, joka on huomioitu työn suunnittelussa. Kaikilla yrityksillä on oma backlog kehitettäville ominaisuuksille, odottavan työn määrää ei rajoiteta backlogissa. Yrityksissä samat henkilöt, usein tuotepäälliköt, tuoteomistajat tai muut asiantuntijat tuovat ominaisuuden tarpeellisuuden portfoliotasolle esiin. Tarpeet ja ideat syntyvät sisäisistä- ja ulkoisista sidosryhmistä. Tuotepäällikkö tai tuoteomistaja tai joku muu asiantuntija muodostaa ominaisuudesta kuvauksen tai aloittaa keskustelun, jonka perusteella analysoidaan sen tarpeellisuus. Analysoitaessa verrataan ominaisuuden hyötyjä ja kustannuksia keskenään. Arviointia pyritään tekemään kustannus ja tuotto näkökulmasta. Usein tämä on kuitenkin vain spekulointia ja analyysista puuttuu data ja tieteellisyys, koska se on haastava toteuttaa tai sitä ei ole. Yksi yritys arvioi tarpeellisuutta myös edelläkävijä näkökulmasta. Yritykset pyrkivät olemaan analysoimatta kehitettäviä asioita liikaa vaan luottavat asiantuntijoihin ja keskittyvät vertailemaan tuotteiden ja ominaisuuksien tärkeyttä suhteessa muihin. Asiantuntijat osallistuvat portfoliotason kokouksiin tarvittaessa. Yrityksissä portfoliotasolla tapahtuu odottamista ulkoisista ja sisäisistä sidosryhmistä johtuen ja yhdessä yrityksessä myös resurssien epävarmuudesta johtuen. Yritykset eivät portfoliotasolla kommunikoi suoraan asiakkaiden kanssa vaan liiketoiminta tuo asiakkaan äänen kuuluviin joka tuotepäällikön tai tuoteomistajan ja tekniikan yhteistyöllä muutetaan tarpeeksi. Yritykset eivät mittaa portfolionhallinnassa onnistumista. Satunnaisesti mitataan läpimenoaikoja, asiakastyytyväisyyttä ja budjettia. Yrityksissä ei aktiivisesti pyritä tunnistamaan portfoliotasolla esiintyvää hukkaa eikä tähän ole työkaluja. Yksi yritys käy jatkuvaa dialogia asianosaisten kanssa koko tuotekehitysketjusta mutta käsitteenä hukasta ei puhuta. Toinen yritys kerää palautetta sidosryhmiltä hukan tunnistamiseksi. Yksi yritys sanoo vähentävänsä portfoliotason hukkaa kehittämällä työskentelyn läpinäkyvyyttä ja avoimuutta, toinen maalaisjärjellä ja vaistolla. Yritykset eivät ole keskittyneet portfoliotason hukan hallintaan. Yhden haastateltavan mielestä portfoliotason prosessi pitäisi mallintaa ja ymmärtää paremmin, jotta hukkaa voisi havaita paremmin ja tätä kautta parantaa prosessia ja minimoida hukkaa. Hukkaa kuitenkin minimoidaan kehittämällä koko tuotekehitysprosessia ja sitouttamalla asianosaisia portfoliotyöskentelyyn ja tätä kautta parantamalla läpinäkyvyyttä ja avoimuutta. Portfolionhallintaan yksi yritys toivoisi, että heillä olisi malli, jolla kehitettävien tuotteiden potentiaalia ja arvoa voisi mitata etukäteen, joka tekisi analysoinnista tehokkaampaa. Toinen yritys kehittäisi tuote- ja tuotekehityksen yhteistyötä, jotta löydettäisiin riippuvuudet ja molempien työskentely etenisi rinnakkain paremmassa flowssa ja näin myös tuotekehitys olisi tehokkaampaa.

Tuotetaso

Haastateltavat vastaavat yhdestä tai useasta yrityksen tuotteista ja niiden hallinnasta. Tuotehallinta nähdään yrityksissä ominaisuuksien ja tuotteiden suunnitteluna, kehittämisenä ja priorisointina, mitä kehitetään ja miten. Elinkaaren hallinta on yrityksissä tuotteiden ja ominaisuuksien jatkokehitystä, ylläpitoa tai alasajoa. Kaikissa yrityksissä joko tuotepäällikkö tai tuoteomistaja vastaa elinkaaren hallinnasta. Elinkaaren hallintaan kiinnitetään huomiota osassa yrityksistä alusta alkaen, ei kaikissa. Yhdessä kiinnitetään vain suurempien tuotteiden kohdalla. Roadmapin pituus vaihtelee yrityksittäin muutamasta kuukaudesta vuoteen. Julkaisun sisältö tiedetään noin yhdestä kuukaudesta kolmeen ennen julkaisua. Priorisointi aiheuttaa kaikissa yrityksissä muutoksia jatkuvasti. Priorisointia tehdään portfoliotasolla noin kvartaalin tarkkuudella. Tätä tarkemmasta priorisoinnista kaikissa yrityksissä vastaa tuotepäällikkö tai tuoteomistaja. Tuottavimmat ominaisuudet priorisoidaan usein korkeimmaksi mutta usein myös pienet tehtävät tehdään nopeasti. Yrityksissä tuotehallintaa toteutetaan Scrumin ja Kanbanin sekä näiden yhdistelmän mukaisesti. Yritykset käyttävät tuotehallinnan tukena Jiraa ja Kanban tauluja visualisoimaan työtä, aikataulua ja etenemistä. Tuotehallinnan tavoitteet poikkeavat yrityksittäin. Yksi yritys pyrkii tuotehallinnalla tarjoamaan parempaa ja toimivampaa tuotetta asiakkaille ja kehittämään omaa tuotekehitystä ja arkkitehtuuria. Toinen yritys pyrkii tuotehallinnalla toimimaan organisoidusti ja parantamaan tuotekehityksen läpinäkyvyyttä priorisointia. Kolmas pyrkii tuotehallinnalla siihen, että tehdään tarpeellisia ominaisuuksia eikä niitä, joista pidetään suurinta ääntä.

Yrityksissä on kehitettäville ominaisuuksille backlog, jota käydään jatkuvasti läpi ja jota priorisoidaan roadmapille. Osalla yrityksistä backlogin sisältö on suuri ja sinne unohtuu ja jätetään paljon asioita. Yrityksissä johtoryhmä tai ohjausryhmä analysoi toteutetaanko jokin työ vai ei ja koska, tuotetasolla tehdään teknisempi analyysi ja tarkempi aikataulu. Yritykset varmistuvat ominaisuuden tarpeesta keräämällä tietoa ja dataa ideasta mutta suuri osa kehitystyöstä on sellaista, jonka liiketoiminnan jatkuvuus ja ulkopuoliset sidosryhmät edellyttävät. Uusien innovaatioiden analysointi on haasteellista ja tähän ei ole kenelläkään keinoa. Yritykset pyrkivät kehittämään ja julkaisemaan tuotteet ja ominaisuudet pienissä osissa. Yrityksissä tuotteen ja ominaisuuden sisällöstä vastaa sen alueen asiantuntija. Analysointi ja tekninen suunnittelu tapahtuu yhdessä kehitystiimin kanssa. Osassa yrityksistä tuotekehityksen aloitus viivästyy useimmiten ulkoisista sidosryhmistä johtuvista syistä, yhdessä suurin syy on puutteellisuus ja epävarmuus resursseista. Yrityksissä on yksi tuotepäällikkö tai tuoteomistaja tuotetta kohti, osalla on useampi, jos tuote on suuri. Haastateltavat eivät kiinnitä omaan ajankäyttöön huomiota. Yksi priorisoi ajankäyttöä työhön, jotka on priorisoitu korkeimmalle ja kehitykseen menevään työhön.

Haastateltavat kehittäisivät tuotehallinnassa työmääräarvioita, vaatimusmäärittelyä sekä työnteon synkronointia ja kommunikointia. Portfoliotasolla asioita voisi suunnitella pidemmälle ja suuntaa muutetaan usein liian nopeasti. Lisäksi priorisointi on ongelma, ei ajatella tuotteiden elinkaarta, kehitetään liian suuria julkaisuja ja ei osata asettaa rajoja ja aikatauluja tuotekehitykselle. Tuotekehitystasolla hukkaa aiheutuu vioista, tehtävänvaihdosta, testausprosessi on

liian raskas ja julkaisusyklin pitäisi olla nopeampi ja tässäkin synkronointi kehityksen ja asiakkaan työn osalta pitäisi olla parempaa.

Tuotekehitystaso

Haastateltavat suunnittelevat, toteuttavat ja testaavat yritysten ohjelmistotuotteita. Yritysten tuotekehitysprosessit noudattavat Kanbanin ja Scrumin malleja sekä näiden yhdistelmää. Yrityksissä Product owner, asiantuntijat ja kehitystiimi riippuen yrityksestä suunnittelevat ominaisuudet ja tuotteet. Osallistuttamalla paljon asiantuntijoita ymmärretään ongelma ja tavoite paremmin ja lopputulos on parempi. Itse tuotekehitys tehdään pienissä osissa, jotta saadaan nopeasti julkaistua ja nähdään tuloksia. Yritykset julkaisevat uusia ominaisuuksia yhdestä viikosta kolmeen, riippuen yrityksestä. Yrityksissä tuotekehitystä hallitaan ja seurataan Jira:ssa ja itse työ toteutetaan eri ohjelmointi- ja testaus sovelluksilla. Kanbanboard on näkyvissä työn hahmottamiseksi.

Yrityksissä kehittäjillä on normaalisti 1-2 asiaa kehitettävänä. Osassa yrityksiä kehittäjä saa itse päättää työstääkö useampaa tikettiä samanaikaisesti. Yhdessä yrityksessä kehittäjät on jaettu tiimeihin tehtävien luonteen mukaisesti, uuskehitys ja ylläpito. Muissa yrityksessä kehittäjät tekevät molempia mutta tietyn kokonaisuuden kuten tuotteen parissa. Kehittäjät päättävät itse omat tiketit mutta kuitenkin pääsääntöisesti prioriteetin mukaan. Yhdessä yrityksessä koko kehitystiimi on mukana ominaisuuksien suunnittelussa, muissa yrityksissä vain suurempien ja tärkeimpien. Haastateltavien mielestä tuotekehityksen valmistumisen nopeuteen vaikuttaa se miten pieniin kokonaisuuksiin työ voidaan pilkkoa. Lisäksi mitä vähemmän eri tehtäviä kehittäjällä on samanaikaisesti työn alla vaikuttaa nopeuteen. Huono suunnittelu on suurin viivettä aiheuttava tekijä tuotekehityksessä. Yhden haastateltavan mielestä määrittelyt ovat välillä liian yksityiskohtaisia. Toisessa yrityksessä tuotekehitystasolla on ymmärrys käyttötarkoituksesta, asiakkaista sekä mitä tapahtuu, jos työtä ei tehdä. Näin kehittäjät voivat vaikuttaa paljon toteutustapaan. Osassa yrityksiä asiakasta ei osallisteta kehitykseen, kuin harvoin, usein tuotepäällikkö, tuoteomistaja tai muu asiantuntija on asiakkaan ääni. Yhdessä yrityksessä asiakas on jatkuvasti mukana kehityksessä sekä tuotepäällikön kuin myös kehittäjien kautta. Näin saadaan jatkuvasti palautetta suoraan asiakkaalta, kun muissa palautetta saadaan vasta, kun ominaisuus on julkaistu ja tuotannossa. Yrityksissä julkaistaan uusi versio yhdestä kolmen viikon välein, jotta saadaan jatkuvasti uusia ominaisuuksia tuotantoon. Yrityksissä testaus on huolellista mutta yhdessä haastateltavan mielestä se on jopa liian huolellista, joka hidastaa muuta tuotekehitystä. Yrityksissä mitataan tuotekehitystason hukkaa vikojen- ja asiakaspalautteen määrällä. Lisäksi saatavilla olisi erilaista dataa mutta ei tarpeeksi kehittynyttä tai riittävästi analysoitavaksi. Hukkaa hallitaan ja vähennetään yrityksissä säännöllisillä retroilla ja puuttumalla esiin tulleisiin ongelmiin ja estämään nämä. Yritykset varmistuvat työn laadusta katselmoinnilla, testauksella ja asiakkaiden palautteen perusteella sekä negatiivisen palautteen vähentymisenä. Haastateltavat tehostaisivat tuotekehityksessä priorisointia niin ei syntyisi inventaariota ja pullonkauloja

