Atte Tuomisto

SUUNNITTELUTIETEELLINEN TUTKIMUS – MITÄ ASIAKKAAN TULEE TIETÄÄ KETTERÄSTÄ OHJELMISTOKEHITYSPROJEKTISTA?

JYVÄSKYLÄN YLIOPISTO
INFORMAATIOTEKNOLOGIAN TIEDEKUNTA
2019

TIIVISTELMÄ

Tuomisto, Atte
Suunnittelutieteellinen tutkimus – Mitä asiakkaan tulee tietää ketterästä ohjelmistokehitysprojektista?
Jyväskylä: Jyväskylän yliopisto, 2019, 59 s.
Tietojärjestelmätiede, pro gradu -tutkielma
Ohjaaja: Seppänen, Ville

Ketterä ohjelmistokehitys on jo melko vakiintunut tapa tuottaa tietojärjestelmiä. Tästä syystä jokaisen ketterään ohjelmistokehitysprojektiin osallistuvan tulisi tietää, mitä ketterällä ohjelmistokehityksellä tarkoitetaan ja mitä siihen kuuluu. Projektiin osallistuvia osapuolia ovat kehitystiimi ja kehitettävän tietojärjestelmän tai muun palvelun tilaaja eli asiakas. Jotta asiakkaan edustajat voivat osallistua projektiin mahdollisimman hyvin, on heillä oltava riittävä tietotaito ketterästä ohjelmistokehitysprojektista. Tässä tutkielmassa tutkittiin, mitä asioita projektiin osallistuvan asiakkaan tulisi tietää ketterästä ohjelmistokehitysprojektista, jotta asiakas saisi riittävän tietotaidon kyseisestä aiheesta. Tähän vastaamiseksi ensin tutkielmassa selvitettiin, mikä on ketterän ohjelmistokehitysprojektin tyypillinen kehityskulku ja vaiheet sekä mitä tyypillisimpiä elementtejä ketterään ohjelmistokehitysprojektiin kuuluu. Näiden selvityksien pohjalta tutkielmassa toteutettiin suunnittelutieteellisen tutkimusmenetelmän vaiheita noudattaen artefakti, jolla pyrittiin koostamaan ne aiheeseen liittyvät asiat, joita asiakkaan tulisi tietää ketterästä ohjelmistokehitysprojektista. Tämän tuloksena luotiin asiakkaalle suunnattu ketterän ohjelmistokehitysprojektin prosessikuvaus, joka esittää asiakkaan kannalta olennaisimmat asiat helposti ymmärrettävään ja yksinkertaiseen visuaaliseen ja kirjalliseen muotoon. Toteutettu prosessikuvaus arvioitiin asiantuntijahaastatteluilla, joista saatujen palautteiden perusteella prosessikuvausta kehitettiin edelleen. Tutkimuksen lopullisena tuloksena muodostettiin prosessikuvaus, jonka avulla projektiin osallistuvat osapuolet pystyvät perehdyttämään asiakkaan ketterään ohjelmistokehitysprojektiin ja varmistamaan asiakkaan riittävän tietotaidon sekä yhteisen ymmärryksen aiheesta ja siihen liittyvistä käsitteistä.

Asiasanat: ketterä ohjelmistokehitys, projekti, asiakas, suunnittelutieteellinen tutkimus

ABSTRACT

Tuomisto, Atte
Design Science Research - What Should a Customer Know About an Agile Software Development Project?
Jyväskylä: University of Jyväskylä, 2019, 59 pp.
Information Systems, Master's Thesis
Supervisor: Seppänen, Ville

Agile software development is already a well-established way of producing information systems. For this reason, everyone involved in agile software development projects should know what is meant by agile software development and what it involves. The parties involved in the project are the development team and the customer. In order to involve in the project, the customer must have sufficient knowledge of an agile software development project. The aim for this thesis was to find out, what kind of things customer should know about an agile software development project, in order to have sufficient knowledge on the subject. To answer this, an analysis was made, that explored the typical development process and stages of an agile software development project and the most typical elements of the topic. Based on these findings, an artefact was implemented by following the steps of the design science research method. The artefact sought to compile the related issues that a customer should know about an agile software development project. As a result, a customer-driven agile software development project process overview was created that presents the most relevant to the customer in an easy-to-understand, simple visual and written format. The implemented process description was evaluated by interviews, on the basis of which the process description was further developed. The final result of the study was a process overview that enables the project partners to familiarize customer with the agile software development project and to ensure the customer's sufficient knowledge of the subject and related concepts.

Keywords: agile software development, project, customer, design science

KUVIOT

TAULUKOT

SISÄLLYS

1 JOHDANTO

Tietojärjestelmiä kehitetään niitä varten perustetuissa projekteissa ja näitä projekteja sekä siihen liittyvää ohjelmistokehitystä toteutetaan erilaisin projektinhallinnallisin keinoin. Ohjelmistokehityksen saralla jo melkeinpä vakiintuneeksi ajattelutavaksi ja projektinhallintakeinoksi on muodostunut ketterä ohjelmistokehitys, jota projektissa toteutetaan tiettyjä siihen kuuluvia pääperiaatteita ja käytäntöjä noudattaen (Dybå & Dingsøyr, 2008). Ketterän ohjelmistokehityksen suosion johdosta jokaisen projektiin osallistuvan tulisi tietää mitä ketterällä ohjelmistokehityksellä tarkoitetaan ja mitä asioita siihen kuuluu (Conboy, 2009).

Yksi projektiin osallistuvista osapuolista on asiakas. Nasirin ja Sahibuddinin (2011), Chowin ja Caon (2008) sekä Ahimbisibwen ym. (2015) mukaan projektin onnistumiseen vaikuttaa olennaisesti se, kuinka sitoutunut asiakas on projektiin. Misra, Kumar ja Kumar (2009) ovat myös samalla kannalla, ja he nostavat lisäksi asiakkaan yhteistyön ja asiakkaan tyytyväisyyden tärkeiksi tekijöiksi. Conboyn (2009) mukaan projektille on myös ensiarvoisen tärkeää, että asiakas ymmärtää yleisesti mitä ketterä ohjelmistokehitys on ja mitä siihen kuuluu.

Asiakkaan sitouttaminen projektiin ei ole kuitenkaan täysin helppoa ja Conboy (2009) kuvaileekin yhtenä sitoutumiseen vaikuttavista ongelmista asiakkaan huonoa tietotaitoa ketterästä ohjelmistokehityksestä. Samoin Ahimbisibwen ym. (2015) esittävät analyysissaan, että projektin onnistumiseen vaikuttaa olennaisesti myös se, kuinka hyvin asiakasta koulutetaan projektiin liittyvissä asioissa ja mikä on asiakkaan aikaisempi kokemus ketteristä ohjelmistokehitysprojekteista.

Tutkimukset eivät tarjoa kattavasti tietoa siitä, miten asiakasta pystyttäisiin sitouttamaan projektiin paremmin. Käsiteltyjen tutkimusartikkelien pohjalta voidaan kuitenkin melko varmasti todeta, että asiakas on erittäin olennaisessa osassa projektin onnistumisen kannalta ja että asiakkaan riittävä tietotaito ketterästä ohjelmistokehitysprojektista on ensiarvoisen tärkeää. Oletettavasti nämä tekijät liittyvät hyvin vahvasti toisiinsa, eli asiakas sitoutuu ja osallistuu ketterään ohjelmistokehitysprojektiin paremmin, jos asiakas on riittävän hyvin koulutettu sen käytänteisiin.

Tästä syystä on tärkeää selvittää, mitkä ovat sellaisia asioita, joita asiakkaan olisi syytä tietää ketterästä ohjelmistokehitysprojektista asiakkaan riittävän ymmärtämisen takaamiseksi ja myös se, miten tätä tietoa voitaisiin asiakkaalle esittää. Tämä tutkielma pyrkii vastaamaan tähän ongelmaan. Tästä tutkimusongelmasta voidaankin johtaa seuraava tutkimuskysymys:

- Mitä asioita asiakkaan on tärkeää tietää osallistuessaan ketterään ohjelmistokehitysprojektiin?

Jotta tähän tutkimuskysymykseen voidaan vastata, on tutkielmalle vielä laadittu kaksi seuraavaa taustoittavaa tutkimuskysymystä:

- Mitä vaiheita tyypilliseen ketterään ohjelmistokehitysprojektiin kuuluu?
- Mitä tyypillisimpiä elementtejä ketterä ohjelmistokehitysprojekti pitää sisällään?

Tämän tutkielman rakenne on seuraava. Johdannossa esitetään tutkielman aihe ja tutkielman tutkimusongelma sekä sen pohjalta määritellyt tutkielman tutkimuskysymykset. Tutkielman toisessa luvussa käydään läpi tutkielman kannalta keskeisiä käsitteitä, joita tutkielman myöhemmässä vaiheessa hyödynnetään. Kolmannessa luvussa vastataan tutkielman kahteen taustoittavaan tutkimuskysymykseen. Ensin luvussa käydään läpi ketterän ohjelmistokehitysprojektin prosessikuvauksia ja tämän analyysin pohjalta muodostetaan ketterän ohjelmistokehitysprojektin tyypilliset vaiheet. Toiseksi luvussa esitetään toteutettu analyysi, jossa koostettiin lista ketterän ohjelmistokehitysprojektiin liittyvistä tyypillisimmistä elementeistä. Nämä kyseisessä luvussa toteutetut analyysit vastaavat tutkielman taustoittaviin tutkimuskysymyksiin ja pohjaavat siten tutkielman varsinaisen tutkimuskysymykseen vastaamista. Tämän jälkeen neljännessä luvussa käydään yleisesti läpi suunnittelutieteellistä tutkimusmenetelmää ja miten tätä menetelmää toteutetaan tässä tutkielmassa. Viidennessä luvussa suunnitellaan, kehitetään ja arvioidaan suunnittelutieteellisen tutkimusmenetelmän mukaisesti artefakti, joka pyrkii vastaamaan tutkielman varsinaiseen tutkimusongelmaan ja tutkimuskysymykseen. Kuudennessa luvussa esitellään lopullinen versio luodusta artefaktista eli asiakkaalle suunnatusta ketterän ohjelmistokehitysprojektin prosessikuvauksesta. Lisäksi luvussa demonstroidaan, miten prosessikuvausta voidaan hyödyntää käytännössä ja mitä prosessikuvaus kokonaisuudessaan pitää sisällään. Seitsemännessä luvussa pohditaan tutkielmassa toteutetun prosessikuvauksen tavoitteiden onnistumista, yleisesti tutkielman tulosten merkitystä sekä sen rajoitteita ja mahdollisia jatkotutkimusaiheita. Lopuksi seitsemännessä luvussa on koottu yhteen tutkielman pääseikat ja toteutettu tutkimusprosessi ja sekä sen tulokset.

2 TIETOJÄRJESTELMÄKEHITYS JA KETTERÄ OHJELMISTOKEHITYS

Tässä luvussa käydään läpi tutkielman kannalta tärkeimmät käsitteet. Nämä esiteltävät käsitteet tukevat varsinaiseen tutkimusongelmaan vastaamista. Luvun ensimmäisessä alaluvussa esitellään yleisesti tietojärjestelmäkehitysprojekteja ja liittyviä muita käsitteitä. Toisessa alaluvussa käydään läpi ketterää ohjelmistokehittämistä ja siihen liittyviä muita käsitteitä.

2.1 Tietojärjestelmäkehitys

Tämän tutkielman kannalta on tärkeää käydä läpi yleisellä tasolla tietojärjestelmien kehittämistä varten perustetun projektin käsitteistöä. Seuraavaksi käsitellään projektin tavoitetta, siihen osallistuvia osapuolia ja kehitettävän tuotteen visiota. Jatkossa tässä tutkielmassa, käytetään tietojärjestelmäkehityksestä myös termiä ohjelmistokehitys.

Tietojärjestelmiä kehitetään niitä varten perustetuissa projekteissa. Näiden projektien tarkoituksena on projektiin osallistuvien tahojen toimesta tuottaa tietojärjestelmä tai jokin muu tietotekninen kokonaisuus. Tätä projektissa tavoiteltavaa lopputulosta voidaan kutsua esimerkiksi tuotteeksi. Tuotteella voidaan tässä kontekstissa tarkoittaa mitä tahansa ohjelmistokehityksen avulla toteutettavaa tuotosta tai muutosta olemassa olevaan tuotokseen. Näitä lopputuotoksia voi olla esimerkiksi uusi mobiilisovellus, tai vaikkapa olemassa olevan tietojärjestelmän jatkokehitys (Vähäniitty, 2012; Wiegers & Beatty, 2013.)

Perustetussa projektissa on monia tahoja, jotka pyrkivät edistämään projektin etenemistä ja projektin lopputulokseen pääsemistä. Wiegers ja Beatty (2013) jakavat kirjassaan henkilöt kahteen eri osapuoleen: toimeksiantajaan ja toimittajaan. He kuvailevat toimeksiantajan, eli toisin sanoen asiakkaan, koostuvan yleensä jonkin yrityksen edustajista, jotka ovat tilanneet varsinaisen tietojärjestelmäkehityksen. Tällä asiakasyrityksellä on jokin ongelma, jota projektin lopputuloksella pyritään korjaamaan. Wiegers ja Beatty (2013) kuvailevat toisen

osapuolen, eli toimittajan koostuvan, varsinaisesta kehitystiimistä, joka toteuttaa projektin tuotteen ja pyrkii siten ratkaisemaan toimeksiantajan ongelman. Jatkossa näitä eri osapuolia kutsutaan termeillä asiakas ja kehitystiimi.

Kuten todettiin, on asiakkaalla jokin ongelma, joka pyritään ratkaisemaan kehitettävän tietojärjestelmän tai olemassa olevaan tietojärjestelmään toteutettavien muutosten avulla. Kun asiakas tilaa tietojärjestelmäkehitystä on asiakkaalla jokin visio varsinaisesta lopputuloksesta, jolla mahdollisesti asiakkaan ongelma voitaisiin ratkaista (Wiegers & Beatty, 2013). Tämä tuotteen visio voidaan määritellä esimerkiksi seuraavilla tavoilla. Vähäniitty (2012) kuvailee tuotteen vision olevan pitkäaikainen näkemys, joka määrittelee tuotteeseen kehitettävät asiat ja niiden prioriteettijärjestyksen ottaen huomioon kehitettävän tuotteen liiketoiminnalliset mahdollisuudet. Wiegers ja Beatty (2013) kuvailevat tuotteen vision sisältävän tuotteen liiketoiminnalliset vaatimukset, joiden pohjalta tuote toteutetaan. Samoilla linjoilla ovat myös Vlaanderen ym. (2011) tutkimuksessaan. Yleisesti visio voidaankin siis nähdä silloisena käsityksenä siitä, millainen tavoiteltavan tuotteen lopputulos on, kenelle se on osoitettu, ja mitä arvoa se pyrkii tuottamaan (Vlaanderen ym., 2011; Vähäniitty, 2012; Wiegers & Beatty, 2013).

Pelkän vision pohjalta on kuitenkin hyvin vaikeaa toteuttaa mitään konkreettisia työtehtäviä projektille ja siten myös hyvin haastavaa käsittää, mitä eri asioita tuotteeseen olisi toteutettava. Tästä syystä visiota pilkotaan pienempiin osiin ja pyritään konkreettisesti määrittämään mitä kaikkea projektissa tarvitsee tehdä (Vlaanderen ym., 2011.)

Taulukkoon 1 on koostettu Vlaanderenin ym. (2011) ja Vähäniityn (2012) sekä Wiegersin ja Beattyn (2013) määritelmien pohjalta sovellettu neljä vaatimuksen tasoa sekä, sekä konkreettiset esimerkit siitä, mitä kukin taso mahdollisesti voi sisältää. Taulukossa esitetystä visiosta on korkein vaatimuksen taso ja tähän visioon pohjautuen pyritään jakamaan vaatimukset tarkemmiksi osakokonaisuuksiksi, joista edelleen se on jaettu toiminnallisuuksiin ja vielä tehtäviin. Tällä tavoin projektissa pystytään helpommin hahmottamaan tuotteen kokonaisuutta ja jakamaan työtehtäviä kehitystiimille ja siten toteuttamaan varsinaista ohjelmistokehitystä projektissa (Vlaanderen ym., 2011; Wiegers & Beatty, 2013).

TAULUKKO 1 Vaatimusten tasot

Vaatimuksen taso	**Kuvaus**	**Esimerkki**	**Vastaavat englanninkieliset termit**
Visio	Jokin ongelma tai edellytys, jonka takia tietojärjestelmää ollaan kehittämässä.	Tietojärjestelmä, johon työntekijät voivat kirjata työtuntinsa.	vision
Osakokonaisuudet	Visiosta pilkottuja laajoja osakokonaisuuksia.	Tietojärjestelmässä on oltava käyttäjähallinta.	theme, concept, epic
Toiminnallisuudet	Osakokokonaisuuksista pilkottuja tarkempia toiminnallisuuksia.	Käyttäjän tulee pystyä rekisteröitymään järjestelmään.	feature, story
Tehtävät	Toiminnallisuuksista pilkottuja tarkempia tehtäviä.	Tehdään rekisteröitymissivusto. Luodaan tietokantataulu käyttäjille.	task, item

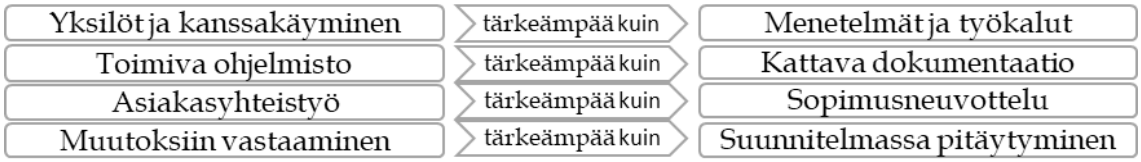
2.2 Ketterä ohjelmistokehitys

Ohjelmistokehittämiseen liittyvää projektinhallintaa on vuosien varrella toteutettu monin eri tavoin. Nykyään melkeinpä jo vakiintunut tapa toteuttaa ohjelmistokehitysprojektiin liittyvää projektinhallintaa on niin sanottu ketterä ohjelmistokehitys. Ketterä ohjelmistokehitys on koko 2000-luvun kasvattanut suosiotaan ja se on siten myös kiinnostava aihe tutkimuksen saralla. Ketterä ohjelmistokehitys on alun perin kuitenkin lähtöisin työelämästä IT-alan asiantuntijoiden toimesta, joten se ei varsinaisesti pohjaudu tieteeseen. Tästä syystä ilmiön määrittely ei ole täysin yksiselitteinen, mutta ilmiötä kuvaamaan on kuitenkin pyritty luomaan yleisiä määrityksiä ja yleistä käsitystä siihen kuuluvista periaatteista. Yksi käytetty määritelmä tietojärjestelmätieteen tutkimuskirjallisuudessa on seuraava Conboyn (2009) määritelmä:

> Tietojärjestelmäkehityksen menettelytapa luontaiselle ja jatkuvalle valmiudelle ja muutoksille vastaamiselle sekä muutoksista oppimiselle, hyödyntäen tapaan liittyviä osia ja niiden välisiä suhteita, samalla tuottaen asiakasarvoa (taloudellisuus, laatu ja yksinkertaisuus) (Conboy, 2009, s. 340).

Ketterä ohjelmistokehitys nähdäänkin ennen kaikkea tietynlaisena ajattelutapana, jolla on tietyt prioriteetit, joilla ohjelmistokehitystä hallitaan ja toteutetaan (Abrahamsson ym., 2017). Ketterä ohjelmistokehityksen aate on lähtöisin, voidaan tulkita alkaneeksi vuonna 1999 kun Kent Beck kirjassaan ja kirjoituksissaan (Beck, 1999; Beck & Gamma, 2000) avasi Extreme Programming -ohjelmistokehitysmenetelmän ja siihen kuuluvan ajattelutavan ja minkä myötä vuonna 2001 ”Agile Manifesto”:ssa (Beck ym., 2001) määriteltiin ajattelutavalle sen tärkeimmät arvot alan merkittävimpien henkilöiden toimesta (Abrahamsson ym., 2017; Lee & Xia, 2010). Tällä julistuksella on ollut merkittävä vaikutus ketterän ohjelmistokehityksen käsitteellistämiseen ja jatkokehitykseen (Abrahamsson ym., 2017; Dybå & Dingsøyr, 2008). Tätä julistusta voidaankin yleisesti ottaen tulkita ketterän ohjelmistokehityksen alkamiseksi, vaikkakin samoja periaatteita on käytetty jo paljon kauemmin. Se sai kuitenkin aikaan alalla monia suuria muutoksia ja näin on muovannut ohjelmistokehityksen prosesseja melko laajalti. (Dingsøyr ym., 2012).

Kuviossa 1 on esitettynä manifestissa (Beck ym., 2001) esitetyt ketterän ohjelmistokehityksen neljä pääpiirrettä. Nämä pääpiirteet osoittavat, miten manifestin kirjoittajat arvottivat erilaiset käsitteet, eli mitä piirteitä he pitivät tärkeämpinä verrattuna toisiin piirteisiin. Ketterän ohjelmistokehityksen pääpiirteinä onkin, että yksilöt ja kanssakäyminen, toimiva ohjelmisto, asiakasyhteistyö sekä muutoksiin vastaaminen ovat tärkeimmässä asemassa. Menetelmät ja työkalut, kattava dokumentaatio, sopimusneuvottelu sekä suunnitelmassa pitäytyminen ovat vastavuoroisesti vähemmän arvossa tässä ajatusmallissa (Abrahamsson ym., 2017.)



KUVIO 1 Ketterän ohjelmistokehityksen pääpiirteet

Ketteryyttä voidaankin käsittää tietynlaisena ajattelutapana ja sitä voidaan kuvailla kykynä olla valpas, joustava, nopea ja kyvykäs. Ketteryyden ytimessä on se, että pyritään poistamaan mahdollisimman paljon kankeita toimintatapoja ja pyritään kohti sellaista työntekoa ja työn järjestelyä, jolla pystytään reagoimaan muuttuviin tilanteisiin hyvinkin nopeasti korostaen samalla ihmisten välistä vuorovaikutusta (Erickson ym., 2005.)

2.2.1 Ketterä ja suunnittelulähtöinen ohjelmistokehitys

Ketterä ohjelmistokehitys ei suinkaan ole ainut ajattelumalli, jonka pohjalta ohjelmistokehitystä voidaan toteuttaa. Erityisesti ennen ketterää ohjelmistokehitystä, sitä toteutettiin hyvin usein suunnitelmalähtöisesti. Tätä ajattelutapaa kutsutaankin hyvin usein traditionaaliseksi ohjelmistokehitykseksi. Ketterä ohjelmistokehitys varsinaisesti kehitettiinkin vastauksena suunnitelmalähtöisen ajattelutavan myötä tulleisiin ongelmiin (Abrahamsson ym., 2017).

Traditionaalinen ohjelmistokehitys eroaa ketterästä ohjelmistokehityksestä monella tavalla. Siinä keskitytään kattavaan suunnitteluun projektin alussa ja tällä suunnittelulla sekä siihen liittyvällä määrittelyllä ja sen dokumentoinnilla voidaan kehitystä toteuttaa hyvinkin lineaarisesti pohjautuen tehtyyn suunnitelmaan. (Dybå & Dingsøyr, 2008). McCauley (2001) kuvaakin tätä ajattelutapaa siten, että ne projektille asetetut vaatimukset, joita projektissa kehitetään, ovat lukittuja ja pitäviä. Tähän ajattelutapaan on muodostettu myös lukuisia eri menetelmiä, joista varmaankin tunnetuin on vesiputousmalli (eng. waterfall model). Taulukossa 2 on esitetty Nerurin, Mahapatran ja Mangalarajin (2005) tutkimuksessa esittämät erot ketterän ja traditionaalisen ajattelumallien välillä.