tuotekehitykseen. Lisäksi käyttäjäpalautteen analysointia pitäisi kehittää, jotta voidaan tehdä käytettävämpiä tuotteita. Tuotekehitystä tehostaisi myös, jos ominaisuudet saisi vietyä heti tuotteeseen, jota kautta voisi testata ja julkaista nopeammin. Myös vuoropuhelu kehittäjien kesken yksinkertaistaisi tuotekehitystä ja tekisi siitä tehokkaampaa. Myös keskittyminen yhteen asiaan kerralla ja osallistuttamalla kaikki sidosryhmät suunnitteluun tehostaa tuotekehitystä.

Haastateltavien oma näkemys mitä hukkaa heillä esiintyy

Portfoliotason haastateltavat näkevät portfoliotasolla hukkana sen, kun suunnitellaan liikaa etukäteen, on liian monta yksittäistä edistettävää asiaa päällekkäin, yksittäisten ihmisten tavoitteet ja tarpeet nousevat yrityksen edelle ja liian yksityiskohtaisen suunnittelun. Tuotetason haastateltavat näkevät tuotetasolla hukkana sen, että suunnitellaan liikaa etukäteen, josta syntyy turhaa työtä ja turhia ideoita varastoon, joille ei tehdä mitään. Lisäksi puutteellisesta viestinnästä johtuen esiintyy esimerkiksi muutospyyntöjä ja tarpeiden muuttumista kesken tekemisen, joka aiheuttaa ylimääräistä työtä. Tuotekehitystason haastateltavat näkevät tuotekehitystasolla hukkana sen, kun toteutetaan turhia ominaisuuksia, ominaisuuden laajuus kasvaa liikaa ja testauksen synnyttämää odottamista. Lisäksi väärä priorisointi aiheuttaa tehtävien vaihtumista ja keskeneräistä työtä, liian tarkat määrittelyt eivät jätä kehittäjälle aina tarpeeksi tilaa ja asiakas ei aina osallistu tarpeeksi määrittelyyn ja testaukseen.

TAULUKKO 5 Eri hukkien esiintyvyys eri ohjelmistojen tuotekehityksen tasolla

Hukka	**Portfoliotaso**	**Tuotetaso**	**Tuotekehitystaso**
Osittain tehty työ	Esiintyy	Esiintyy	Esiintyy
Uudelleen oppiminen	Esiintyy osalla	Esiintyy	Esiintyy
Luovutukset	Ei esiinny	Ei esiinny	Ei esiinny
Viivästykset	Esiintyy osalla	Esiintyy	Esiintyy
Tehtävien vaihto	Esiintyy	Esiintyy osalla	Esiintyy
Ylimääräinen käsittely	Esiintyy osalla	Esiintyy	Esiintyy
Liian tarkat määrittelyt	Ei esiinny	Esiintyy osalla	Esiintyy
Päällekkäiset tehtävät	Esiintyy osalla	Ei esiinny	Ei esiinny
Keskitetty päätöksenteko	Ei esiinny	Esiintyy osalla	Ei esiinny
Odottaminen	Esiintyy	Esiintyy	Esiintyy
Vanhentunut informaatio	Esiintyy osalla	Esiintyy osalla	Esiintyy
Viat		Esiintyy	Esiintyy
Asiakkaan osallistumisen puute		Esiintyy	Esiintyy osalla
Ylimääräiset ominaisuudet		Esiintyy	Esiintyy
Viivästynyt vahvistus			Esiintyy osalla

Haastateltavien oma näkemys hukan hallinnasta

Portfoliotason haastateltavat näkevät, että jotta hukkaa voisi portfoliotasolla hallita ja minimoida portfoliotason malli pitäisi mallintaa ja ymmärtää paremmin, jotta koko prosessia voisi parantaa. Erityisesti tarvittaisiin malli, jolla voitaisiin analysoida tuotekehityksen kannattavuutta paremmin, jotta tiedetään mihin resurssit kannattaa milloin suunnata. Hukkaa hallitaan ja minimoidaan pääsääntöisesti kehittämällä koko tuotekehitysprosessia ja sitouttamalla asianosaisia portfoliotyöskentelyyn. Tuote- ja tuotekehitystason hukkaa vähennettäisiin yhteistyötä kehittämällä. Tuotekehitys sujuisi tehokkaammin, molempien tekeminen etenisi rinnakkain niin riippuvuudet ja etenemisen esteet havaittaisiin aikaisemmin ja työskentely olisi tehokkaampaa.

Tuotetason haastateltavat näkevät, että portfoliotasolla hukkaa voisi vähentää suunnittelemalla kauaskatseisemmin sekä olla muuttamatta prioriteetteja liian herkästi. Myös suunnittelemalla tuotteiden elinkaarta enemmän sekä asettamalla selkeät raamit julkaisujen laajuuteen ja aikatauluihin hukkaa hallittaisiin paremmin. Tuotetasolla hukkaa voitaisiin minimoida kehittämällä työmääräarvioita ja vaatimusmäärittelyä. Lisäksi työnteon synkronointi asianosaisten kesken ja parempi kommunikointi kaikkien tasojen välillä kehittäisi hukan hallintaa tuotetasolla. Hukkaa hallitaan ja minimoidaan pääsääntöisesti pyrkimällä pitämään kehitysjono mahdollisimman lyhyenä ja julkaisemalla uutta mahdollisimman usein. Tuotekehitystasolla tuotepäälliköt ja omistajat sanovat, että erityisesti testausprosessi pitäisi olla tehokkaampi ja edetä synkronoidusti muun kehityksen kanssa, jottei esimerkiksi tehtävänvaihtoa esiinny. Myös julkaisemalla nopeammin ja tätä kautta testaamalla nopeammin saataisiin uusia ominaisuuksia nopeammin käyttöön.

Tuotekehitystason haastateltavat näkevät, että portfoliotasolla hukkaa voisi vähentää luottamalla siihen, että tarpeet tulevat liiketoiminnasta eikä yrittää liikaa itse luoda tarpeita. Portfoliotasolla ei myös pidä keskittyä yksityiskohtiin. Hukkaa tuote- ja tuotekehitystasolla hallittaisiin paremmin, jos ei suunniteltaisi liikaa valmiiksi vaan pyrittäisiin suunnittelemaan mahdollisimman myöhään. Lisäksi tuotekehityksen hukkaa vähentäisi, jos voisi keskittyä vain yhteen asiaan kerralla. Haastateltavien mielestä myös teknisten henkilöiden pitäisi olla enemmän asiakasrajapinnassa, jotta ymmärrettäisiin asiakkaiden tarpeet paremmin. Hukkaa hallitaan ja minimoidaan järjestämällä pääsääntöisesti retroja ja puuttumalla esiin tulleisiin ongelmiin ja estämään nämä.

7 POHDINTA

Tässä luvussa esitetään havainnot empiirisestä tutkimuksesta ja perustellaan havainnot kirjallisuudesta löytyvien teorioiden ja tutkimusten avulla. Luvussa vastataan tutkimuskysymyksiin haastattelututkimuksen ja kirjallisuuden näkökulmista. Lisäksi luvussa pohditaan miten havainnot poikkeavat kirjallisuudesta sekä arvioidaan tutkimusta. Ennen näitä esitellään yleiset havainnot empiirisestä tutkimuksesta. Ensimmäiseksi esitellään vastaukset tutkimuskysymyksiin sekä tutkimusongelmaan. Tämän jälkeen esitetään havainnot empiirisestä tutkimuksesta ja perustellaan havainnot aiempien teorioiden ja tutkimusten kautta. Kappaleessa myös pohditaan miten empiirisen tutkimuksen havainnot poikkeavat kirjallisuuden väittämistä. Lopuksi arvioidaan tutkimuksen tieteellistä kontribuutiota, toteutusta ja havaintojen luotettavuutta, yleistettävyyttä ja rajoitteita.

Tutkimuksen haastatteluista saatiin hyvä kuva, miten yritykset ymmärtävät portfolio, tuote- ja tuotekehitystason sekä mitä hukka on näillä tasoilla, mistä sitä syntyy sekä miten sitä hallitaan ja vähennetään joka tasolla. Keskeisimmät havainnot, mitä vastauksista voidaan tehdä, on että käsite hukka tunnistettiin ja osattiin nimetä joka tasolla. Asia joka haastateltavien mielestä aiheuttaa hukkaa jokaisella tasolla on suunnittelu liian aikaisin.

> "Hukkaa on portfoliotasolla muun muassa, kun asioihin kiinnitetään liian varhain huomiota ja näin ollen niitä pyöritellään turhaan"

> "Tuotetasolla ei pitäisi suunnitella näin paljon etukäteen niin suunnitteluun ei kuluisi aikaa ja ei syntyisi niin paljoa turhaa, jos jotain ei tehdäkään"

> "Tuotekehitys ei pidä suunnitella liikaa valmiiksi ja suunnittelun pitäisi tapahtua mahdollisimman myöhään, kuitenkin sen verran etukäteen suunnitella, ettei synny pullonkauloja. Oikea-aikaisuus on tärkeää, ettei ideat kerkeä muuttua ennen toteutusta."

Lisäksi hukkaa pyritään tunnistamaan ja hallitsemaan aktiivisemmin tuote- ja tuotekehitystasolla, portfoliotasolla hukkaan ja sen hallintaan ei kiinnitetä paljoa huomiota.

> "Portfoliotason prosessi pitäisi mallintaa, jotta tällä tasolla syntyvää hukkaa voitaisiin tunnistaa ja minimoida paremmin"

> "Tuotehallintaa voisi kehittää vaatimusmäärittelyn ja tähän käytetyn ajan osalta. Usein nämä tehdään aivan liian tarkasti etukäteen, kun olennaisimmat asiat saattaa tulla mieleen vasta kehitettäessä."