TAULUKKO 2 Ketterän ja traditionaalisen ohjelmistokehityksen erot

	Traditionaalinen	Ketterä
Keskeinen näkemys	Järjestelmät ovat täysin määritettävissä, ennustettavissa ja ne voidaan rakentaa huolellisella ja laajalla suunnittelulla.	Pienet tiimit voivat kehittää laadukkaita, mukautuvia ohjelmistoja käyttämällä jatkuvan suunnittelun parantamisen ja testauksen periaatteita, jotka perustuvat nopeaan palautteeseen ja muutokseen.
Ohjaus	Prosessikeskeinen	Ihmiskeskeinen
Johtamistapa	Komento ja kontrolli	Johtajuus ja yhteistyö
Tietämyksenhallinta	Täsmällinen	Induktioon perustuva
Roolin määritys	Yksilöllinen erityisosaaminen	Itseorganisoituvat joukkueet – kannustavat roolien vaihdettavuuteen
Kommunikointi	Muodollinen	Arkikielinen

(jatkuu)

TAULUKKO 2 (jatkuu)

Asiakkaan rooli	Tärkeä	Kriittinen
Projektin tahti	Tehtävä- tai aktiviteettivetoinen	Tuotteen ominaisuusvetoinen
Toivottu organisaatiorakenne	Muodollinen ja byrokraattinen	Joustava ja osallistava – sosiaaliseen yhteistyöhön perustuva

Nerur, Mahapatra ja Mangalaraj (2005) nostavat tutkimuksessaan näiden kahden ajattelutavan selvimpänä erona sen, että traditionaalinen ohjelmistokehitys pyrkii painottamaan suunnittelun projektin alkuun ja pyrkii formaaliin dokumentointiin ja kanssakäymiseen, kun taas ketterä ajattelumalli pyrkii mukautumaan tilanteeseen, jossa korostetaan kanssakäymistä ja jatkuvaa suunnittelua.

Vaikkakin joissain tapauksissa suunnittelulähtöinen tapa voi olla sopiva, ei se yleisesti ottaen vastaa oikean maailman realiteetteja, joissa kehitettävän tuotteen vaatimukset muuttuvat alati (Abrahamsson ym., 2017). Traditionaalisen ohjelmistokehitystapa on koettukin kömpelöksi tavaksi kehittää ohjelmistoja ja se ei vastaa riittävästi ohjelmistokehityksen luonteeseen. Muuttuvat tarpeet ja asiakaslähtöinen kehittäminen on huomattu ensisijaisen tärkeiksi projektin kehityksen kannalta. Ketteryyteen kuuluukin, että muutoksia jopa odotetaan ja niihin pyritään olemaan aina valmiita. Ketterä ohjelmistokehitys ja menetelmät ovatkin kehitetty vastaamaan juuri niihin ongelmakohtiin, joihin suunnitelmapohjainen ohjelmistokehittäminen ei pysty vastaamaan.

2.2.2 Ketterät ohjelmistokehitysmenetelmät

Ketterään ohjelmistokehitykseen liittyvät olennaisesti myös siihen pohjautuvat ketterät ohjelmistokehitysmenetelmät. Tämän arvopohjan ja periaatteiden pohjalta on kehitetty projektinhallinnan menetelmiä, jotka ottavat tarkemmin kantaa projektin vaiheisiin, tapahtumiin ja yleiseen tapaan toimia projektissa. (Abrahamsson ym., 2017)

Nämä kehitetyt menetelmät nojaavat hyvin vahvasti iteratiiviseen kehitystapaan. Iteratiivisuudella tarkoitetaan projektin vaiheiden toteuttamista sykleissä, joissa projektia toteutetaan pala palalta pienemmissä osissa toistaen samoja työvaiheita useamman kerran. Iteratiivisuus toteutuu yleensä siten, että ohjelmistokehitys tapahtuu kehityserissä tietyn ajanjakson ajan, jossa pyritään toteuttamaan jokin kokonaisuus kehitettävästä tietojärjestelmästä tietyn ajanjakson aikana. (Levy ym., 2015; Wiegers & Beatty, 2013.) Iteratiivisuuden lisäksi ketterät menetelmät sisältävät hyvin paljon muita tapahtumia ja toteutustapoja, joita käydään tarkemmin läpi myöhemmin tässä tutkielmassa.

Erilaisia ketteriä menetelmiä on kehitetty lukuisia. Yhdysvaltalaisen teknologiayrityksen VersionOnen vuosittain toteuttaman kyselyn perusteella selvästi suosituin ketterä ohjelmistokehitysmenetelmä on Scrum. Kyselyyn vastanneista 56 prosenttia oli osallistunut projektiin, jossa oli käytössä Scrum. Tämän lisäksi vastanneista kahdeksan prosenttia osallistui Scrumia ja Kanbania sekä kuusi prosenttia Scumia ja XP:ä yhdistäviin projekteihin. Vastanneista 14

prosenttia käyttää määrittelemättömiä yhdistelmämalleja ja loput vastanneista käyttää jotain muita pienempiä malleja. Tämän kyselyn pohjalta voidaankin huomata, että Scrum on selkeästi yleisin ketterä ohjelmistokehitysmenetelmä. Toisaalta huomion arvoista on se, että merkittävä osuus kyselyyn vastanneista hyödynsi ketteriä menetelmiä soveltaen useampaa ketterää ohjelmistokehitysmenetelmää.

3 KETTERÄN OHJELMISTOKEHITYSPROJEKTIN VAIHEET JA ELEMENTIT

Tässä luvussa muodostetaan käsitys ketterän ohjelmistokehitysprojektin yleisestä prosessista ja vaiheista, sekä kehitysprojektiin olennaisesti kuuluvista elementeistä toteuttamalla teoriakatsaus aiheeseen. Tämän luvun tarkoituksena on vastata tutkielman kahteen taustoittavaan tutkimuskysymykseen. Ensimmäisessä alaluvussa tarkastellaan olemassa olevia ketterän ohjelmistokehityksen prosessikuvauksia ja pyritään koostamaan olemassa olevien kuvausten pohjalta tyypilliset projektin etenemisprosessin vaiheet. Toisessa alaluvussa on toteutettu kirjallisuuskatsaus, jossa koostetaan tutkimuksissa esiintyneet yleisimmät ketterän ohjelmistokehityksen liittyvät elementit.

3.1 Yleiskuva ketterän ohjelmistokehitysprojektin tyypillisestä prosessista ja vaiheista

Tässä alaluvussa pyritään muodostamaan käsitys ketterälle ohjelmistokehitysprojektille tyypillisestä prosessista ja sen vaiheista. Ensin toteutetaan katsaus olemassa oleviin prosessikuvauksiin ja pyritään niistä luomaan yleiskuva ketterän ohjelmistokehityksen prosessista. Tämän jälkeen toteutetaan näiden prosessikuvausten pohjalta yleistys, jossa on koostettu vaiheet ketterän ohjelmistokehitysprojektin tyypilliselle prosessille.

Ketterän ohjelmistokehitysprojektin yleiskuvan muodostamiseen on tärkeää tarkastella siihen liittyviä olemassa olevia prosessikuvauksia, jotka pyrkivät esittämään sen elementit. Terminä prosessikuvaus ja siihen liittyvä prosessimalli eivät ole täysin yksiselitteisiä käsitteitä. Tässä tutkielmassa käytetään Rollandin (1998) tutkimuksen esittelemää määrittelyä prosessimallista ja prosessikuvauksesta. Rolland (1998) kuvailee prosessimallin olevan kuvaus siitä, miten jokin prosessi tai prosessit yhdessä muodostavat toimivan kokonaisuuden. Rollandin mukaan prosessimallien tarkoituksena on selittää tärkeimmät tekijät kuvatusta ilmiöstä ja siitä mitä, milloin ja miksi jotain tapahtuu. Rolland

(1998) tutkimuksessaan kuvailee, että prosessimallilla on kolme tavoitetta. Niiden tulee olla kuvaavia, eli ne esittävät, mitä prosessissa tapahtuu. Niiden tulee olla määrääviä, eli ne kuvaavat halutut prosessit ja miten niitä tulisi, kuuluisi tai voitaisiin toteuttaa. Niiden on oltava selittäviä, eli tuottaa tieto ja käsitys siitä, mitä kullakin prosessin osasella tarkoitetaan. Tähän verrattuna prosessikuvaus (eng. process overview tai process description) on löyhempi kuvaus prosessista, ja se kuvailee prosessia ja siihen kuuluvia asioita vain yleispiirteisesti (Rolland, 1998). Tutkimalla erilaisia prosessikuvauksia pystytään käsittämään yleisesti sitä, miten yleisesti ketterän ohjelmistokehityksen projektit etenevät ja tällä tavoin se auttaa tässä tutkielmassa yleistyksen toteuttamista.

Prosessikuvausten valinta toteutettiin seuraavasti. Ensimmäisiksi tarkasteltavaksi prosessikuvauksiksi valittiin Scrum-menetelmää kuvaavat prosessikuvaukset, sillä kuten jo luvussa 2.2 todettiin, on se yleisin ketterä ohjelmistokehitysmenetelmä ja siten sen mukainen projektitoiminta määrittää myös suurta osaa projekteista (VersionOne, 2018). Näin ollen on tämän menetelmän esittämien prosessikuvauksien tarkasteleminen hyvä lähtökohta ketterän ohjelmistokehityksen vaiheiden tarkastelemiselle. Scrumia kuvaavia prosessikuvauksia valittiin tarkasteluun kaksi kappaletta, joihin valikoitiin Abrahamssonin ym. (2017) tutkimuksessa esitetty prosessikuvaus sekä Vlaanderenin ym. (2011) tutkimuksessa esitetty yksinkertaistettu Scrumin prosessikuvaus.

Näiden kahden prosessikuvauksen lisäksi pyrittiin löytämään tutkimuksista ja muusta tietokirjallisuudesta yleisiä prosessikuvauksia, jotka esittäisivät yleisesti ketterän ohjelmistokehitysprojektin prosessia. Näiden prosessikuvausten etsimiseen hyödynnettiin Google Scholar -hakukonetta seuraavilla hakusanoilla ”agile process model”, ”agile process”, ”scrum process”, ”generic agile process”, “generalized agile”. Etsimisen tuloksena löydettiin vain kaksi niin yleistävää prosessikuvausta. Nämä olivat Levyn, Shortin ja Measeyn (2015) kirjassa esitetty yleinen prosessikuvaus ja Bhaleraon, Puntambekarin ja Inglen (2009) tutkimuksessa toteutettu yleinen elinkaarimalli.

Näiden hakutulosten perusteella tarkempaan tarkasteluun lopulta valikoitui neljä prosessikuvausta. Näistä kaksi olivat prosessikuvauksia Scrum-menetelmästä ja toiset kaksi yleistäviä prosessikuvauksia. Tarkasteltaessa löytyneitä prosessikuvauksia on tarkoituksena löytää niissä esitetty projektin kulun ketterä prosessi ja sen vaiheet.

Ensin tarkastellaan tarkemmin Scrum-menetelmään liittyviä kahta prosessikuvausta (ks. liite 1). Abrahamssonin ym. (2017) prosessikuvauksessa prosessi on jaettu kolmeen eri vaiheeseen. Alkuvaiheeseen (pregame phase), kehitysvaiheeseen (development phase) ja lopetusvaiheeseen (postgame phase). Alkuvaiheessa on esitetty varsinainen suunnittelu ja korkean tason suunnittelu sekä työn jakautuminen tuotteen kehitysjonolle (product backlog). Esitetyssä kehitysvaiheessa työ on jaettu iteraatioihin. Kunkin iteraation työt on jaettu niille kuuluville kehitysjonoille. Töitä toteutetaan iteraation kehitysvaiheen ajan, jonka loputtua siirrytään loppuvaiheeseen, jossa toteutetaan kehitystyöhön kuuluvat julkaisut ja testaukset. Vastaavasti Vlaanderenin ym. (2011) tiedonkulun kuvauksessa esitetty prosessi ja sen vaiheet ovat samankaltaisia kuin Abra-

hamssonin ym. (2017) kuvauksessa. Vlaanderenin ym. (2011) esittämässä yksinkertaisemmassa Scrumin prosessikuvauksessa on esitetty visiosta jalostetun työn kulkeutuminen kehitysjonolle, joista edelleen työ jaetaan kullekin iteraatiolle. Kunkin iteraation jälkeen palataan takaisin työstämään vaatimuksia tuotteen kehitysjonolle.

Seuraavaksi tarkastellaan tarkemmin kahta löydettyä yleistävää prosessikuvausta (ks. liite 2). Keskenään nämä prosessikuvaukset eroavat esitystavoiltaan melko paljon. Levyn, Shortin ja Measeyn (2015) tietokirjassa esittämä yleinen kuvaus ketterästä ohjelmistokehitysprojektista on abstraktiotasoltaan yksinkertaisempi kuin muut tarkasteltavat kuvaukset. Tämä prosessikuvaus esittää yleispiirteisesti tärkeimmät elementit, joita ketterään ohjelmistokehitykseen kuuluu sekä sen yleisen luonteen. Kuvaus on jaettu kolmeen osaan, joissa on keskellä varsinaiset tapahtumat ja prosessi, ylhäällä keskeiset työkalut ja alla keskeiset roolit. Yleisesti tapahtumien kulkua kuvataan iteratiivisena jatkuvana toimintana. Kuvaus ei varsinaisesti ota kantaa prosessin vaiheisiin vaan esittää yleisesti mitä tapahtumia prosessiin kuuluu.