> "Tuotekehityksessä työ tehostuisi, jos voisi keskittyä yhteen asiaan kerralla"

7.1 Tutkimusongelmaan ja apukysymyksiin vastaaminen

Tutkimuksen tutkimusongelma on "miten voidaan vähentää hukkaa ohjelmistojen tuotekehityksen portfolio tasolla, tuotehallinnan tasolla ja tuotekehityksessä?" Tutkimusongelmaan vastaamiseksi pitää löytää vastaus seuraaviin kysymyksiin ohjelmistojen tuotekehityksen yhteydessä:

1. Mitä on portfolionhallinta
2. Mitä on tuotehallinta
3. Mitä on tuotekehitys
4. Mitä on hukka ohjelmistojen tuotekehityksen eri tasoilla
5. Mitä hukkaa eri ohjelmistojen tuotekehityksen tasoilla esiintyy ja syntyy
6. Mitä hukasta aiheutuu
7. Miten hukkaa eri ohjelmistojen tuotekehityksen tasoilla pyritään tunnistamaan
8. Miten hukkaa ja sen riskiä hallitaan ja minimoidaan eri ohjelmistojen tuotekehityksen tasoilla
    a. Mitä operatiivisia menetelmiä käytetään
    b. Mitä johtamismenetelmiä käytetään

Kirjallisuuskatsaus määrittelee portfolion eli tuotekehityssalkun siten, että portfolio kertoo yrityksen nykyiset ja tulevaisuuden kehityshankkeet (Vähäniitty & Rautiainen, 2005.). Portfolion hallinnalla tarkoitetaan yrityksen tuoteportfolioon liittyvien päätösten tekemistä. Portfolion hallinta on työkalu, jonka avulla yritys allokoi omia resursseja tuottavuuden kasvattamiseksi, strategian saavuttamiseksi ja riskien hallintaan (Rautiainen, Schantz & Vähäniitty, 2011). Lisäksi portfolion hallinnalla yritys pystyy hallitsemaan jatkuvaa ideoiden virtaa ja osaa valita oikeat kehityskohteet yrityksen tuotekehitys salkkuun sekä tietää mihin kohdistaa resurssit ja milloin (Vähäniitty & Rautiainen, 2005.).

Haastateltavat määrittelee portfolion tuoteportfolioksi tai tuotekehityssalkuksi, joka sisältää kehitettävät ja ylläpidettävät tuotteet. Portfolionhallinta on suunnittelun, priorisoinnin ja resursoinnin apuväline. Lisäksi portfolionhallinta nähdään yhtenä sisäisen viestinnän työkaluna, jolla pyritään tekemään tuotekehityksestä läpinäkyvämpää ja tätä kautta poistamaan epätietoisuutta ja tehostamaan tuotekehitystä.

Kirjallisuuskatsaus määrittää tuotehallinnan liiketoiminnalliseksi prosessiksi, joka ohjaa tuotetta koko tuotteen elinkaaren ajan saavuttaakseen tuotteelle suurimman mahdollisen liiketoiminnallisen arvon (Maglyas, Nikula & Smolander, 2011.). Tuotehallinta sisältää erilaisia tuotteen elinkaaren hallinnan tehtäviä, joista Weerd ym. (2006) nostaa neljä tärkeintä: Portfolion hallinta eli tuotteen hallinta yrityksen koko tuoteportfoliossa, Roadmapping eli tuotekehityksen ja julkaisujen aikataulutus, julkaisun suunnittelu eli mitä vaatimuksia tulevat julkaisut sisältävät ja vaatimusten hallinta eli jokaisen yksittäisen vaatimuksen toiminnan tarkoitusta ja miten ja miksi se on tärkeä.

Haastateltavat näkevät tuotehallinnan ominaisuuksien ja tuotteiden suunnitteluna, kehittämisenä ja priorisointina, mitä kehitetään ja miten.

Tuotehallinnalla pyritään kehittämään tuotekehitystä, läpinäkyvyyttä, priorisointia ja arkkitehtuuria. Lisäksi tuotehallinnalla pyritään varmistumaan, että tehdään niitä asioita, joilla on merkitystä ja toteutetaan tarpeelliset ominaisuudet.

Kirjallisuuskatsaus määrittää tuotekehityksen joukoksi eri työtehtäviä, jotka kaikki yhdessä hallitussa kokonaisuudessa tuottavat valmiin ohjelmiston (Weerd ym. 2006). Luin & Chanin (2006) mukaan ohjelmointi sisältää seuraavat eri tehtävät: vaatimusten ymmärtäminen, suunnittelu, ohjelmointi, testaus ja integrointi.

Haastateltavat näkevät tuotekehityksen ominaisuuksien ja tuotteiden suunnitteluna, ohjelmointina, testauksena, integrointina ja ylläpitona. Tuotekehityksellä pyritään tuottamaan toimivampaa ja käytettävämpiä ominaisuuksia tehokkaammin ja nopeammin tuotantoon.

Kirjallisuuskatsaus määrittää hukaksi ohjelmistojen tuotekehityksen prosessissa jotain, joka ei luo arvoa prosessille, lopputuotteelle tai asiakkaalle. Eri tutkijat ovat määritelleet erilaisia ohjelmistojen tuotekehityksessä esiintyviä hukkia, näitä ovat: Osittain tehty työ, ylimääräiset prosessit, ylimääräiset ominaisuudet, tehtävänvaihto, Odottaminen, keskitetty päätöksenteko, viat, uudelleenoppiminen, kahdennettu käsittely, asiakkaan osallistumisen puute ja vanhentunut informaatio.

Haastateltavat näkevät hukkana, kun suunnitellaan asioita liikaa etukäteen, on olemassa liian monta päällekkäin edistettävää asiaa, liian yksityiskohtaisen suunnittelun, yksittäisten ihmisten tavoitteet, puutteellinen viestintä, turhien ominaisuuksien toteutus, kun ominaisuuden laajuus kasvaa liikaa, väärä priorisointi ja asiakas ei osallistu tarpeeksi määrittelyyn ja testaukseen.

Hukkaa syntyy ja siitä seuraa, kun suunnitellaan liikaa etukäteen josta seuraa osittain tehtyä työtä ja turhia ideoita varastoon, on liian monta yksittäistä edistettävää asiaa päällekkäin josta seuraa tehtävien vaihtoa, yksittäisten ihmisten tavoitteet eivät kohtaa yrityksen strategian kanssa, suunnitellaan liian yksityiskohtaisesti josta seuraa ylimääräistä käsittelyä, viestintä on puutteellista, toteutetaan turhia ominaisuuksia josta seuraa ylimääräisiä ominaisuuksia, ominaisuuden laajuus kasvaa liikaa josta seuraa viivästyksiä, testataan manuaalisesti, josta seuraa odottamista, priorisointi muuttuu josta seuraa tehtävän vaihtoa, asiakas ei osallistu kehittämiseen josta seuraa viivästyneitä vahvistuksia.

Kirjallisuuskatsaus määrittää, että ohjelmistojen tuotekehitystä tekevä yritys pystyy tunnistamaan hukan ja eliminoimaan sen ja sen riskit, on yrityksen ohjelmistojen tuotekehitys tehokkaampaa (Poppendieck ja Poppendieck 2006). Hukan tunnistamiseen ja eliminointiin on useita eri tieteellisiä menetelmiä, jotka tarjoavat tähän erilaisia tieteellisiä menetelmiä ja työkaluja. Menetelmiä ovat muun muassa kirjallisuuskatsauksessa esitellyt Lean, Kanban ja Six-Sigma. Työkaluja ovat muun muassa Statistical Process Control, Value Stream Mapping, Kanban-taulu, Kano-analyysi ja Quality Function Deployment.

Haastateltavat näkevät, että hukkaa voitaisiin joka tasolla tunnistaa ja eliminoida eri keinoin ja haastateltavilla on paljon yksittäisiä operatiivisia keinoja tähän. Yritysten tuotehallintaa ja tuotekehitystä tehdään tunnistettujen menetelmien mukaisesti, yhdessä myös portfolionhallintaa ja yrityksillä on käytössä työkaluja, joilla hukkaa voidaan tunnistaa. Hukan tunnistamista ja eliminointia tapahtuu kuitenkin vähän ja joillain tasoilla ei ollenkaan. Haastateltavien mukaan

heiltä puuttuu systemaattiset keinot tehdä tätä. Tuotekehitystasolla esiintyvää hukkaa tunnistetaan vikojen- ja asiakaspalautteen määrällä ja eliminoidaan retroilla. Myös työkaluista saatavaa dataa olisi saatavilla mutta tätä ei hyödynnetä. Tuote- ja portfoliotasolla esiintyvää hukkaa ei osassa yrityksissä pyritä tunnistamaan ja eliminoimaan ollenkaan. Lisäksi jotta hukkaa pystyttäisiin tunnistamaan ja eliminoimaan tehokkaasti joka tasolla pitäisi prosessit mallintaa ja ymmärtää paremmin. Yhdessä yrityksessä hukkaa tunnistetaan mittaamalla tuotekehityksen läpimenoaikaa asiakastyytyväisyyttä, budjetissa pysymistä ja keräämällä palautetta työntekijöiltä. Tunnistaminen on kuitenkin satunnaista, jonka vuoksi kukan eliminointia ei voida tehdä systemaattisesti. Yksi haastateltavista sanoo, että heillä pyritään kehittämään koko tuotekehitys ketjua mutta käsitteenä hukasta ei puhuta.

7.2 Havainnot tutkimuksen ja kirjallisuuskatsauksen välillä

Yrityksen tehokkuus on tärkeää yrityksen toiminnan ja kilpailuedun kannalta. Tehokkuus syntyy monen eri toiminnon optimaalisesta toteutumisesta (Poppendieck ja Poppendieck, 2003; Womack ja Jones, 2003.). Yksi tehokkuutta haittaava tekijä tuotantoteollisuudessa sekä ohjelmistojen tuotekehityksessä on Hukka. Hukka määritellään ohjelmistojen tuotekehityksessä miksi tahansa tehtäväksi tai prosessiksi, joka vie aikaa tai muita resursseja tuotekehitysprosessissa ilman, että se tuottaa arvoa lopputuotteelle, prosessille tai aliprosesseille (Mujtaba, Feldt ja Petersen. 2010; Poppendieck ja Poppendieck. 2016; Al Baik ja Miller. 2016). Käytämme tässä tutkimuksessa hukan määritelminä Poppendieckin ja Poppendieckin (2006) esittämiä ohjelmistojen tuotekehityksen hukkien määritelmiä sekä niiden tukena Al-Baikin ja Millerin (2014) määrittelemiä hukkia. Lisäksi taulukkoon on merkattu, millä tasoilla aikaisempien tutkimusten mukaan hukkaa esiintyy. Esiintyvyys on merkattu, jos yksi tai useampi julkaisu on esittänyt hukan esiintyvän sillä ohjelmistojen tuotekehityksen tasolla tai kuvannut esimerkin hukasta jollakin ohjelmistojen tuotekehityksen tasolla.

TAULUKKO 6 Koonti hukkien esiintyvyydestä ohjelmistojen tuotekehityksessä kirjallisuuskatsauksen mukaan

Poppendieck ja Poppendieck (2006)	**Al-Baik ja Miller (2014)**	**Esiintyy portfolio-tasolla**	**Esiintyy tuote-tasolla**	**Esiintyy tuotekehitystasolla**
Osittain tehty työ		X	X	X
	Ylimääräiset työkalut ja metodit			X
Ylimääräiset ominaisuudet	Liian tarkat määrittelyt		X	X

Taulukko jatkuu seuraavalla sivulla.