Toisena yleistävänä prosessikuvauksena on Bhaleraon, Puntambekarin ja Inglen (2009) tutkimuksessa muodostettu ketterän ohjelmistokehityksen prosessikuvaus, jota he kutsuvat ketterän ohjelmistokehityksen elinkaaren prosessikuvaukseksi. Tämä kuvaus (ks. liite 2) on huomattavasti tarkempi ja vaiheikkaampi kuin Levyn ym. (2015) kirjassa esittämä kuvaus. Bhalerao ym. (2009) ovat jakaneet prosessikuvauksen viiteen eri vaiheeseen, jossa projekti alkaa projektin alkumäärittelystä, sitten vaatimusten keräämisestä, jonka jälkeen vaatimukset priorisoidaan ja suunnitellaan projektin seuraava iteraatio. Tämän jälkeen toteutetaan suunniteltu iteraatio ja sen loputtua julkaistaan toteutettu tuote. Lopuksi palataan takaisin vaatimusmäärittelyvaiheeseen. Yleisesti Bhaleraon ym. (2009) prosessikuvauksesta voidaan huomata selvästi projektin vaiheet, tapahtumat ja niiden väliset riippuvuudet sekä työn kulkeutumisen projektissa.

Jokainen näistä neljästä tarkastellusta prosessikuvauksesta, oli esitystavoiltaan ja tarkkuudeltaan erilaisia. Ne osoittavat millä tavoin ketterää ohjelmistokehitysprojektia voidaan kuvailla prosessikuvausten avulla. Näistä kaikista prosessikuvauksista oli huomattavissa samat perusperiaatteet ja elementit. Jokaisessa kuvauksessa esitettiin tapahtumien etenemistä projektin aikana sekä työn kulkeutumista eri tapahtumalta toiselle. Samoin myös kehitysjonot olivat läsnä näissä jokaisessa prosessikuvauksessa. Nämä tarkastelut prosessikuvaukset toimivat tärkeänä pohjatietona prosessin vaiheiden muodostamiselle ja toimivat malliesimerkkeinä siitä, minkälaisia prosessikuvauksia on toteutettu aiheesta.

Näin ollen voidaan muodostaa ketterälle ohjelmistokehitysprojektille tyypillisen prosessin vaiheet. Tarkoituksena on pystyä muodostamaan yleiset vaiheet projektille ketterästä ohjelmistokehitysmenetelmästä riippumatta ja näin saada luotua käsitys projektin yleistetystä kulusta. Seuraavaksi esitetäänkin vaiheistus, joka esittää suurpiirteiset vaiheet ketterän ohjelmistokehitysprojektin kulusta.

1) Projekti alkaa visiosta ja yleisestä alustavasta määrittelystä.
2) Tämän vision pohjalta määritellään tarkemmin kehitettävään tuotteeseen tarvittavat työt, ja ne kirjataan kehitysjonolle.
3) Ennen iteraatiota valitaan kehitysjonolta tehtävät seuraavalle iteraatiolle.
4) Iteraation aikana toteutetaan sille priorisoituja tehtäviä.
5) Iteraation jälkeen käydään läpi toteutettuja asioita, mahdollisesti julkaistaan toteutetut asiat ja palataan takaisin suunnittelemaan seuraavaa iteraatiota.

Prosessin tärkeimmiksi osiksi voidaankin luonnehtia iteraatiot, joista projekti koostuu, sekä projektiin kuuluvaa kehitysjonoa, johon työtä toteutetaan. Lisäksi visioon pohjautuva jatkuva suunnittelu on projektissa tärkeässä asemassa. Tämä prosessi ja sen vaiheet vastaavatkin tämän tutkielman toiseen taustoittavaan tutkimuskysymykseen sekä tässä luvussa toteutettu prosessikuvausten tarkastelu tukee tutkielmassa toteutettuja muita osioita.

3.2 Ketterälle ohjelmistokehitykselle tyypilliset elementit

Tässä alaluvussa käydään läpi toteutettu analyysi, jossa pyrittiin löytämään ketterän ohjelmistokehityksen tyypillisimmät elementit ja joka auttaa ketterän ohjelmistokehityksen yleistämisessä ja siihen liittyvien tyypillisimpien käsitteiden määrittelemisessä. Luvun ensimmäisessä alaluvussa käydään läpi analyysiin valikoitujen tutkimusten etsiminen ja esittely, ja toisessa alaluvussa käsitellään löydökset.

3.2.1 Aihepiiriin liittyvän lähdemateriaalin etsiminen

Tavoitteena oli löytää mahdollisimman luotettavaa ja kattavaa tutkimukseen pohjautuvaa lähdemateriaalia, jossa on pyritty koostamaan ketterään ohjelmistokehitykseen liittyviä yleisimpiä elementtejä, kuten tapahtumia, käsitteitä ja muita piirteitä. Tutkimusten etsimiseen käytettiin Google Scholar -hakukonetta, ja hakusanoina käytettiin sanoja ”agile practices”, ”agile principles”, ”agile software development usage”, ”agile usage”, ja ”agile development”. Kussakin haussa käytiin läpi kaksi ensimmäistä hakusivua ja hakutulosten analysointi toteutettiin manuaalisesti. Tuloksista valikoitiin ne lähteet, joissa esitettiin ketterän ohjelmistokehityksen yleisimpiä elementtejä. Analyysiin valikoituja lähteitä saatiin yhteensä 14 kappaletta. Näistä lähteistä 12 kappaletta oli tutkimusartikkelia ja 2 oli aiheeseen liittyvää tietokirjallisuutta. Analyysiin valikoidut lähteet ovat listattuna taulukossa 3.

TAULUKKO 3 Analyysiin valikoidut lähteet

#	Tyyppi	Otsikko	Tekijät	Vuosi
1	Tutkimus	Agile Software Development Methodologies: Survey of Surveys	Al-Zewairi ym.	2017
2	Tutkimus	Requirements engineering in agile software development	De Lucia ja Qusef	2010
3	Tutkimus	Review on Agile requirements engineering challenges	Elghariani ja Kama	2016
4	Tutkimus	A systematic literature review on agile requirements engineering practices and challenges	Inayat ym.	2015
5	Kirja	Agile Foundations: Principles, practices and frameworks Review on Agile requirements engineering challenges	Levy, Short ja Measey	2015
6	Tutkimus	Perceived importance of agile requirements engineering practices – A survey	Ochodek ja Kopczyńska	2018
7	Tutkimus	Agile requirements engineering practices and challenges: an empirical study	Ramesh, Cao ja Baskerville	2010
8	Tutkimus	Survey on agile and lean usage in Finnish software industry	Rodríguez ym.	2012
9	Tutkimus	Agile Practices: An Assessment of Perception of Value of Professionals on the Quality Criteria in Performance of Projects	De Santos ym.	2011
10	Tutkimus	A disciplined approach to adopting agile practices: the agile adoption framework	Sidky, Arthur ja Bohner	2007
11	Tutkimus	Exploring the Relationship between Organizational Adoption Motives and the Tailoring of Agile Methods	Tripp ja Armstrong	2014
12	Tutkimus	Assimilation of agile practices in use: Assimilation of agile practices in use	Wang, Conboy ja Pikkarainen	2012
13	Tutkimus	VersionOne 12th Annual State of Agile Report	VersionOne	2018
14	Tutkimus	A systematic mapping study on the combination of software architecture and agile development	Yang, Liang ja Avgeriou	2016

3.2.2 Löydetyt elementit

Analyysiin olennaista materiaalia sisältävien lähteiden etsimisen ja valikoimisen jälkeen listattiin kussakin lähteessä esitetyt ketterän ohjelmistokehityksen yleisimmät elementit. Näissä löydetyissä lähteissä oli paljon eroavaisuuksia ja joissain lähteissä painotettiin enemmän käytänteitä, kun taas toisissa enemmän esimerkiksi teknisempiä käsitteitä. Analyysin lopputuloksena toteutettiin taulukointi, jossa on esitettynä löydöksissä esitetyt elementit, ja niiden esiintyvyys. Tällä tavoin pystytään esittämään, mitkä käsitteet ovat esiintyneet useimmiten näissä lähteissä ja mitkä harvemmin.

Analyysin tulokset ovat esitettynä taulukossa 4. Tyypillisimpiä elementtejä liittyen ketterään ohjelmistokehitykseen löydettiin yhteensä 28 kappaletta. Tekijät jaettiin karkeasti kolmeen eri kategoriaan, jotta taulukkoa on helpompi tulkita. Nämä kategoriat ovat: ”tapahtumat ja toimenpiteet”, ”ominaispiirteet” sekä ”työkalut ja tekniset tekijät”. Taulukossa on esitettynä kunkin elementin esiintyvyys kussakin lähdemateriaalissa. Yleisesti voidaan huomata, että ketterään ohjelmistokehitykseen liittyy paljon elementtejä ja niitä on myös montaa eri tyyppiä. Toteutettu analyysi osoittaa näiden löydettyjen lähteiden pohjalta yleisimmät elementit, joita esiintyy ketterässä ohjelmistokehityksessä.

TAULUKKO 4 Analyysin tulokset

Tapahtumat ja toimenpiteet	**Esiintyvyys**
Retrospektiivit (eng. retrospectives)	1, 3, 4, 5, 6, 8, 9, 11, 12, 13, 14
Iteraation suunnittelu (eng. iteration planning)	1, 5, 6, 8, 9, 10, 11, 12, 13, 14
Päivittäispalaverit (eng. daily meetings)	5, 6, 8, 9, 11, 12, 13, 14
Katselmointipalaverit (eng. review meetings)	1, 3, 4, 5, 6, 7, 13, 14
Tarinoiden kartoitus (eng. story mapping)	3, 4, 5, 6, 9, 10, 13, 14
Priorisointi (eng. priorization)	1, 4, 5, 6, 7, 13
Julkaisun suunnittelu (eng. release planning)	5, 8, 9, 11, 13, 14
Hyväksyttäminen (eng. acceptance)	1, 3, 4, 6, 7, 13
Työmäärän arviointi (eng. work estimation)	5, 9, 10, 13, 14
Prototyypitys (eng. prototyping)	1, 3, 4, 6, 7
Etenemissuunnitelu ja visiointi (eng. roadmapping and visioning)	5, 6, 13
Ominaispiirteet	
Osapuolten tiivis osallistuminen	1, 2, 3, 4, 5, 6, 8, 9, 10, 12, 13, 14
Pienet, itseohjautuvat ja monitoimiset tiimit	1, 2, 3, 4, 9, 10, 13, 14
Lyhyet iteraatiot ja julkaisut	2, 5, 8, 10, 12, 13, 14
Kasvokkainen vuorovaikutus	1, 3, 4, 5, 7, 10
Kehittyvät ja muuttuvat vaatimukset	1, 2, 3, 4, 7, 13
Jatkuva suunnittelu	1, 3, 4, 7
Kestävä työtahti	5, 13, 14
Työkalut ja tekniset tekijät	
Testauslähtöinen kehitys (eng. test-driven)	1, 3, 4, 8, 9, 10, 11, 12, 13, 14
Ohjelmakoodin jatkuva refaktorointi (eng. refactoring)	1, 3, 4, 8, 9, 11, 12, 13, 14
Jatkuva integraatio (eng. continuous integration)	8, 9, 10, 11, 12, 13, 14
Kehitysjono (eng. backlog)	5, 8, 9, 10, 13, 14
Pariohjelmointi (eng. pair-programming)	8, 9, 10, 12, 13, 14
Edistymiskäyrä (eng. burn-down chart)	5, 8, 9, 11, 13
Yhteisvastuu ohjelmakoodista (eng. collective code ownership)	8, 9, 12, 13, 14
Yhteinen työskentelytila	6, 10, 13, 14
Automatisoidut järjestelmäkoonnit (eng. automated builds)	8, 11, 13
Kanban-taulu (eng. kanban board)	5, 9, 13

3.3 Yhteenveto teoriakatsauksesta

Tässä luvussa toteutettiin kaksi erilaista analyysiä. Ensimmäisessä alaluvussa tarkasteltiin olemassa olevia prosessikuvauksia ja näiden prosessikuvausten pohjalta pyrittiin luomaan pelkistetyt vaiheet ketterälle ohjelmistokehitysprojektin prosessille. Toisessa alaluvussa toteutettiin analyysi, jossa etsittiin ketterän ohjelmistokehityksen tyypillisimpiä elementtejä. Tuloksena saatiin koostettua taulukko yleisimmistä ketterän ohjelmistokehityksen elementeistä. Näiden kahden toteutetun analyysin pohjalta voidaan vastata tutkielman kahteen sille asetettuun taustoittavaan tutkimuskysymykseen, jotka pohjaavat varsinaista tutkittavaa ongelmaa ja siihen liittyvää päätutkimuskysymystä. Luvussa 3.1

vastattiin kysymykseen liittyen ketterän ohjelmistokehitysprojektin vaiheista ja luvussa 3.2 vastattiin kysymykseen liittyen ketterän ohjelmistokehitysprojektin yleisimmistä elementeistä. Näiden toteutetut analyysien avulla saatiin muodostettua käsitys yleisestä ketterän ohjelmistokehitysprojektin kulusta ja siitä, miten ketterää ohjelmistokehitystä yleisesti voidaan kuvailla prosessikuvausten avulla. Lisäksi pystyttiin koostamaan lista yleisimmistä ketterään ohjelmistokehitykseen liittyvistä elementeistä ja näiden elementtien avulla saadaan käsitys siitä, mitä seikkoja yleisesti liittyy ketterään ohjelmistokehitykseen. Nämä analyysit pohjaavat tämän tutkielman varsinaista tutkimusongelmaa ja pohjaavat tärkeää tietoa tämän tutkielman varsinaiselle tutkivalle osuudelle.

4 TUTKIMUSMENETELMÄ

Tämän tutkimuksen varsinaiseen tutkimusongelmaan ja siihen liittyvään päätutkimuskysymykseen pyritään vastaamaan konstruktiivisella tutkimusotteella. Tähän ongelmaan kyseinen tutkimusote soveltuu hyvin, sillä tavoitteena on ratkaista reaalimaailman ongelma eli se, mitä asioita asiakkaan on tärkeää tietää osallistuessaan ketterään ohjelmistokehitysprojektiin. Konstruktiivisessa tutkimusotteessa on tarkoituksena pyrkiä luomaan jotain uutta. Pyritään siis luomaan ratkaisua reaalimaailman ongelmaan luomalla siihen jokin ongelman ratkaisuun sopiva artefakti, esimerkiksi jokin malli, diagrammi, suunnitelma tai vaikkapa tuote. Tämä ongelman ratkaisu, eli artefakti pohjautuu jo olemassa olevaan tietoon ja tutkimusotteessa pyritäänkin yhdistelemään jo olemassa olevaa tietämystä ja tutkijan omaa näkemystä ongelmasta. Näin ollen ratkaisu perustuukin hyvin vahvasti jo aikaisempaan tietoon (Lukka, 2014.)

Tässä tutkielmassa konstruktiiviseksi tutkimusmenetelmäksi on valittu tietojärjestelmätieteen tutkimuksessa käytettävä suunnittelutieteellinen tutkimusmenetelmä. (eng. design science) (Peffers ym., 2007). Tutkielma noudattelee Peffersin ym. (2007) tutkimuksen määrittelemää prosessia suunnittelutieteelliselle tutkimukselle. Seuraavaksi käydään lyhyesti läpi, mitä eri vaiheita suunnittelutieteellinen tutkimusmenetelmä sisältää, ja miten tutkimuksen kulku sen osalta etenee. Peffersin ym. (2007) koostama prosessi muodostuu kuudesta eri perusvaiheesta ja ne ovat seuraavat:

1) Ongelman tunnistaminen ja motivointi
2) Ratkaisun tavoitteiden määrittely
3) Suunnittelu ja kehittäminen
4) Demonstrointi
5) Arviointi
6) Viestintä

Peffers ym. (2007) kuvaavat vaiheita seuraavasti. Ensimmäisessä vaiheessa määritellään tutkimuksen ongelma ja perustelu sen tärkeydelle. Toisessa vaiheessa kuvataan, mitä tavoitteita ratkaisulla on, ja mitä on mahdollista toteuttaa.

Kolmannessa vaiheessa luodaan ongelmalle ratkaisu eli artefakti, joka on esimerkiksi jokin malli tai tuote. Neljännessä vaiheessa pyritään osoittamaan, miten luodulla ratkaisulla pystytään ratkaisemaan ongelma, johon artefakti on luotu ratkaisuksi. Viidennessä vaiheessa arvioidaan, kuinka hyvin artefakti ratkaisee ongelman, ja mitä puutteita siinä esimerkiksi on. Kuudennessa eli viimeisessä vaiheessa viestitään ratkaisu.

Peffersin ym. (2007) mukaan suunnittelutieteellisen menetelmän vaiheiden toteuttaminen voidaan aloittaa neljästä eri lähtökohdasta. Prosessi voi alkaa ongelmasta, jolloin prosessi aloitetaan kohdasta yksi. Toisaalta se voi olla tavoitelähtöinen ja silloin prosessi alkaa kohdasta kaksi. Prosessi voidaan aloittaa myös suoraan suunnittelusta ja kehittämisestä, jolloin prosessi alkaa kohdasta kolme. Lopuksi prosessi voi myös alkaa demonstroinnista, eli kohdasta neljä.

Tässä tutkielmassa tätä esiteltyä suunnittelutieteellistä menetelmää sovelletaan seuraavalla tavalla. Tämän tutkielman suunnittelutieteellisenä lähtökohtana on ongelmakeskeisyys. Kuten suunnittelutieteelliseen tutkimusmenetelmään kuuluu, tunnistetaan olemassa oleva ongelma ja perustellaan miksi ongelma tulisi ratkaista. Näitä käsitellään tämän tutkielman johdannossa sekä luvussa 5.1.1.

Määriteltyyn ongelmaan pohjautuen tutkielmassa on esitetty ongelman ratkaisulle sen tavoitteet, sekä keino, jolla ongelma pyritään ratkaisemaan. Suunnittelutieteelliseen tutkimusmenetelmään liittyen ongelma pyrittiin ratkaisemaan sille kehitellyllä artefaktilla. Tässä tutkielmassa artefaktina toimii tutkielmassa kehitetty prosessikuvaus, joka osoittaa asiakkaalle ketterän ohjelmistokehityksen kulun ja siihen liittyvät yleisimmät käytänteet ja termit. Tämän luodun prosessikuvauksen avulla pystytään vastaamaan siihen, mitä asiakkaan tulisi tietää ketterästä ohjelmistokehitysprojektista. Samoin asiakas konkreettisesti pystyy hyödyntämään prosessikuvausta osallistuessaan ketterään ohjelmistokehitysprojektiin ja tämän avulla saamaan yleiskäsityksen aiheesta. Tämän ratkaisun tavoitteet on määritelty luvussa 5.1.2.

Ratkaisun tavoitteiden määrittelyn jälkeen luvussa 5.1.3 on esitettynä suunnittelutieteellisen tutkimusmenetelmän vaiheiden mukaisesti artefaktin suunnittelu ja kehittäminen. Tämän kehittämisen tuloksena saatiin luonnos artefaktista, joka pyrkii vastaamaan sille asetettuun ongelmaan ja tätä luonnosta demonstroitiin kuviossa 2. Artefaktin suunnittelun ja kehittämisen sekä demonstroinnin jälkeen artefaktia tulee arvioida jollain keinolla. Artefaktin, eli asiakkaalle suunnatun ketterän ohjelmistokehitysprojektin prosessikuvauksen arvioiminen toteutettiin haastatteluiden avulla. Arviointiprosessi ja sen tulokset ovat esitettynä luvussa 5.2. Näiden toteutettujen haastatteluiden avulla oli kolme tavoitetta. Ensinnäkin niillä varmennettiin entisestään tutkimusongelman olemassaoloa ja sen tarpeellisuutta. Toiseksi haastatteluilla pyrittiin arvioimaan prosessikuvauksen vastaamista sille asetettuun ongelmaan ja sen tavoitteisiin, sekä yleisesti prosessikuvauksen hyödyllisyyttä sen käyttötarkoitukseen. Ja kolmanneksi haastatteluiden avulla kerättiin muutosehdotuksia, joiden avulla prosessikuvaus vastaisi paremmin sille asetettuihin tavoitteisiin.

Tämän haastatteluiden avulla toteutetun arvioinnin jälkeen siirryttiin tutkielmassa iteratiivisesti suunnittelutieteellisen tutkimusmenetelmän kohtaan 3. Haastatteluissa esiintyneiden muutosehdotusten mukaisesti prosessikuvausta jatkokehitettiin, jotta se vastaisi paremmin määriteltyä ongelmaa ja prosessikuvaukselle asetettuja tavoitteita. Prosessikuvaukseen toteutetut muutokset on esitetty luvussa 5.2. Tutkielman suunnittelutieteellisen tutkimusmenetelmän mukaisesti toteutetun artefaktin lopullinen versio on esitetty kuviossa 3 ja sen mahdollisia käyttötarkoituksia ja tarkempi selostus sen osa-alueista on esitetty luvussa 6.

5 PROSESSIKUVAUKSEN KEHITTÄMINEN

Tässä luvussa käydään läpi tutkielmassa käytetty tutkimusmenetelmä sekä tutkimusongelman ratkaiseminen tutkimusmenetelmän mukaisesti. Ensimmäisessä alaluvussa käsitellään toteutettavan tutkimuksen toteutustapaa ja siihen valittua tutkimusmenetelmää. Toisessa alaluvussa kuvataan tutkimusmenetelmään pohjautuva tutkimus sekä siihen liittyvä tuotos. Kolmannessa alaluvussa arvioidaan tutkimuksessa toteutettu tuotos haastattelujen avulla ja neljännessä esitellään haastatteluista kerätyt palautteet tuotoksesta. Viidennessä, eli viimeisessä alaluvussa esitetään tuotokseen toteutetut korjaukset haastatteluissa esitettyjen palautteiden pohjalta.

5.1 Prosessikuvauksen kehittämisen vaiheet

Tässä alaluvussa käydään läpi prosessikuvauksen rakentaminen ja vaiheet, joilla prosessikuvauksen lopputulokseen on päästy. Ensin käydään läpi sitä, mikä nykyisissä ratkaisuissa on pulmallista, ja mitä tavoitteita luotavalla ongelman ratkaisulla on. Ongelman ja tavoitteiden määrittelyn jälkeen toteutetaan luonnos ongelman ratkaisusta.

5.1.1 Ongelman tunnistaminen

Kuten jo tämän tutkimuksen johdannossa käsiteltiin lyhyesti, on asiakkaan osallistuminen projektiin ensiarvoisen tärkeää projektin yleisen onnistumisen kannalta. Jos kuitenkin tarkastellaan yleisesti ketterää ohjelmistokehitystä ja siihen liittyviä menetelmiä, on ne suunniteltu projektiin osallistuvan kehitystiimin käyttöön, eikä varsinaisesti asiakasta varten. Samoin tutkielmaa toteutettaessa ei löydetty ainuttakaan asiakkaalle suunnattua prosessikuvausta ketterästä ohjelmistokehitysprojektista. On siis ilmeistä, että vaikka asiakas on hyvin tärkeässä osassa projektia, ei asiakkaalle välttämättä ole tarjolla hyviä resursseja tai materiaaleja ketterän ohjelmistokehityksen käsittämiseen. Tähän väittämään

ei ole toteutettu riittävästi tutkimusta, joten yksiselitteisesti ei voida tehdä johtopäätöstä siitä, että asiakkaalla yleensä ei ole riittävää osaamista ketterästä ohjelmistokehitysprojektista. Tätä ongelmaa kuitenkin pyrittiin varmentamaan tutkielmassa toteutetuissa haastatteluissa ja näitä tuloksia käydään tarkemmin läpi luvussa 5.2.2.