Tehtävän vaihto			X	X
Viivästykset	Odottaminen	X	X	X
Luovutukset	Keskitetty päätöksenteko	X	X	X
Viat	Viat			X
Uudelleenop-piminen			X	X
	Kahdennettu käsittely ja prosessit			X
	Asiakkaan osallistumisen puute ja väärät oletukset		X	X
	Vanhentunut informaatio tai versio tuotteesta	X	X	X

Haastateltavien oma näkemys mitä hukkaa heillä esiintyy

Haastateltavilta kysyttiin tunnistavatko he jotain kirjallisuudesta johdettua hukkaa. Taulukkoon on koottu kirjallisuudesta johdetut hukat ja esiintyykö hukkaa yrityksissä-. Aluksi haastateltavilta kysyttiin, mikä on heidän mielestään hukkaa.
Portfoliotason haastateltavat näkevät portfoliotasolla hukkana sen, kun suunnitellaan liikaa etukäteen, on liian monta yksittäistä edistettävää asiaa päällekkäin, yksittäisten ihmisten tavoitteet ja tarpeet nousevat yrityksen edelle ja liian yksityiskohtaisen suunnittelun. Tuotetason haastateltavat näkevät tuotetasolla hukkana sen, että suunnitellaan liikaa etukäteen, josta syntyy turhaa työtä ja turhia ideoita varastoon, joille ei tehdä mitään. Lisäksi puutteellisesta viestinnästä johtuen esiintyy esimerkiksi muutospyyntöjä ja tarpeiden muuttumista kesken tekemisen, joka aiheuttaa ylimääräistä työtä. Tuotekehitystason haastateltavat näkevät tuotekehitystasolla hukkana sen, kun toteutetaan turhia ominaisuuksia, ominaisuuden laajuus kasvaa liikaa ja testauksen synnyttämää odottamista. Lisäksi väärä priorisointi aiheuttaa tehtävien vaihtumista ja keskeneräistä työtä, liian tarkat määrittelyt eivät jätä kehittäjälle aina tarpeeksi tilaa ja asiakas ei aina osallistu tarpeeksi määrittelyyn ja testaukseen.

TAULUKKO 7 Koonti hukkien esiintyvyydestä ohjelmistojen tuotekehityksessä kirjallisuuskatsauksen mukaan

Hukka	**Portfoliotaso**	**Tuotetaso**	**Tuotekehitystaso**
Osittain tehty työ	Esiintyy kaikissa yrityksissä	Esiintyy kaikissa yrityksissä	Esiintyy kaikissa yrityksissä
Uudelleen oppiminen	Esiintyy osassa yrityksiä	Esiintyy kaikissa yrityksissä	Esiintyy kaikissa yrityksissä
Luovutukset	Ei esiinny	Ei esiinny	Ei esiinny
Viivästykset	Esiintyy osassa yrityksiä	Esiintyy	Esiintyy
Tehtävien vaihto	Esiintyy kaikissa yrityksissä	Esiintyy osassa yrityksiä	Esiintyy kaikissa yrityksissä
Ylimääräinen käsittely	Esiintyy osassa yrityksiä	Esiintyy kaikissa yrityksissä	Esiintyy kaikissa yrityksissä
Liian tarkat määrittelyt	Ei esiinny	Esiintyy osassa yrityksiä	Esiintyy kaikissa yrityksissä
Päällekkäiset tehtävät	Esiintyy osassa yrityksiä	Ei esiinny	Ei esiinny
Keskitetty päätöksenteko	Ei esiinny	Esiintyy osassa yrityksiä	Ei esiinny
Odottaminen	Esiintyy kaikissa yrityksissä	Esiintyy kaikissa yrityksissä	Esiintyy kaikissa yrityksissä
Vanhentunut informaatio	Esiintyy osassa yrityksiä	Esiintyy osassa yrityksiä	Esiintyy kaikissa yrityksissä
Viat		Esiintyy kaikissa yrityksissä	Esiintyy kaikissa yrityksissä
Asiakkaan osallistumisen puute		Esiintyy kaikissa yrityksissä	Esiintyy osassa yrityksiä
Ylimääräiset ominaisuudet		Esiintyy kaikissa yrityksissä	Esiintyy kaikissa yrityksissä
Viivästynyt vahvistus			Esiintyy osassa yrityksiä

Tutkimuksen havainnot hukan esiintyvyydestä yrityksissä

Haastateltavien omien mielipiteiden, mitä hukkaa heillä esiintyy lisäksi, haastateltavilta kysyttiin erilaisia kysymyksiä portfolio- ja tuotehallinnasta sekä tuotekehityksen toteuttamisesta. Tutkimuksesta havaittiin vastaajien omien näkemysten lisäksi, mitä hukkaa yrityksissä esiintyy ja mikä sen aiheuttaa.

Portfoliotasolla yrityksissä esiintyy osittain tehtyä työtä. Portfoliotasolla analysoidaan paljon uusia tuotekehitysideoita ja tarpeita, jotka eivät välttämättä

koskaan päädy kehitettäväksi. Toisaalta analysointi tällä tasolla auttaa yrityksiä hylkäämään ei tärkeät ideat ja tätä kautta vähentää hukkaa muilta tasoilta. Tuotetasolla yrityksissä esiintyy osittain tehtyä työtä, kun tehdään teknisempi analyysi portfoliotasolta tuleville tuotekehityksen ideoille ja tarpeille, jotka usein kuitenkin hylätään portfoliotasolla. Toisaalta analysointi tällä tasolla auttaa portfoliotasoa hylkäämään ei kannattavat kehityshankkeet ja tätä kautta vähentää työtä tuote- ja tuotekehitystasoilta. Tuotekehitystasolla yrityksissä esiintyy osittain tehtyä työtä, kun portfolio- ja tuotetason priorisoinnin muutokset vaikuttavat tuotekehityksen työhön ja osa työstä jää kesken. Yritykset pyrkivät ehkäisemään tätä tekemällä kehitystä mahdollisimman pienissä osissa, jotta saataisiin nopeasti julkaistua uutta sekä pitämällä keskeneräisen työn määrän mahdollisimman vähäisenä.

Portfoliotasolla analyysin tekevät suurimmassa osassa yrityksiä samat henkilöt, joten uudelleenoppimisen hukkaa ei esiinny. Samoin tuotetasolla ei esiinny uudelleen oppimisen hukkaa, koska tuotepäälliköt vastaavat yksin tuotteesta. Toisaalta tietoa on liikaa yksittäisillä ihmisillä ja usein asioita dokumentoidaan liian vähän, joka voi aiheuttaa uudelleen oppimisen hukkaa, jos esimerkiksi henkilöt vaihtuvat. Tässä tilanteessa luovutukset ja informaation vaihto aiheuttaisi uudelleen oppimisen hukkaa yrityksissä, kun ei ole dokumentoitu ja kommunikoitu tarpeeksi. Tähän vaikuttaa erityisesti yrityksen henkilöstön vaihtuvuus. Tuotekehitystasolla osassa yrityksissä aiheutuu uudelleen oppimisen hukkaa, kun tuotekehitystä ei ole jaettu esimerkiksi ylläpidon ja uuden tuotekehityksen välille vaan kaikki tekevät kaikkea tärkeyden mukaan ja joutuvat opettelemaan jatkuvasti uusia asioita.

Portfoliotasolla yrityksissä ei esiinny luovuttamisesta aiheutuvaa hukkaa sillä samat henkilöt vastaavat analyysista alusta loppuun. Tuotetasolla osassa yrityksiä luovutukset aiheuttavat hukkaa, kun tuotepäällikkö vaihtuu ja erityisesti mikäli tietoa ei ole dokumentoitu tarpeeksi. Tähän vaikuttaa erityisesti yrityksen henkilöstön vaihtuvuus. Tuotekehitystasolla yrityksissä ei esiinny luovuttamisesta aiheutuvaa hukkaa. Tuotekehitys on vahvasti alusta asti mukana suunnittelussa ja toteutuksessa, jolla ehkäistään sitä, että tietoa häviäisi hankkeen edetessä suunnittelusta toteutukseen.

Portfolio- ja tuotetasolla yrityksissä esiintyy viivästyksistä aiheutuvaa hukkaa. Tämän aiheuttaa, kun joudutaan odottamaan informaatiota sisäisiltä ja ulkoisilta sidosryhmiltä sekä epävarmuus resursseista. Tuotekehitystasolla yrityksissä esiintyy viivästyksistä aiheutuvaa hukkaa. Viivästyksiä aiheuttaa, jos joudutaan edistämään montaa asiaa yhtä aikaa, joka vaikeuttaa keskittymistä sekä huono suunnittelu, josta seuraa turhaa työtä, lisäsuunnittelua ja mahdollisesti tuotteen laajuuden kasvamista.

Portfoliotasolla yrityksissä ei esiinny liian tarkoista määrittelyistä aiheutuvaa hukkaa. Yritykset eivät analysoi ideoita teknisellä tasolla vaan pyrkivät vertaamaan kustannuksia hyötyihin ja tämän perusteella tekevät päätöksen. Tuotetasolla osassa yrityksiä liian tarkat määrittelyt aiheuttavat hukkaa tuotetasolla. Tämän aiheuttaa muun muassa se, kun tuotepäällikkö keskittyy liikaa teknisiin ongelmiin. Tuotekehitystasolla yrityksissä ei esiinny liian tarkoista määrittelyistä aiheutuvaa hukkaa. On tärkeää, että kuvauksesta selviää käyttötarkoitus, asiakkaat ja toiminta mutta ei liian yksityiskohtaisesti.

Portfoliotasolla yritykset eivät osallista asiakasta, joka on hukkaa. Toisaalta asiakkaan osallistaminen portfoliotasolla tapahtuu sisäisten sidosryhmien kautta, joka on tehokkaampaa, kun tätä kautta tavoitetaan suurempi määrä asiakkaita. Tuote- ja tuotekehitystasolla osa yrityksistä eivät osallista asiakasta kehitykseen. Tähän vaikuttaa selkeästi yrityksen asiakkaiden määrä. Mikäli asiakaskunta on suuri, on tehokkaampaa kerätä syötteitä sisäisten sidosryhmien kautta, jotka ovat asiakasrajapinnassa työnsä puolesta valmiiksi ja tätä kautta kerätä ideoita ja palautetta. Mikäli asiakaskunta pienempi osallistutetaan asiakas suoraan tuotteen kehitykseen.

Portfoliotasolla osalla yrityksistä esiintyy ylimääräistä käsittelyä, kun asioita pyöritellään johtoryhmässä ja tai ohjausryhmässä liikaa liian monesta näkökulmasta, joka aiheuttaa hukkaa. Tähän vaikutti yrityksen koko, ikä ja tätä kautta työskentelykulttuuri. Tuotetasolla osassa yrityksiä esiintyy ylimääräistä käsittelyä tuotetasolla, kun analysointiin ja suunnitteluun osallistuu liian suuri joukko. Toisaalta tällä usein varmistetaan se, että tuote on suunniteltu oikein ja kaikkien näkökulma tulee esiin. Tuotekehitystasolla yrityksissä esiintyy ylimääräistä käsittelyä. Tätä aiheutuu, kun kommunikointi ei toimi ja kehittäjä pyörittelee asioita yksin eikä tiimissä.