Toinen ongelmallinen seikka on se, että projektit, joissa ketterää ohjelmistokehitystä toteutetaan, ovat luonteeltaan kaikki hyvin erilaisia. Kuitenkin, kuten luvussa 3 pystyttiin todentamaan, on ketterillä ohjelmistokehitysprojekteilla hyvin tyypilliset vaiheet ja elementit käytetystä menetelmästä huolimatta. Näin ollen asiakas huomioiden on tärkeää, ettei keskitytä niinkään mihinkään tiettyyn ketterään ohjelmistokehitysmenetelmään, vaan keskitytään nimenomaan tyypillisimpiin asioihin. Tätä seikkaa puoltavat myös esimerkiksi VersionOne-yrityksen (2018) saamat tulokset, joissa yhdistellyt ketterät ohjelmistokehitysmenetelmät olivat osuutena melko merkittävä osa käytetyistä menetelmistä. On siis selvää, että useissa projekteissa näitä menetelmiä sovelletaan ja niitä ei toteuteta täysin menetelmiä koskevien ohjeistuksien mukaan.

Ketterän ohjelmistokehityksen perehdyttämiseen on monia eri keinoja. Kuitenkin menetelmiä kuvaillaan hyvin usein prosessikuvauksilla, joita esimerkiksi luvussa 3.1 esiteltiin. Prosessikuvauksista voidaan nopeasti hahmottaa elementtien kokonaisuus ja niiden väliset suhteet visuaalisten kuvausten ansiosta. Tästä syystä prosessikuvaus on myös asiakkaan perehdyttämiseen hyvä valinta. Tämän tutkielman tarkoituksena onkin luoda sellainen prosessikuvaus, joka ei niinkään nojaa mihinkään tiettyyn ketterään ohjelmistokehitysmenetelmään, vaan pyrkii esittämään asiakkaan kannalta kaikkein olennaisimmat ja tyypillisimmät asiat ketterästä ohjelmistokehitysprojektista.

Taulukkoon 5 on vielä koostettu tässä alaluvussa käsitellyt ongelmat nykyisissä prosessikuvauksissa. Ensinnäkin yleistäviä prosessikuvauksia ketteristä ohjelmistokehitysprojekteista ei ole riittävästi, ja toiseksi, olemassa olevat prosessikuvaukset ovat suunnattu projektin kehitystiimille. Viimeinen ongelma on se, ettei prosessikuvauksissa korosteta asiakkaalle oleellisia asioita.

TAULUKKO 5 Nykyisten prosessikuvausten ongelmat

#	**Ongelma**
1	Yleistettyjä prosessikuvauksia ei juuri ole, vaan ne yleensä kuvaavat jotain tiettyä kehitysmenetelmää
2	Pääosin prosessikuvaukset on tarkoitettu muiden kuin asiakkaan käyttöön
3	Prosessikuvauksista ei ilmene mihin asioihin asiakkaan tulisi erityisesti kiinnittää huomiota

5.1.2 Prosessikuvauksen tavoitteet

Edellisessä luvussa taulukkoon 5 määriteltyjen ongelmien pohjalta on muodostettavissa tavoitteet uudelle kehitettävälle prosessikuvaukselle, jolla näitä ongelmia pyritään ratkaisemaan. Ratkaisun tavoitteena on luoda prosessikuvaus, joka on tarkoitettu asiakkaan käyttöön, ja joka sisältää ne yleisimmät elementit

ja projektin vaiheet ketterästä ohjelmistokehitysprojektista, jotka ovat asiakkaalle kaikkein olennaisimmat ja tukevat asiakkaan osallistumista projektiin. Prosessikuvauksen visuaaliseen esitykseen otetaan mallia aiemmin luvussa 3.1 esitetyistä prosessikuvauksista ja lisäksi siihen sisällytetään tämän tutkielman teoriakatsauksessa esitettyjä asioita. Nämä prosessikuvauksen tavoitteet on kerrottu tarkemmin taulukossa 6.

TAULUKKO 6 Uuden prosessikuvauksen tavoitteet

#	Tavoite	Mihin ongelmaan tavoite liittyy
1	Prosessikuvauksen laatimisessa hyödynnetään tutkimukseen pohjautuvaa tietoa	1, 2
2	Esittää ketterän ohjelmistokehitysprojektin tyypillisen prosessin ja vaiheet	1
3	Esittää yleisimmät ketterän ohjelmistokehitysprojektin elementit	1
4	Rajoittuu asiakkaalle olennaisimpiin asioihin	2, 3
5	Riittävän yksinkertainen ja helposti ymmärrettävissä	2, 3

5.1.3 Prosessikuvauksen kehittämisprosessi

Jotta prosessikuvaus saavuttaisi sille asetetut tavoitteet luodaan se sellaisissa vaiheissa, joissa pyritään ottamaan huomioon nämä prosessikuvaukselle asetetut tavoitteet. Seuraavaksi esitetään prosessikuvauksen toteuttamisen vaiheet, ja mitä kukin vaihe pitää sisällään.

1) Asiakkaalle olennaisten elementtien rajaaminen
2) Prosessin mallintaminen
3) Elementtien mallintaminen
4) Selitetekstiliitteen tuottaminen

1) Asiakkaalle olennaisten elementtien rajaaminen
Prosessikuvaus kehitetään asiakkaan käyttöön, joten siihen sisällytettävät elementit on valittava siten, että kuvaukseen sisällytetään elementtejä, jotka ovat asiakkaalle olennaisia. Luvussa 3.2 määritellyt elementit toimivat perustana laadittavalle prosessikuvaukselle. Määritellyistä elementeistä mukaan otetaan ne, jotka katsotaan asiakkaan kannalta oleellisimmiksi. Taulukon 4 elementtien rajaaminen asiakkaalle olennaisiin asioihin on toteutettu seuraavasti. Listasta on karsittu elementit, jotka eivät ole asiakkaalle tärkeitä. Ne on listattu alla olevassa luettelossa. Nämä kahdeksan käsitettä ovat asiakkaalle epäolennaisia siksi, että ne liittyvät pelkästään kehitystiimiin, ja tästä syystä ne eivät ole olennaisia asiakkaalle.

- Automatisoidut järjestelmäkoonnit
- Jatkuva integraatio
- Ohjelmakoodin refaktorointi

- Pariohjelmointi
- Pienet, itseohjautuvat ja monitoimiset tiimit
- Testauslähtöinen kehitys
- Yhteinen työskentelytila
- Yhteisvastuu ohjelmakoodista

Asiakkaalle epäolennaisten elementtien karsimisen jälkeen 28 alkuperäisestä elementistä on jäljellä 19 elementtiä. Jäljelle jääneitä elementtejä pystytään kuitenkin myös yhdistelemään, sillä moni näistä käsitteistä koostuu päällekkäisistä käsitteistä. Yhdistetyt ja yksinkertaistetut termit ovat seuraavat:

- Edistymiskäyrä – sisällytetty kehitysjonoon
- Etenemissuunnittelu ja visiointi – sisällytetty jatkuvaan suunnitteluun
- Kanban-taulu – sisällytetty kehitysjonoon
- Tarinoiden kartoitus – sisällytetty jatkuvaan suunnitteluun
- Työmäärän arviointi – sisällytetty priorisointiin

Näiden asiakkaalle epäolennaisten elementtien ja muutamien elementtien yhdistämisen jälkeen on prosessikuvaukseen sisällytettäviä elementtejä ja käsitteitä yhteensä 14. Nämä jäljelle jääneet elementit ovat seuraavat:

- Hyväksyttäminen
- Iteraation suunnittelu
- Jatkuva suunnittelu
- Julkaisun suunnittelu
- Kasvokkainen vuorovaikutus
- Katselmointipalaverit
- Kehittyvät ja muuttuvat vaatimukset
- Kehitysjono
- Kestävä työtahti
- Lyhyet iteraatiot ja julkaisut
- Osapuolten tiivis osallistuminen
- Priorisointi
- Päivittäispalaverit
- Retrospektiivit

Näiden elementtien pohjalta aloitetaan prosessikuvauksen varsinainen mallintaminen. On tärkeää kuitenkin huomioida, että nämä elementit toimivat prosessikuvauksen pohjana, mutta prosessikuvauksen tekemisessä hyödynnetään myös muita teoriakatsauksessa esitettyjä elementtejä. Tämä lista muodostaa pohjan asiakkaalle olennaisimmista elementeistä ja siten auttaa prosessikuvauksen kokonaisuuden mallintamisessa.

2) Prosessin mallintaminen
Mallinnettaessa prosessia on tärkeää pitää mielessä prosessikuvaukselle tarkoitettu kohderyhmä, eli projektiin osallistuva asiakas. Tästä syystä on pyrittävä kuvaamaan asiat oikealla abstraktiotasolla, jotta prosessi ei olisi liian monimutkainen ja vaikeasti ymmärrettävissä asiakkaalle, mutta kuitenkin sellainen, että se sisältäisi riittävän kattavasti asiakkaalle tärkeät prosessin vaiheet. Kaikkein tärkeintä onkin, että prosessin mallintaminen pyrkii helppoon ymmärrettävyyteen, ja ettei lopputulos edellytä kattavaa pohjatietoa ketterästä ohjelmistokehityksestä. Olennaisinta on, että prosessikuvauksessa mallinnettu prosessi sisältää samat vaiheet, joita luvussa 3.1 koostettiin. Näin ollen prosessikuvauksessa on selvästi oltava esillä projektin alku, loppu sekä ketterään ohjelmistokehitykseen vahvasti kuuluva iteratiivinen luonne.

3) Elementtien mallintaminen
Pohja prosessikuvaukseen valittaville elementeille on toteutettu vaiheessa yksi. Elementtien mallintamiseen käytetään hyödyksi myös teoriakatsauksessa käsitteistöä ketterästä ohjelmistokehitysprojektista. Tämän pohjalta muodostetaan mahdollisimman järkevä elementtikokonaisuus, joka samalla sopii vaiheessa kaksi mallinnettuun projektin prosessiin ja sen vaiheisiin. Toisin sanoen nämä elementit pyritään lisäämään yhteen prosessin mallintamisen kanssa.

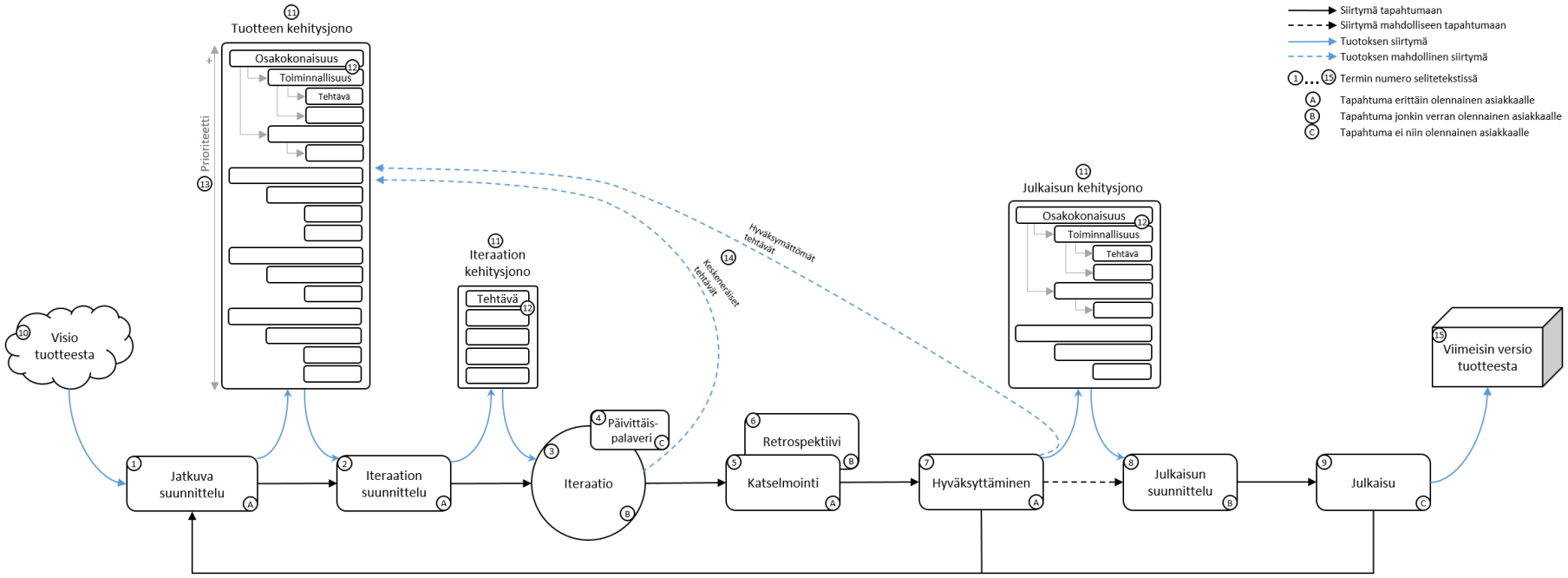
4) Selitetekstiliitteen tuottaminen
Asiakkaan ymmärtämisen varmistamiseksi on tärkeää toteuttaa myös prosessikuvaukseen liittyvä erillinen liite, joka sisältää prosessikuvauksessa käytetyt käsitteet ja niiden määrittelyt. Lisäksi, koska joitain ketterään ohjelmistokehitykseen olennaisesti liittyviä käsitteitä ei pystytä prosessikuvauksessa visuaalisesti mallintamaan, ne on käsiteltävä kirjallisessa muodossa. Tämän erillisen tekstiliitteen avulla asiakas pystyy tarkemmin ymmärtämään prosessikuvausta ja siihen liittyvää terminologiaa sekä yleistä ketterän ohjelmistokehityksen luonnetta.