Yrityksissä esiintyy tehtävän vaihtoa kaikilla tasoilla ja tämän aiheuttaa muutokset yrityksen tuotekehityksen priorisoinnissa. Portfoliotasolla tehdyt päätökset, kun joudutaan priorisoimaan tuotekehityshankkeita uudelleen muuttuvan ympäristön vuoksi valuvat niin sanotusti alaspäin ja vaikuttavat kaikkien tekemiseen. Tuotekehitystasolla tehtävän vaihtoa aiheuttaa lisäksi testaus, kun kehittäjä joutuu odottamaan testaajaa ja näin vaihtamaan tehtäviä sitä mukaan, kun testaus etenee.

Portfoliotasolla osalla yrityksistä esiintyy päällekkäisiä tehtäviä johtoryhmässä ja tai ohjausryhmässä. Tämän aiheuttaa puutteellinen viestintä ja organisaatiorakenne ja tähän vaikutti selkeästi yrityksen koko ja tuotteiden määrä portfoliossa. Tuotetasolla yrityksillä on yksi tuotepäällikkö tuotetta kohti, joten päällekkäisiä tehtäviä ei tällä tasolla pääsääntöisesti synny. Tuotekehitystasolla päällekkäisiä tehtäviä ei yrityksissä ole.

Portfoliotasolla yrityksissä esiintyy keskitettyä päätöksentekoa, koska johtoryhmä ja tai ohjausryhmä on elin, joka tekee päätökset, jotka vaikuttavat kokonaisuuteen. Tämä on kuitenkin tällä tasolla hyvä asia, kun päätökset vaikuttavat yrityksen kokonaiskuvaan ja näin muilla tasoilla on selkeä ymmärrys tuotekehityksen suunnista. Tuotetasolla yrityksissä ei esiinny keskitettyä päätöksentekoa. Tuote- ja tuotekehitystasolla yrityksissä muun muassa suunnitellaan julkaisujen sisältö mutta tähän osallistuu koko tiimi ja päätös mitä julkaisuun sisältyy, on yhteinen.

Portfoliotasolla osalla yrityksistä informaatio johtoryhmässä ja tai ohjausryhmässä on välillä vanhaa. Tämän aiheuttaa puutteellinen viestintä ja tähän vaikutti selkeästi yrityksen koko. Tuote- ja tuotekehitystasolla vanhentunut informaatio aiheuttaa osassa yrityksissä hukkaa, kun suunnitellaan liian aikaisin. Tähän vaikutti selkeästi yrityksen tuotekehitysjonon suuruus. Oikea aikainen suunnittelu oli kaikkien haastateltavien mielestä yksi tärkein tekijä hukan ehkäisemiseksi.

Tuote- ja tuotekehitystasolla ylimääräiset ominaisuudet aiheuttavat hukkaa. Ylimääräiset ominaisuudet syntyvät, kun kokonaisuuden laajuus kasvaa liikaa, ominaisuudet vanhenevat, liian tarkat vaatimusmäärittelyt ja kun portfoliotasolla tehdään virheitä ideoiden ja tarpeiden tarpeellisuuden analysoinnissa.

Tuote- ja tuotekehitystasolla viat aiheuttavat hukkaa. Vikoja tulee, kun suunnitteluun ei osallistu asiakas, liiketoiminta ja tai tekniikka tarpeeksi.

Tuotetasolla esiintyy odottamista. Tämän aiheuttaa se, kun joudutaan odottamaan tietoa ja tai päätöksiä sidosryhmiltä. Tuotekehitystasolla esiintyy myös odottamista, kun joudutaan odottamaan testausta, asiakkaan hyväksyntää ja julkaisua.

Tuotekehitystasolla osa yrityksistä joutuu usein odottamaan asiakkailta palautetta uusista tuotteista ja ominaisuuksista. Tähän vaikuttaa se, että nämä yritykset eivät osallista asiakasta tuotekehitykseen.

TAULUKKO 8 Tutkimuksen havainnot hukista eri ohjelmistojen tuotekehityksen tasoilla

Hukka	**Portfoliotaso**	**Tuotetaso**	**Tuotekehitystaso**
Osittain tehty työ	Esiintyy kaikissa yrityksissä	Esiintyy kaikissa yrityksissä	E Esiintyy
Uudelleen oppiminen	Ei esiinny	Ei esiinny	Esiintyy osassa yrityksiä
Luovutukset	Ei esiinny	Esiintyy osassa yrityksiä	Ei esiinny
Viivästykset	Esiintyy kaikissa yrityksissä	Esiintyy kaikissa yrityksissä	Esiintyy kaikissa yrityksissä
Tehtävien vaihto	Esiintyy kaikissa yrityksissä	Esiintyy kaikissa yrityksissä	Esiintyy kaikissa yrityksissä
Ylimääräinen käsittely	Esiintyy osassa yrityksiä	Esiintyy osassa yrityksiä	Esiintyy kaikissa yrityksissä
Liian tarkat määrittelyt	Ei esiinny	Esiintyy osassa yrityksiä	Ei esiinny
Päällekkäiset tehtävät	Esiintyy osassa yrityksiä	Ei esiinny	Ei esiinny
Keskitetty päätöksenteko	Ei esiinny	Esiintyy osassa yrityksiä	Ei esiinny
Odottaminen		Esiintyy kaikissa yrityksissä	Esiintyy kaikissa yrityksissä
Vanhentunut informaatio	Esiintyy osalla	Esiintyy osassa yrityksiä	Esiintyy osassa yrityksiä

Taulukko jatkuu seuraavalla sivulla

Viat		Esiintyy kaikissa yrityksissä	Esiintyy kaikissa yrityksissä
Asiakkaan osallistumisen puute	Esiintyy kaikissa yrityksissä	Esiintyy osassa yrityksiä	Esiintyy osassa yrityksiä
Ylimääräiset ominaisuudet		Esiintyy kaikissa yrityksissä	Esiintyy kaikissa yrityksissä
Viivästynyt vahvistus			Esiintyy osassa yrityksiä

Tutkimuksen havainnot hukasta suhteessa haastateltavien näkemykseen

Haastateltavien oman näkemyksen ja haastatteluista tehtyjen havaintojen perusteella yrityksissä esiintyvä hukka poikkeaa hieman toisistaan. Havaintojen perusteella portfoliotasolla ei esiinny uudelleen oppimisen hukkaa, koska samat henkilöt tekevät päätökset. Henkilöstön vaihtuvuus ja dokumentaation puute voi kuitenkin aiheuttaa uudelleen oppimisen hukkaa. Lisäksi havaintojen perusteella portfoliotasolla kaikissa yrityksissä esiintyy viivästyksiä, joita aiheuttavat sisäiset- ja ulkoiset sidosryhmät sekä epävarmuus resursseista.

Samoin kuin portfoliotasolla havaintojen perusteella tuotetasolla ei esiinny uudelleen oppimisen hukkaa, koska samat henkilöt tekevät päätökset. Henkilöstön vaihtuvuus ja dokumentaation puute voi kuitenkin aiheuttaa uudelleen oppimisen hukkaa. Tästä syystä luovutukset aiheuttavat osassa yrityksiä hukkaa, kun henkilöstö tai vastuut tuotteiden välillä vaihtuu ja tietoa ei ole dokumentoitu tarpeeksi. Lisäksi tehtävän vaihtoa esiintyy kaikissa yrityksissä, jonka aiheuttaa muutokset tuotekehityksen priorisoinnissa. Ylimääräistä käsittelyä esiintyy osassa yrityksiä tuotetasolla, jonka aiheuttaa se, että analysointiin ja suunnitteluun osallistuu liian suuri joukko. Myös asiakkaan osallistamisen puutetta on osassa yrityksiä, kun asiakasta ei osallistuteta aktiivisesti tuotekehitykseen.

Tuotekehitystasolla havaintojen perusteella uudelleen oppimista esiintyy osassa yrityksistä, kun tuotekehitystä ei ole jaettu kehittäjien välille esimerkiksi ylläpitoon ja tuotekehitykseen. Tuotekehitystasolla liian tarkoista määrittelyistä aiheutuvaa hukkaa ei havaintojen mukaan esiinny, kuvaukset tulevat kehittäjille, sillä tasolla, että niistä ymmärtää käyttötarkoituksen mutta ei liian yksityiskohtaisesti jättäen kehittäjille tilaa. Vanhentunutta informaatiota ei tuotekehitystasolla esiintynyt kaikissa yrityksissä. Osalla ongelman aiheuttaa se, kun suunnitellaan liian aikaisin.

Keinot hukan hallintaan kirjallisuuskatsauksen mukaan

Hukan minimoinnin tavoite on pysyvästi kehittää ohjelmistojen tuotekehityksen tehokkuutta (Al-Baik ja Miller, 2014). Jotta hukkaa voidaan minimoida, täytyy se ensin tunnistaa. Hukan tunnistamiseen, hallintaan ja minimoimiseen voidaan käyttää erilaisia menetelmiä. Luvussa neljä esittelimme yksityiskohtaisemmin kirjallisuuskatsauksessa useimmiten ilmenneet menetelmät.

Menetelmistä esitetty Lean on enemmän ajattelutapa, jonka omaksumalla ja periaatteita noudattaen yritys voi tunnistaa ja eliminoida hukkaa tuotekehityksessä (Mujtaba ym. 2010.) Lisäksi Lean tarjoaa kaksi erilaista työkalua, joilla yritys voi tarkastella visuaalisesti tuotekehitysprosessia ja siinä esiintyvää hukkaa. Statistical Process Control visualisoi joukon yksittäisiä vastaavia tuotekehitysprosesseja ja esittää mitkä ovat menneet suunnitellusti ja mitkä eivät (Womack & Jones, 1990). Value Stream Mapping visualisoi koko tuotekehitysprosessin ja siihen sisältyvät arvoa tuottavat ja ei arvoa tuottavat tekijät (Mujtaba ym. 2010.).

Toinen työn visualisointiin perustuva menetelmä ja työkalu nimeltä Kanban. Sen keskiössä on taulu, jonka avulla pystytään visuaalisesti hahmoittamaan nykyiset ja tulevat tehtävät sekä niihin kuluva aika. Kanban tarjoaa yritykselle keinon tunnistaa ja eliminoida hukkaa yrityksen tuotekehityksessä visualisoimalla työn eri vaiheet ja tätä kautta ymmärtää tuotekehityksen etenemisen paremmin (Kniberg & Skarin, 2010; Ahmad ym. 2013.).

Kolmas asiakaskeskeinen menetelmä nimeltä Six-Sigma ja siitä johdettu Lean Six-Sigma muodostuu kahdesta yrityksen omaksumasta tekijästä. Ensimmäinen on ymmärtää käyttää niin vähän resursseja kuin mahdollista tarvittavan arvon tuottamiseksi asiakkaalle. Toinen on tunnistaa ja ymmärtää, mitkä prosessit oikeasti lisäävät tuotteen arvoa asiakkaalle. Lean Six-Sigma tarjoaa kaksi eri työkalua Kano-analyysin asiakkaiden tarpeiden ymmärtämiseksi ja näiden priorisoimiseksi. Toinen työkalu nimeltään Quality Function Deployment on työkalu asiakkaan vaatimusten kääntämiseksi tuotteen ominaisuuksiksi tuotteen myöhemmissä vaiheissa kehitysprosessia (Raffo ym. 2000.).