Prosessikuvauksen liite koostuu kolmenlaisista käsitteistä. Ensimmäiseksi liitteessä on lyhyt kuvaus prosessikuvauksesta. Toiseksi liitteessä on lueteltu lyhyesti kuvauksessa esiintyneet termit määritelmineen. Viimeiseksi liite sisältää mallintamatta jääneet ketterälle ohjelmistokehitykselle kuuluvat tärkeät piirteet. Nämä mallintamattomat termit ovat jatkuva suunnittelu, kasvokkainen vuorovaikutus, kehittyvät ja muuttuvat vaatimukset, kestävä työtahti, lyhyet iteraatiot ja julkaisut sekä osapuolten tiivis osallistuminen.

5.1.4 Prosessikuvauksen luonnostelma

Näiden edellisessä alaluvussa esitettyjen ratkaisun toteuttamisen vaiheiden lopputuloksena kehitettiin luonnos prosessikuvauksesta. Tämä luonnos on esitettynä kuviossa 2. Tämän luonnostelman pohjalta voidaan aloittaa luonnoksen arvioiminen ja sen jatkokehitys.

KUVIO 2 Luonnos prosessikuvauksesta



5.2 Luonnostellun prosessikuvauksen arviointi asiantuntijahaastatteluilla

Tässä alaluvussa käydään läpi toteutetun ratkaisun arvioiminen haastattelujen perusteella. Arvioinnilla voidaan varmistaa, että ratkaisu vastaa sille asetettuun ongelmaan. Luvussa esitellään ensin haastattelun toteuttaminen ja kerätyn aineiston analysointi, ja sen jälkeen käydään läpi haastattelujen tuloksia.

5.2.1 Haastattelujen toteuttaminen ja analysointi

Suunnittelutieteellisen tutkimusmenetelmän prosessiin kuuluu olennaisesti ongelmaan tuotetun vastauksen, eli artefaktin arvioiminen jollain keinolla. Arvioinnilla pyritään osoittamaan ja varmistamaan, että artefakti ylipäätään vastaa tarkoitustaan ja arvioimaan, kuinka hyvin se tässä onnistuu (Peffers ym., 2007.) Tässä tutkielmassa tuotetun artefaktin, eli asiakkaalle suunnatun ketterän ohjelmistokehityksen prosessikuvauksen, arvioiminen toteutetaan haastatteluilla.

Haastatteluihin päädyttiin, sillä haastattelujen avulla pystytään keräämään tarkempia kokemuksia ja tulkintoja tutkittavasta aiheesta. Haastattelujen kautta toteutettu tiedonkeruu on erityisen perusteltua silloin, kun tiedossa on, että haastattelu saattaa tuottaa hyvin erilaisia vastauksia. Samalla pystytään esittämään myös tarkentavia lisäkysymyksiä haastateltavalle ja siten saada mahdollisesti tarkempia vastauksia (Hirsjärvi ym., 2009.) Tarkemmin haastattelun tyypiksi valittiin puolistrukturoitu teemahaastattelu. Tämän avulla haastattelua pystytään tarkemmin ohjaamaan valmiiksi määriteltyjen teemojen avulla ja siten varmistamaan, että haastattelussa käsitellään tutkimuksen kannalta olennaisia asioita. Näin se antaa niin haastateltavalle kuin haastattelijalle mahdollisuuden avoimeen keskusteluun, mutta kuitenkin siten, että haastattelussa on määritelty rakenne, joka keskustelua ohjaa (Hirsjärvi ym., 2009.)

Tarkoituksena oli haastatella IT-alan ammattilaisia mahdollisimman kattavasti eri rooleista. Pyrkimyksenä oli haastatella sellaisia henkilöitä, joilla on tietämystä ketterästä ohjelmistokehityksestä ja asiakkaan kanssa toimimisesta projekteissa. Haastateltavien henkilöiden löytäminen haastatteluun osoittautui hieman hankalaksi kesälomien johdosta, mutta haastatteluun saatiin valittua 8 IT-alan ammattilaista Tamperelaisesta IT-yrityksestä, jossa myös tutkielman kirjoittaja työskentelee. Haastattelun tavoitteita oli kolme. Ensin haastattelussa pyrittiin vielä varmistamaan tutkielman tutkimusongelman paikkansapitävyyttä, sillä tutkimuskirjallisuus aihealueesta on niukkaa ja siten tutkimuksen pohjautuvaa perustelua ei ollut riittävän kattavasti. Toisena tavoitteena oli yleisesti haastattelujen avulla yleisesti arvioida kehitetyn prosessikuvauksen vastaaminen sille asetettuun ongelmaan sekä tavoitteisiin. Kolmantena haastatteluissa pyrittiin saamaan mahdollisimman paljon kehitysideoita prosessikuvaukseen,

jotta prosessikuvauksesta saataisiin jatkokehitettyä näiden esiintyneiden palautteiden perusteella parempi versio.

Haastateltaviin otettiin yhteyttä sähköpostitse lähettämällä heille saatekirje sekä haastattelurunko (Liite 6). Haastattelut järjestettiin vuoden 2019 heinäkuussa kullekin haastateltavalle sopivana ajankohtana. Yhteensä haastatteluja toteutettiin 8 kappaletta. Jokainen haastattelu toteutettiin yksilöhaastatteluina. Yksilöhaastatteluihin oli varattu 30-45 minuuttia aikaa ja keskimäärin ne kestivät 39 minuuttia. Haastattelut nauhoitettiin, jotta niitä oli helpompi analysoida jälkikäteen. Haastattelujen litterointi toteutettiin referoivana litterointina, jossa haastattelu purettiin suurpiirteisesti ranskalaisia viivoja käyttäen ja pääpointteihin (Saaranen-Kauppinen & Puusniekka, 2018).

Haastatteluiden analyysissä oli tärkeintä tunnistaa haastatteluissa esiintyneet asiat ja karsia merkityksettömämmät asiat. Haastattelun analysointi toteutettiin käyttämällä samoja teemoja kuin millä haastattelu oli toteutettu. Näiden teemojen tarkoituksena oli jaotella haastattelu ja tutkimukselle olennaiset asiat neljään eri teemaan. Nämä teemat olivat ongelman todentaminen, prosessikuvauksen kehitysideat ja yleinen arvio prosessikuvauksesta ja sen hyödyllisyydestä. Haastatteluun osallistuvat henkilöt numeroitiin osallistumisjärjestyksessä, ja kunkin osallistujan vastaukset ovat esitetty kyseisen numeron perusteella. Haastattelun tulokset jäsenneltiin haastattelun teemojen mukaan kukin erillisiin taulukoihin. Taulukoiden esitystavassa otettiin mallia Pulkkisen ja Kapraalin (2015) tutkimuksessa käyttämästä esitystavasta, jossa arvioinnit esitettiin plus ja miinus -merkein esittämään samaa mieltä tai eri mieltä väitteen kanssa.

Haastateltujen taustatiedot on esitetty taulukossa 7. Haastateltuja oli yhteensä kahdeksan kappaletta, ja haastateltujen työnimikkeitä olivat liiketoiminta-arkkitehti, ohjelmistoarkkitehti, ohjelmistosuunnittelija ja projektipäällikkö. Työtehtävät vaihtelivat haastattelijoilla myynnin tukemisesta projektipäällikön tehtäviin ja myös ohjelmistosuunnitteluun ja määrittelyyn. Kokemus IT-alasta vaihteli vastaajilla melko paljon. Vähimmillään IT-alan kokemusta oli 3½ vuotta ja enimmillään 20 vuotta; keskimäärin kokemusta oli 11 vuotta. Kaikki vastaajista olivat tekemisissä asiakkaiden kanssa työssään ja he myös osallistuivat ketterään ohjelmistokehitykseen. Projektien asiakkaiden toimialat vaihtelivat teollisuuden, logistiikan, autoilun, maksamisen, kaupan ja finanssialan välillä. Projektit, joissa haastateltavat olivat haastatteluhetkellä mukana, olivat henkilömäärältään keskimäärin kuuden henkilön projekteja, pienin projekti oli 2–3:n henkilön projekti ja suurin 10–15 henkilön projekti.

TAULUKKO 7 Haastateltujen taustatiedot

Haastateltava	1	2	3	4	5	6	7	8
Työnimike	Liiketoiminta-arkkitehti	Liiketoiminta-arkkitehti	Ohjelmistoarkkitehti	Ohjelmistosuunnittelija	Ohjelmistoarkkitehti	Projektipäällikkö	Liiketoiminta-arkkitehti	Ohjelmistosuunnittelija
Työtehtävät	Myynnin tuki, liiketoimintaratkaisut, projektipäällikkö	Myynnin tuki, liiketoimintaratkaisut, projektipäällikkö	Ohjelmistosuunnittelu ja määrittely, myynnin tuki	Ohjelmistosuunnittelu ja määrittely	Ohjelmistosuunnittelu ja määrittely, projektinhallinta	Projektinhallinta	Myynnin tuki, ohjelmistokehitys ja arkkitehtuuri	Ohjelmistokehitys ja esimiestehtävät
IT-alan kokemus	7-8	16-18	6½	7-8	20	3½	10	18
Tekemisissä asiakkaiden kanssa	+ Paljon	+ Paljon	+ Paljon	+ Paljon	+	+ Paljon	+	+
Asiakkaiden toimialat	Teollisuus, logistiikka	Autoilu, teollisuus, maksaminen, kaupan ala	Autoilu, logistiikka	Kaupan ala, teollisuus	Teollisuus, autoilu	Finanssiala	Teollisuus, kaupan ala	Finanssiala
Osallistuu kett.ohj. proj.	+/- Tarvittaessa	+ Useampaan	+	+	+	+	+	+
Projektin hlö.määrä	3-7	3-4	5-6	3-4	4-5	10-15	2-3	10
Projektin toimiala	Teollisuus	Autoilu	Autoilu	Kaupan ala	Teollisuus	Finanssiala	Teollisuus	Finanssiala

5.2.2 Tutkimusongelman varmentaminen

Yhtenä haastattelujen tavoitteena oli varmentaa tutkimusongelman paikkansapitävyyttä. Taulukossa 8 on esitetty haastateltavien näkemykset tutkimusongelmasta ja siitä, pitävätkö väitteet paikkansa. Haastateltavien vastaukset olivat pääosin kaikki yksimielisiä haastattelukysymysten väittämien kanssa. Näistä vastauksista ilmenee, että useimmiten asiakkaalla ei ole riittävää osaamista ketterästä ohjelmistokehityksestä osallistuessaan projektiin. Muutama haastattelija kuitenkin totesi, että asiakkaiden osaamisessa on hyvin paljon vaihtelua. Haastateltavat olivat myös samaa mieltä siitä, että asiakasta on projektin aikana opastettava ketterään ohjelmistokehitykseen, ja monet haastateltavista korostivat erikseen sitä, että opastamista on toteutettava toistuvasti. Lisäksi haastateltavat kertoivat, että tästä tietämättömyydestä koituu joitain ongelmia, kuten esimerkiksi, ettei asiakas ymmärrä omaa vastuutaan projektissa tai yleisesti ketterää prosessia eikä siihen liittyviä elementtejä. Kysyttäessä, onko yrityksen kokoluokalla tai toimialalla merkitystä asiakkaan osaamistasoon, esittivät haastateltavat arvioitaan kokoluokan vaikutuksesta suuntaan ja toiseen. Sen sijaan haastateltavat olivat yksimielisiä siitä, että merkittävin tekijä on ennemminkin se, millaisia henkilöitä asiakkaan puolelta projektiin osallistuu kuin sitä, minkä alan tai kokoluokan asiakasyrityksestä on kyse. Yleisesti nämä tulokset vielä

vahvistavat sitä ajatusta, että asiakas ei kovin usein ymmärrä riittävästi ketterästä ohjelmistokehitysprojektista.