Haastateltavien näkemykset hukan hallinnasta ohjelmistojen tuotekehityksessä

Haastateltavilla oli paljon ajatuksia, miten tuotekehityksen hukkaa voisi hallita ja vähentää kaikilla tasoilla. Portfoliotasolla hukkaa voisi hallita ja vähentää, jos olisi malli olla voitaisiin analysoida tuotekehityksen kannattavuutta paremmin, jotta tiedetään mihin resurssit kannattaa milloin suunnata. Lisäksi hukkaa vähentäisi, jos portfoliotasolla suunniteltaisiin kauaskatseisemmin ja prioriteetit eivät muuttuisi niin herkästi. Myös ajattelemalla jo portfoliotasolla tuotteen elinkaarta ja julkaisujen laajuutta voitaisiin hukkaa hallita paremmin. Myös turhaa ideointia tulisi vähentää ja luottaa siihen, että tuotekehityksen tarpeet tulevat liiketoiminnasta eikä yrittää liikaa itse luoda tarpeita.
Tuotetasolla hukkaa voisi hallita ja vähentää kehittämällä työmääräarvioita ja vaatimusmäärittelyä. Lisäksi työnteon synkronointi asianosaisten kesken ja parempi kommunikointi kaikkien tasojen välillä kehittäisi hukan hallintaa tuotetasolla.

Tuotekehitystason hukka vähenisi myös, kun tasojen välistä yhteistyötä kehitettäisiin. Tuotekehitys sujuisi tehokkaammin, kun tuote- ja tuotekehityksen tekeminen etenisi rinnakkain. Näin riippuvuudet ja etenemisen esteet havaittaisiin aikaisemmin ja työskentely olisi tehokkaampaa. Lisäksi

testausprosessin pitäisi olla tehokkaampi ja edetä synkronoidusti muun kehityksen kanssa, jottei esimerkiksi tehtävänvaihtoa esiinny tai synny pullonkauloja. Myös julkaisemalla nopeammin ja tätä kautta testaamalla nopeammin saataisiin uusia ominaisuuksia nopeammin käyttöön. Lisäksi tuotekehityksen hukkaa vähentäisi, jos voisi keskittyä vain yhteen asiaan kerralla. Haastateltavien mielestä myös teknisten henkilöiden pitäisi olla enemmän asiakasrajapinnassa, jotta ymmärrettäisiin asiakkaiden tarpeet paremmin.

Lisäksi kaikilla tasoilla pitäisi pyrkiä siihen, että ei suunniteltaisi liikaa valmiiksi vaan pyrittäisiin suunnittelemaan mahdollisimman myöhään.

Tutkimuksen havainnot hukan hallintamenetelmistä

Portfoliotasolla hukkaa pyritään hallitsemaan ja vähentämään yrityksissä kehittämällä koko tuotekehitysprosessia ja sitouttamalla asianosaisia portfoliotyöskentelyyn. Tuotetasolla hukkaa hallitaan ja minimoidaan pääsääntöisesti pyrkimällä pitämään kehitysjono mahdollisimman lyhyenä ja julkaisemalla mahdollisimman usein. Tuotekehitystasolla hukkaa pyritään hallitsemaan ja vähentämään yrityksissä järjestämällä säännöllisesti retroja ja puuttumalla esiin tulleisiin ongelmiin ja estämään nämä sekä rakentamalla laadukasta ohjelmistoa. Lisäksi yrityksissä on jokaisella tasolla käytössä Kanban taulu työn määrän ja etenemisen kuvaamiseen. Yritykset toteuttavat tuotehallintaa ja tuotekehitystä Scrumin tai Kanbanin menetelmien oppien mukaisesti, yhdessä yrityksessä toteutetaan myös portfolionhallintaa SAFE:n periaatteiden mukaisesti.

Vastausten perusteella yrityksillä on vähän keinoja hallita ja vähentää hukkaa. Yritykset käyttävät pääasiassa Leanista ja Kanbanista tuttuja keinoja hallita ja vähentää hukkaa. Kirjallisuuskatsauksessa esiteltyjä työkaluja ja menetelmiä yrityksillä on käytössä muun muassa Kanban taulun käyttö työn visualisoimiseen sekä Kanban tuotekehityksen menetelmänä, jonka yksi periaate on työn määrän rajoittaminen. Muita kirjallisuuskatsauksessa esiteltyjä keinoja ovat koko tuotekehitysprosessin optimointi, toimittamalla nopeasti ja rakentamalla laadukasta ohjelmistoa, jotka ovat Leanin periaatteita. Muita keinoja yrityksillä hukan hallitsemiseen ja vähentämiseen on muun muassa Scrum tuotekehitysmenetelmästä tuttu Retrospektiivi, jossa mietitään jälkikäteen, miten työskentelyä voisi parantaa.

Tutkimuksen havainnot hukan hallintamenetelmistä suhteessa haastateltavien näkemykseen

Haastateltavien oman näkemyksen ja haastatteluista tehtyjen havaintojen perusteella yrityksissä on vähän tunnistettuja keinoja hukan hallintaan ja vähentämiseksi mutta paljon uusia ideoita. Osa keinoista on uusia ja osa Leanista ja Kanbanista tuttuja keinoja. Näitä olivat muun muassa ”testaamalla toteutusvaiheessa” ja ”toimittamalla nopeasti” jotka ovat Leanin periaatteita. Myös nopeat sprintit, kanban taulun käyttö ja samanaikaisen työn määrän rajoittaminen ovat Leanista tuttuja periaatteita.

Varsinaisia hukan hallintaan ja vähentämiseen tarkoitettuja työkaluja yrityksillä ei ole käytössä muita kuin Kanban taulu. Kuitenkin Jira ja monet muut yritysten käyttämät työkalut muun muassa palautteen keräämiseen tarkoitetut työkalut tähtäävät hukan hallintaan ja vähentämiseen. Työkalujen ja johtamismenetelmien lisäksi haastateltavilla oli paljon ajatuksia työn tekemiseksi joka tasolla, jolla hukkaa voitaisiin hallita ja vähentää paremmin.

7.3 Tutkimuksen luotettavuus, yleistettävyys ja rajoitukset

Tässä kappaleessa käsitellään tutkimuksen luotettavuutta, yleistettävyyttä ja rajoituksia. Tutkimuksella tarkoitetaan tämän tutkimuksen teorian, tutkimusmenetelmien ja tutkimustulosten osa-alueita. Tutkimuksen tutkimusotannan ja valittujen analyysimenetelmien luotettavuutta käsiteltiin erikseen luvussa 5.

Tämän tutkimuksen luotettavuuden arviointiin käytetään reliabiliteettia ja validiteettia. Reliabiliteetti tarkoittaa sitä, miten luotettavasti tutkimuksessa käytettävät menetelmät mittaavat tutkimuksen kohteena olevaa ilmiötä. Korkea reliabiliteetti tarkoittaa sitä, että tutkimuksessa tehdyt havainnot eivät ole sattumanvaraisia, vaan mikäli tutkimus toistetaan uudelleen, ovat seuraavan tutkimuksen tulokset yhtenevät tämän tutkimuksen tuloksiin. Reliabiliteettia mitataan kahdella eri osatekijällä: stabiliteetilla ja konsistenssilla. Stabiliteetti kuvaa sitä, miten tutkimuksessa käytetyt menetelmät kestävät aikaa, eli säilyvät samanlaisina riippumatta satunnaisista olosuhdemuutoksista tai ajan kulusta. Konsistenssilla tarkoitetaan sitä, että esimerkiksi haastattelututkimuksessa teeman eri alakysymykset mittaavat samaa asiaa. (KvaliMOTV, 2006).

Validiteetti tarkoittaa sitä, miten hyvin tutkimus mittaa sillä tutkittavaa ilmiötä. Tässä tutkimuksessa validiteetin tasoa pyrittiin kasvattamaan perustamalla haastattelututkimus kirjallisuuskatsauksen havaintoihin, tuloksiin ja teorioihin. Sen lisäksi aineiston keruu tehtiin kohdejoukosta, joka pyrittiin valikoimaan kirjallisuudesta tehtyjen havaintojen perusteella. Näin tutkittiin sopivia kohdeorganisaatioita ja ihmisiä. (KvaliMOTV, 2006.)

Reliabiliteettia pyrittiin kasvattamaan kuvaamalla tutkimuksen toteutus ja menetelmät mahdollisimman tarkasti tutkimusprosessin aikana. Kirjallisuuskatsauksen havainnot kerättiin tietokantaan. Haastattelututkimuksen reliabiliteettia kasvatettiin suunnittelemalla haastattelun protokolla ja teemoittelu mahdollisimman hyvin ja kuvaamalla protokollat, jolla ne voidaan toteuttaa myös jatkossa uudelleen. Tämä kasvatti tutkimuksen tarkkuutta. Lisäksi haastattelukysymykset asetettiin niin, että ne eivät olleet johdattelevia, vaan haastateltavan vastaus kuvasi parhaiten heidän mielipiteitään. Haastattelujen litterointi sekä suorat lainaukset kasvattavat tutkimuksen validiteettiä. Lisäksi tämän tutkimuksen liitteiksi lisättiin haastattelukutsu, jota voidaan käyttää myös toistettavassa jatkotutkimuksessa.

Haastattelututkimuksessa käytettiin runsas määrä alakysymyksiä yhdessä teemassa. Tämän voidaan nähdä lisäävän tutkimuksen reliabiliteettia konsistenssin osalta. Haastateltavan mielipidettä teemaan kysyttiin usealla eri

kysymyksellä, jolloin haastateltavan mielipide tuli hyvin tarkasti esiin sopivalla laajuudella. Haastattelukysymykset asetettiin yksiselitteisiin ja ymmärrettäviin sanamuotoihin, jotta haastateltava ymmärsi kysymyksen vain yhdellä tavalla. Tutkimusta näillä perusteilla voidaan toistaa ja hyödyntää jatkotutkimuksissa tulosten ja luotettavuuden osalta.

Tutkimustyö tehtiin parityönä, koska tutkimusalue oli niin laaja. Oli kuitenkin havaittavissa, että opinnäytetyön laajuinen tutkimus asetti resurssirajoitukset tutkimuksen toteuttamiselle. Ajankäytöllisesti tutkimuksen toteuttamisen rajoitteita ei tutkimuksen aikana kohdattu tai havaittu, sillä tutkimukseen käytettiin se aika, että siitä saatiin tutkijoiden mielestä valmis. On kuitenkin huomioitava, että loputtomilla resursseilla olisi voitu tutkimus tehdä erilaisilla monipuolisilla tutkimusmenetelmillä; Esimerkiksi haastatella laajempaa joukkoa laajemmalla haastattelulla ja mahdollisesti tehdä havainnointia ja tutkia ohjelmistojen tuotekehityksen järjestelmistä saatavaa dataa. Saatuja tuloksia voidaan näistä rajoitteista huolimatta pitää erittäin luotettavina ja yleistettävinä. Haastatteluiden tulokset alkoivat toistaa itseään, minkä takia aineiston pohjalta voitiin tehdä yleistäviä johtopäätöksiä ilmiön kuvaamiseksi.

Tutkimuksen tavoite kuvata hukan ilmiötä laajasti eri ohjelmistojen tuotekehityksen tasoilla rajoitti tutkimusta niin, että eri tasoilta pyrittiin tutkimaan saman kaltaista kirjallisuuskatsauksen tapaista hukkaa. Ilmiö on hyvin laaja ja sitä voi laajentaa myös yrityksen muuhun toimintaan ohjelmistojen tuotekehitysprosessin ulkopuolelle. Mahdollisella tiukemmalla tutkimusalueen rajaamisella olisi voitu keskittyä tiettyihin hukan osa-alueisiin tutkimusta syvemmällä tarkkuudella. Toimeksiantajan toive kuitenkin oli, että ilmiötä käsiteltäisiin näillä tasoilla, joilla sitä on tutkimuksessa käsitelty.

Tutkimukseen haastateltavien yritysten määrittely pyrittiin tekemään niin, että tutkimukseen voitaisiin valita eri ikäisiä ohjelmistojen tuotekehitystä tekeviä yrityksiä. Tällöin tutkimuksen otanta rajoittuu suomalaisiin tutkimuksen määrittelyyn sopiviin ohjelmistojen tuotekehitystä tekeviin yrityksiin. Tutkimuksessa haastatellut henkilöt olivat myös määritelty tarkasti, joten nekään eivät rajoitu kovin tarkasti jatkotutkimuksissa, vaan voitiin osoittaa, että määrittelyyn sopiva henkilö pystyy vastaamaan kokonaisvaltaisesti haastattelukysymyksiin, jotta ilmiötä voidaan ymmärtää. Haastattelukutsuun myöntävät yritykset ovat kuitenkin olleet kiinnostuneita ja motivoituneita tutkimusta kohtaan, mikä voi tarkoittaa sitä, että hukan aiheuttamia asioita ja hukkaa ymmärretään yrityksessä paremmin kuin niissä yrityksissä, jotka vastasivat negatiivisesti haastattelukutsuun. Tutkimuskysymykset asetettiin niin, että niihin vastaaminen oli mahdollisimman neutraalia. Eli yrityksen henkilö ei voinut nostaa itseään jalustalle tai vähätellä omaa osaamistaan. Edellä mainitut asiat kasvattavat tutkimuksen luotettavuutta.

8 YHTEENVETO

Tässä luvussa kerrataan tutkimuksen tavoite, mitä tutkimuksessa tutkittiin, miten tutkimus toteutettiin ja esitetään tutkimuksen johtopäätökset. Lopuksi kerrotaan suositukset tuleville tutkimuksille.

8.1 Tutkimuksen johtopäätökset

Tutkimuksessa selvitettiin, mitä ohjelmistojen tuotekehityksen hukalla ja sen hallinnalla tarkoitetaan ja miten organisaatiot tunnistavat ja hallitsevat hukkaa. Tutkimus toteutettiin systemaattisella kirjallisuuskatsauksella ja tapaustutkimuksena. Tutkimuksen avulla saatiin kattava kuva, miten yritykset toteuttavat tuotekehitystä ja mitä hukkaa yrityksissä esiintyy ja miten tätä hallitaan ja vähennetään.

Haastateltavat tiesivät mitä hukka on ja osasivat nimetä esimerkeillä mitä hukkaa omalla tasolla tuotekehityksessä esiintyy. Lisäksi haastateltavat osasivat melko hyvin sanoa myös mitä hukkaa muilla tasoilla esiintyy. Hukan tunnistamisen lisäksi haastateltavilla oli paljon ajatuksia, miten tuotekehityksen hukkaa voisi hallita ja vähentää kaikilla tasoilla.

Haastateltavien oman näkemyksen ja haastatteluista tehtyjen havaintojen perusteella yrityksillä on kuitenkin vain vähän keinoja hukan hallintaan ja vähentämiseen. Keinot, joita yrityksillä on hukan hallintaan ja vähentämiseen tulevat ohjelmistojen tuotekehityksen menetelmistä ja ovat vakiintuneita tapoja, jotka kuuluvat yleisesti ketteriin kehitysmenetelmiin. Näitä olivat muun muassa toimittamalla nopeasti, työn määrän rajoittaminen ja työn visualisointi.

Yrityksiltä puuttui systemaattinen lähestymistapa hukan tunnistamiseen ja hallintaan ja myös vähentämiseen. Yritykset kaipasivat erityisesti portfoliotasolla portfolionhallinnan prosessin kuvaamista, jotta tällä tasolla ymmärrettäisiin paremmin, miten työtä tehdään ja tätä kautta, miten sitä voisi kehittää. Tuote- ja tuotekehitystasolla yritykset toteuttavat tekemistään tunnistettujen ketterien menetelmien mukaisesti joihin osana kuuluu hukan tunnistaminen ja vähentäminen. Kuitenkin aiheena hukasta puhutaan näilläkin tasoilla vähän eikä sitä erikseen yritetä tunnistaa ja vähentää paljoa.

Yritykset käyttävät portfolion- ja tuotehallinnan sekä tuotekehityksen toteuttamiseen erilaisia työkaluja kuten projektinhallinta, kustannus ja testaus työkaluja. Näistä työkaluista vain yritysten käyttämä Kanban taulu on yksi puhtaasti ohjelmistojen tuotekehityksen hukan hallintaan ja vähentämiseen tarkoitettu työkalu. Muita puhtaasti hukan hallintaan ja vähentämiseen tarkoitettuja työkaluja yrityksillä ei ollut käytössä. Toki myös muiden työkalujen käyttö työ tekemiseen tekee työnteosta tehokkaampaa ja tätä kautta vähentää hukkaa.

Yksi tutkimuksen havainto tuotekehityksen eri tasoista oli, että sama hukka eri tasoilla ei ole samanlaista. Esimerkiksi portfolionhallinta ja tuotekehityksen toteuttaminen on työnä hyvin erilaista, jolloin myös työssä esiintyvä saman

niminen hukka on hyvin erilaista. Tämän vuoksi hukan hallintaan ja vähentämiseen tarkoitettujen menetelmien ja työkalujen tulee olla erilaisia eri tasoilla tai niitä täytyy ainakin käyttää eri tavalla.

Toinen tutkimuksen havainto kirjallisuudesta johdetusta hukasta oli, että vaikka puhuttiin samasta hukasta kirjallisuudessa ja tutkimuksessa, niin niiden kuvaus poikkesi toisistaan eri konteksteissa (eri tasoilla). Tämän vuoksi tutkimuksen toteutus samanlaisena eri tasoilla aiheutti haasteita tutkijoille.

8.2 Tutkimuksen rajallisuus ja suositukset tuleville tutkimuksille

Tutkimuksen kohteena oli hukan tunnistaminen, riskin minimoiminen ja hallinta. Tutkimuksen laajuuden ja rajoitusten takia tutkimuksessa suoritettiin haastattelututkimus, jonka avulla pyrittiin vastaamaan tutkimuskysymykseen apukysymysten avulla. Tutkimus perustui kirjallisuuskatsauksen hukkiin, mutta sen perusteella ei pystytty tekemään täysin selvää johtopäätöstä, mikä hukka esiintyy milläkin tasolla ja millä tavalla. Tämän takia hukan syytä tutkimusta ei pystytty suorittamaan haastattelututkimuksen yhteydessä, vaan tutkimuksessa keskityttiin siihen, esiintyykö kyseistä hukkaa eri ohjelmistojen tuotekehityksen tasolla.

Tutkimus suoritettiin haastattelututkimuksena, minkä takia havainnot perustuivat haastateltavan henkilön näkemykseen ja mielipiteisiin. Tutkimus olisi voitu toteuttaa havainnoimalla, mutta se ei ollut tämän tutkimuksen laajuudessa mahdollista monelle eri yritykselle. Lisäksi havainnoinnin kautta oltaisiin voitu tarkastella helpommin syitä, miksi mikäkin hukka syntyy ja onko yrityksillä esimerkiksi yrityskulttuuriin pohjautuvia hukan hallintakeinoja konkreettisten ohjelmistojen tuotekehitysmenetelmien lisäksi. Havainnoinnin avulla olisi myös voitu havaita hukan syntymissyitä tarkemmin kuin haastatteluilla.

Tutkimuksessa ei mitattu eri hukkien ajallista vaikutusta ohjelmistojen tuotekehitykseen. Ajallista vaikutusta olisi voitu tutkia havainnoinnilla tai ohjelmistojen tuotekehitysjärjestelmän datalla. Tällä tavalla olisi ollut mahdollista järjestää hukat vaikuttavuuden mukaan järjestykseen. Lisäksi tutkimuksessa sivuttiin aihetta, jossa haastateltavalta kysyttiin, onko hänellä näkemystä toisen tason hukkiin. Laajemmassa tutkimuksessa olisi voitu mitata, miten ylemmän tason hukat vaikuttavat alemman tason tehokkuuteen ja päinvastoin.

Yritykset kehittävät ohjelmistojen tuotekehitysmenetelmiään aktiivisesti. Tämän tutkimuksen rajoitusten takia sitä ei tutkittu, miten menetelmien muutokset vaikuttavat hukkaan ja voidaanko ne nähdä hukan hallintakeinona. Lisäksi yritykset ostavat konsultointia ulkopuolisilta yrityksiltä. Konsultoinnin vaikutusta olisi mahdollista tutkia jatkotutkimuksilla.

Hukan määritelmä ohjelmistojen tuotekehityksessä on hyvin erilainen kuin tuotantoteollisuudessa, minkä takia hukka tulisi määritellä ohjelmistojen tuotekehityksen tasoille uusilla tutkimuksilla.

LÄHTEET

Ahmad, M., O., Markkula, J., & Oivo, M. 2013. Kanban in Software Development: A Systematic Literature Review. Software Engineering and Advanced Applications (SEAA), 39th EUROMICRO Conference, 2013. pp. 9-16.

Al-Baik, O. & Miller, J. 2014. Waste identification and elimination in information technology organizations. Empirical Software Engineering. Volume 19, Issue 6, pp 2019- 2061.

Anderson, D. J. 2010. Kanban: Successful Evolutionary Change for Your Technology Business. Sequim, Washington: Blue Hole Press, 2010.

Bjarnason, E., Wnuk, K. & Regnell, B. 2010. Overscoping: Reasons and consequences—A case study on decision making in software product management. Software Product Management (IWSPM), 2010 Fourth International Workshop on. IEEE, 30-39.

P. Clements, D. Garlan, R. Little, R. Nord & Stafford, J. 2003. "Documenting software architectures: views and beyond," 25th International Conference on Software Engineering, 2003. Proceedings., Portland, OR, USA, 2003, pp. 740-741.

Dahlman, M., Olsson, U. & Akillioglu, H. 2014. Applicability of Value Stream Mapping to Software Development Context. Industrial Production and Management, The Royal Institute of Technology – Stockholm. Conference Proceedings.

Denzin, N. & Lincoln, Y. 2011. The SAGE Handbook of Qualititive Research. Thousand Oaks, California: Sage.

Drury, M., Conboy, K. ja Power, K. 2012. Obstacles to Decision Making in Agile Software Development teams. The Journal of Systems and Software. Volume 85 (6), pp 1239-1254.

Eskola, J & Suoranta, J 2000: Johdatus laadulliseen tutkimukseen. Tampere: Vastapaino, 2000.

Falk, J., Kaner, A. & Nguyen, H. 1999. Testing Computer Software. Second Edition (2nd. ed.). John Wiley & Sons, Inc., USA, 1999.

Hicks, B.J. 2007. Lean Information Management: Understanding and eliminating Waste. International Journal of Information Management. Volume 27, Issue 4. Elsevier 2007, pp. 233-249.

Hirsjärvi, S. & Hurme, H. 2001. Tutkimushaastattelu: teemahaastattelun teoria ja käytäntö. Helsinki: Yliopistopaino.

Hirsjärvi, S., Remes, P. & Sajavaara, P. 2009. Tutki ja kirjoita. Helsinki: Tammi, 2009.

Huizinga, D., Kolawa, A. 2007. Automated Defect Prevention: Best Practices in Software Management. Wiley-IEEE Press, 2007.

Ikonen, M., Kettunen, P., Oza, N. ja Abrahamsson, P. 2010 Exploring the Sources of Waste in Kanban Software Development Projects. In Proceedings of the 2010 36th EUROMICRO Conference on Software Engineering and Advanced Applications (SEAA '10). IEEE Computer Society, Washington, DC, USA, pp 376-381.

Kniberg, H. & Skarin, M. 2010. Kanban and Scrum - making the most of both. C4Media, Publisher of InfoQ.com, 2010.

Kananen, J. (2008). Kvali – Kvalitatiivisen tutkimuksen teoria ja käytänteet. Jyväskylän Ammattikorkeakoulu, 2008.

Korkala, M. & Mauer, F. 2014. Waste Identification as the Means for Improving Communication in Globally Distributed Agile Software Development. The Journal of Systems and Software Volume 95, pp 122-140.

Krishna, S. T. & Sreekanth, S. 2016. Performance of Ten Software Development Process Models with Principles. IJCSET, Vol 6, Issue 6, 2016. pp 161-171.

Krishnan, V. & Ulrich, K. T. 2001. Product development decisions: A review of the literature. Management science 47 (1), 1-21.

Larman, C., & Vodde, B. 2010. Practices for Scaling Lean & Agile Development: Large, Multisite, and Offshore Product Development with Large-Scale Scrum.

Lehtonen, T, Kilamo, T, Suonsyrjä, S & Mikkonen, T. 2016, Continuous, Lean, and Wasteless: Minimizing Lead Time from Development Done to Production Use. in 2016 42th Euromicro Conference on Software Engineering and Advanced Applications (SEAA). IEEE, pp. 73-77, Euromicro Conference on Software Engineering and Advanced Applications

Leffingwell, D. 2011. Agile software requirements – Lean requirements practices for teams, programs and the enterprise. Addisson-Wesley

Lui, K. M. & Chan, K. C. 2006. Programming Task Demands. Cognitive Informatics, 2006. ICCI 2006. 5th IEEE International Conference on. IEEE, 765-770.

Mandic, V., Oivo, M., Rodriguez, P., Kuvaja, P., Kaikkonen, H. ja Turhan, B. 2010. What is Flowing Lean Software Development? Lecture Notes in Business Information Processing. 65. 72-84.

Maglyas, A., Nikula, U., & Smolander K. 2011. "What do we know about software product management? - a systematic mapping study," 2011 Fifth International Workshop on Software Product Management (IWSPM), Trento, 2011, pp. 26-35.

Middleton, P. ja Joyce, D. 2010. Lean Software Management: BBC Worldwide Case Study. IEEE Transactions on Engineering Management 59 (1), 20-32.

Mujtaba, S., Feldt, R. ja Petersen, K. 2010. Waste and Lead Time Reduction in a Software Product Customization Process with Value Stream Maps. In Proceedings of the 2010 21st Australian Software Engineering Conference (ASWEC '10). IEEE Computer Society, Washington, DC, USA, 139-148. (Mujtaba ym.. 2010)

Myers, M. D., & Newman, M. 2007. The qualitative interview in IS research: Examining the craft. Information and organization, 2007, Vol 17, pp. 2-26.

Nambisan, S. & Wilemon, D. 2000. Software development and new product development: Potentials for cross-domain knowledge sharing. IEEE Transactions on Engineering Management 47 (2), 211-220.

Ohno, T. 1988. Toyota Production System: Beyond Large-Scale Production (English translation ed.). Portland, Oregon: Productivity Press.

Okoli, C. Schabram, K. 2010. A Guide to Conducting a Systematic Literature Review of Information Systems Research. Sprouts. Working Papers on Information Systems. Sprouts

Petersen, K. 2012. A Palette of Lean Indicators To Detect Waste In Software Maintenance: A Case Study. Wohlin C. (eds) Agile Processes in Software Engineering and Extreme Programming. XP 2012. Lecture Notes in Business Information Processing, vol 111. Springer, Berlin, Heidelberg.

Pichler, R. 2010. Agile Product Management with Scrum: Creating Products That Customers Love.

Poppendieck, M. ja Poppendieck, T. 2003 Lean Development Software – An Agile Toolkit for Software Development Managers. Addison-Wesley Longman

Poppendieck, M. ja Poppendieck, T. 2006. Implementing Lean Software Development, from concept to cash. Addison-Wesley Professional, 2006.

Raffo, D., Mehta, M., Anderson, D. J., & Harmon, R. 2010. Integrating Lean principles with value based software engineering. In Technology Management for Global Economic Growth (PICMET), 2010 Proceedings of PICMET'10, IEEE, pp. 1-10.

Rao, S., Raghunathan, T., Vonderembse, M. 1997. A post-industrial paradigm: To integrate and automate manufacturing, International Journal of Production Research, Vol, 35 Issue 9, pp. 2579-2600

Rautiainen, K., von Schantz, J. & Vahaniitty, J. 2011. Supporting scaling agile with portfolio management: case Paf. com. System Sciences (HICSS), 2011 44th Hawaii International Conference on. IEEE, pp. 1-10.

Saaranen-Kauppinen, A. & Puusniekka, A. 2006. KvaliMOTV - menetelmäopetuksen tietovaranto [pdf-verkkojulkaisu]. Tampere: Yhteiskuntatieteellinen. Viitattu 10.3.2020

Shalloway, A., Beaver, G. & Trott, J. 2010. Lean-Agile software development: achiving enterprise agility. Adisson-Wesley.

Vadivu, M. S. & Tapaskar, V. 2011. Enacted Software Development Process Based On Agile And Agent Methodologies. International Journal of Engineering Science and Technology, Vol 1, Issue 3, pp. 8019-8029.

Van De Weerd, I., Brinkkemper, S., Nieuwenhuis, R., Versendaal, J. & Bijlsma, L. 2006. On the creation of a reference framework for software product management: Validation and tool support. Software Product Management, 2006. IWSPM'06. International Workshop on. IEEE, pp. 3-12.

Vashisht, V. 2014 Enhancing Software Process Management through Control Charts. Journal of Software Engineering and Applications, 2014, 7, 87-93

Vähäniitty, J. & Rautiainen, K. 2005. Towards an approach for managing the development portfolio in small product-oriented software companies. System Sciences, 2005. HICSS'05. Proceedings of the 38th Annual Hawaii International Conference on. IEEE, 314c.

Wang, X., Conboy, K. ja Cawley, O. 2012. "Leagile" Software Development: An Experience Report Analysis of the Application of Lean Approaches in Agile Software Development. Journal of Systems and Software. Volume 85, pp 1287–1299.

Wallin, C., Ekdahl, F. & Larsson, S. 2002. Integrating business and software development models. IEEE Software 19 (6), 28-33.

Womack, J.P., Jones, D.T. and Roos, D. 1990. The Machine that Changed the World: The Story of Lean Production. HarperCollins Publishers, New York, USA.

Womack, J. P., & Jones, D. T. 2003. Lean thinking: Banish waste and create wealth in your corporation. New York: Free Press.

LIITE 1 HAASTATTELUKUTSU

”Hei (yrityksen edustaja)

Teemme työparini kanssa pro-gradu tutkielmaa, jonka aihe on ohjelmistojen tuotekehityksen hukka. Olisiko teidän yrityksellä kiinnostusta ja aikaa osallistua haastattelututkimukseen aiheesta?

Lyhyt kuvaus aiheesta:

Ohjelmistojen tuotekehitys lähtee tuotteen tai tuotteen ominaisuuden tuomisesta muiden yrityksen henkilöiden tietoisuuteen ja loppuu tuotteen tai ominaisuuden elinkaaren lopettamiseen. Tällä välillä tapahtuu erilaisia prosesseja portfoliotasolla, tuotetasolla ja tuotekehitystasolla. Prosessien aikana tehdään operaatioita, jotka tuottavat arvoa kehitettävälle tuotteelle. Jotkin operaatiot ja aliprosessit ovat kuitenkin toimintaa, jotka eivät tuo arvoa tuotteelle. Näitä operaatiota kutsutaan hukaksi. Hukkaa voi tapahtua kaikilla tasoilla

Gradun kuvaus:

Gradussa tarkastellaan ohjelmistojen tuotekehityksen hukkaa kaikilla kolmella tasolla. Kirjallisuuskatsauksen tueksi, vertailuksi ja uuden tiedon luomiseksi teemme haastatteluja eri ohjelmistojen tuotekehitystä tekeville yrityksille kaikilla kolmella eri tasolla. Tarkoituksena on tunnistaa olemassa olevan hukan käsitteitä ja löytää uusia hukkia. Sen lisäksi tavoitteena on saada tietoa siitä, miten yritykset tunnistavat, minimoivat ja välttävät hukkaa aiheuttavia prosesseja ja tilanteita ohjelmistojen tuotekehityksessä.

Haastattelut:

Haastattelut toteutetaan paikan päällä ja yrityksestä haastatellaan kolmea henkilöä, yhtä kultakin tasolta. Haastattelujen tuloksia käsitellään anonyymisti niin, että ulkopuolinen henkilö ei voi tunnistaa yritystä tai haastateltavia henkilöitä. Tulokset käsittelevät ainoastaan yrityksen tuotekehityksen hukkaa koskevia asioita.

- Portfoliotaso (Tuotesalkun hallinta ja ideoiden eteenpäinvienti tuotetasolle)
- Tuotetaso (Yksittäisten tuotteiden ja ominaisuuksien eteenpäinvienti tuotekehitystasolle)
- Tuotekehitystaso (Tuotekehitys, testaaminen ja julkaisu)

Olisiko teidän mahdollista osallistua haastatteluihin esimerkiksi **(ajankohta)**. Haastattelut voivat olla mihin aikaan vaan päivästä. Haastattelu kestää noin tunnin / haastattelu ja henkilöitä haastatellaan erikseen. Haastatteluun ei tarvitse

valmistautua millään tietyllä tavalla. Vastaamista nopeuttaa kuitenkin se, että haastattelija miettii etukäteen tuotekehityksen oman tason prosesseja ja tehtäviä.

Haastattelutilaksi sopii neuvotteluhuone. Haastattelut voidaan tehdä myös toimiston tiloissa.

Mitä hyötyä gradujen haastatteluun osallistumisesta on?

Yritys saa kirjallisuuteen ja muiden yrityksien tuloksiin perustuvaa tietoa heidän tuotekehitysprosessin hukista ja mahdollisuuksista minimoida tai välttää niitä.

Miltä kuulostaa? Olisiko teillä mahdollisuutta osallistua haastateltavana yrityksenä graduun?”