TAULUKKO 8 Haastateltujen käsitys tutkimusongelman paikkansapitävyydestä

Haastateltu	1	2	3	4	5	6	7	8
Asiakkaalla ei yleensä riittävää osaamista	+	+ Vaihtelevasti	+ Usein ei tiedä mitään	+ Vaihtelevasti	+ Ei välttämättä tarvitsekaan, plussaa jos tietää	+	+/- Ehkä, vaihtelee hyvin paljon	+
Joutuu opastamaan projektissa	+ Usein	+ Usein	+/- Ehkä	+ Usein	+	+ Usein	+ Jonkin verran	+ Toisinaan
Osaamisen puutoksesta koituu ongelmia	+	+ Asiakas ei ymmärrä vastuuttaan	+ Ei ymmärretä ketterää prosessia Ei keskitytä olennaisiin asioihin	+ Ei ymmärretä ketterää prosessia ja käytänteitä	+ Muutosvastarintaa Ei löydy yhteistä ymmärrystä	+ Ei ymmärrä ketterää prosessia tai käytänteitä Epäselvyys mitä tehdään milloinkin	+ Ei ymmärrä vastuun määrää Vastuuhenkilöllä ei ole riittävästi päätäntävaltaa	+ Iterointia ei ole sisäistetty, palataan vesiputousmalliin Jatkuva suunnittelu haasteellista
Asiakasyrityksen kokoluokan tai toimialan merkitys	+/- Henkilöistä kiinni Teollisuudessa ei ehkä niin valveutuneita	+/- Henkilöistä kiinni	+/- IT-alan firmat tietävät enemmän Isot vanhemmat yritykset ei välttämättä niin hyvin	+/- Henkilöistä kiinni	- Henkilöistä kiinni	- Muutosten tekeminen suuremman kokoluokan yrityksen projekteissa vaikeampaa	+/- Isommissa yleensä oma IT-osasto, jotka osaavat Teollisuuden ala ehkä hieman kankeampi	+/- IT-alan yritykset osaavat yleensä, muut ei välttämättä Isommissa projekteissa osataan yleensä vähän paremmin

5.2.3 Yleinen arvio luonnostellusta prosessikuvauksesta

Haastatteluiden tavoitteena oli myös saada yleinen arvio prosessikuvauksesta ja siitä, miten se vastaa sille asetettuihin tavoitteisiin. Taulukossa 10 on esitetty yleiset arviot haastattelussa esitetystä prosessikuvauksen luonnoksesta. Haastateltavat olivat sitä mieltä, että prosessikuvaus yleistää riittävän hyvin ketterän ohjelmistokehitysprojektin ja kuvaus vastaa käytäntöä. Kaikki haastateltavat olivat myös sitä mieltä, että kyseinen luonnos prosessikuvauksesta on soveltuva asiakkaan käyttöön. Haastateltavat kokivat, että prosessikuvaus ei vielä sellaisenaan ollut riittävän yksinkertainen ja helposti ymmärrettävä, mutta palautteissa annettujen korjausehdotusten huomioimisen jälkeen siitä tulisi sellainen. Toisaalta puolet haastateltavista oli sitä mieltä, että prosessikuvaus oli jo sellaisenaan riittävä. Kaikki haastatelluista olivat sitä mieltä, että prosessikuvausta

voisi käyttää oikeissa projekteissa asiakkaan perehdyttämisessä ketterään ohjelmistokehitysprojektiin. Puolet haastateltavista koki, että kuvausta tulisi kuitenkin visuaalisesti vielä parannella ennen käyttöönottoa. Prosessikuvauksen selitetekstiliite nähtiin tarpeellisena ja varsinkin selitteksteissä esiintyvät ketterän ohjelmistokehityksen piirteet nähtiin erittäin tärkeinä ketterän ohjelmistokehityksen ymmärtämisen kannalta. Yleisesti prosessikuvauksesta oltiin sitä mieltä, että tämänkaltaisella prosessikuvauksella voidaan taata yhteisymmärrys asiakkaan ja tiimin välillä, mikä helpottaa asiakkaan osallistumista projektiin ja prosessikuvaus nähtiin erittäin tärkeänä työkaluna.

TAULUKKO 9 Haastateltujen yleinen arvio prosessikuvauksesta

Haastateltu	1	2	3	4	5	6	7	8
Yleistää riittävän hyvin ja vastaa reaalimaailmaa	+ Ihannetilan, jota tavoitellaan	+	+ Projekteissa paljon eroja	+ Pääpiirteittäin, projekteissa paljon eroja	+ Pääpiirteittäin	+	+	+ Kun toteutetaan kokonaisvaltaisesti
Soveltuu asiakkaalle	+	+	+	+	+	+	+	+
Riittävän yksinkertainen ja helposti ymmärrettävä	+/-, Muutosten jälkeen Vaatii perusymmärryksen	+/- Muutosten jälkeen	+/- Muutosten jälkeen	+/- Yksinkertaistamisen jälkeen	- Visuaalisen yksinkertaistamisen ja selkeyttämisen jälkeen	+ Läpikäytynä	+ Hyvä abstraktiotaso asiakkaalle Selitetekstien avulla	+ Läpikäytynä
Voisi käyttää projekteissa	+/- Visuaalisesti paranneltuna	+	+/- Muutosten jälkeen	+/- Muutosten jälkeen ehdottomasti	+/- Visuaalisesti paranneltuna	+/- Visuaalisesti paranneltuna	+	+
Selitetekstiliite tarpeellinen	+	+ Ehdottomasti	+ Ehdottomasti	+ Ehdottomasti	+ Pakollinen	+ Ehdottomasti	+ Ehdottomasti	+
Selitetekstit piirteistä tarpeellinen	+ Erittäin tärkeitä	+ Erittäin tärkeitä	+ Erittäin tärkeitä	+ Erittäin tärkeitä	+	+ Erittäin tärkeitä	+ Ehdottomasti	+
Muut kommentit	Läpikäytynä asiakkaan kanssa taataan yhteisymmärrys	Prosessikuvaukselle on tarvetta, tuo vakuuttavuutta ja riskit pienenevät	Asiakkaalle tärkeintä tietää missä mennään		Asiakkaalle tärkeintä on tietää, miksi asiat tehdään ja miten tehdään Yhteisymmärrys tärkeintä		Projektin alussa pitäisi käydä yhdessä läpi projektin kulku Yhteisymmärrys tärkeintä	Kehitysjonot ovat hyvä lisä prosessikuvauksessa

Tehtyjen haastatteluiden perusteella on selvää, että tutkielmassa esitetty ongelma asiakkaan puutteellisista tiedoista on olemassa, eli asiakkaalla ei välttä-

mättä ole riittävää tietotaitoa tai käytännön osaamista ketterästä ohjelmistokehityksestä. Prosessikuvaus koettiin yleisesti hyväksi tavaksi tutustuttaa asiakas ketterään ohjelmistokehitykseen. Prosessikuvaus oli valtaosan mielestä riittävän hyvällä tasolla yleisesti ja moni koki prosessikuvauksen erittäin tarpeelliseksi. Muutamien muutosehdotusten jälkeen jokainen haastateltava oli sitä mieltä, että prosessikuvaukselle olisi oikeaa käyttöä projektissa ja sitä voisi hyödyntää projekteissa asiakkaan perehdytykseen.

5.2.4 Muutosehdotukset luonnosteltuun prosessikuvaukseen

Haastatteluissa kerättiin muutosehdotuksia kehitettyyn prosessikuvaukseen, jotta se vastaisi paremmin käytäntöä ja samoin sille asetettuja tavoitteita. Nämä haastatteluissa esiintyneet muutosehdotukset on esitetty taulukossa 10. Näiden palautteiden pohjalta tuotetaan prosessikuvauksen luonnoksesta varsinainen prosessikuvauksen valmis versio. Haastatteluiden analyysin pohjalta pystyttiin koostamaan muutokset, jotka prosessikuvaukseen on syytä toteuttaa. Taulukossa 10 esitellyt haastatteluissa esitetyt muutosehdotukset, on vielä tarkemmin jäsenneltynä alla. Näiden ideoiden perusteella pystyttiin luomaan käsitys prosessikuvaukseen muutettavista asioista. Muutettaviin tekijöihin valittiin ne ehdotukset, jotka esiintyivät vähintään puolella haastateltavista. Seuraavaksi esitetään prosessikuvaukseen toteutetut muutokset. Kunkin muutosehdotuksen esiintymisen määrä järjestetyissä haastatteluissa on osoitettu muutosehdotuksen otsikon lopussa suluissa.

Jatkuvan suunnittelun muutokset (6/8)
Haastateltavien tulkitessa prosessikuvausta, jatkuvaa suunnittelua kuvaava tapahtuma tuotti ongelmia prosessin ymmärtämisessä. Se miellettiin osittain epäselkeäksi ja sen koettiin kuuluvan iteraation aikana tapahtuvaksi toiminnaksi. Tästä syystä katsottiin, että jatkuva suunnittelu tulisi muuttaa iteraation aikana tapahtuvaksi tapahtumaksi.

Yleinen yksinkertaistaminen, korostaminen ja selkeyttäminen (5/8)
Prosessikuvaus koettiin osittain hieman epäselkeäksi. Palautteena esitettiinkin, että prosessikuvauksessa tulisi kiinnittää huomiota sen yksinkertaisuuteen ja siihen, että kuvauksessa pystyttäisiin visuaalisesti korostamaan kuvauksen eri osioita. Tapahtumien ja kehitysjonojen ulkoasua toivottiin korostettavan enemmän. Iteraation toistuvuutta ja jatkuvien muutosten tapahtumista kehitysjonolle toivottiin myös visualisoitavan paremmin. Korostamiseen ja parempaan visualisoimiseen ehdotettiin värien käyttöä.

Esiselvitysvaiheen puuttuminen (4/8)
Haastatteluissa nousi esiin esiselvitysvaiheen puuttuminen prosessikuvauksesta. Puolet haastateltavista koki, että prosessin alkuvaihetta tulisi korostaa enemmän. Ratkaisuksi ehdotettiin, että kuvaukseen voisi lisätä uuden esiselvitystapahtuman prosessin alkuun. Tätä perusteltiin sillä, että jokaisessa projek-

tissa on aina jonkinlainen alkusuunnittelu, jossa muodostetaan alustava käsitys toteutettavasta tuotteesta.

Asiakkaan vastuiden epäselvä esittäminen (4/8)
Asiakkaan vastuista todettiin, että ne tulisi esittää visuaalisesti selkeämmin. Tähän ehdotettiin värejä, joilla asiakas pystyisi helposti ja nopeasti näkemään tapahtumien tärkeydet. Samoin koettiin, että alkuperäinen kolmen tason tärkeysasteikko on liian tarkka, ja ratkaisuksi ehdotettiin sen muuttamista kahteen tasoon. Perusteluna tälle oli se, että asiakkaalle jokainen prosessikuvauksessa esitetty tapahtuma on vähintään jonkin verran olennainen.

Julkaisun kehitysjonon nimi (4/8)
Haastateltavat kokivat julkaisun kehitysjonon nimen harhaanjohtavaksi ja sellaiseksi, jota ei kovinkaan usein käytetä. Tästä syystä nähtiinkin tarve osion nimenmuutokselle siten, että osiota ei käsitettäisi erillisenä kehitysjonona vaan pelkästään listana valmiita tehtäviä. Paremmaksi nimeksi ehdotettiin ”Valmiit ja hyväksytyt tehtävät”.

Selitetekstien tärkeiden piirteiden korostaminen (4/8)
Selitetekstit koettiin erittäin tärkeäksi osaksi prosessikuvausta. Selitteksteissä esitetyt ketterän ohjelmistokehityksen piirteet koettiin kaikkein olennaisimmiksi käsitteiksi liitteessä. Tästä syystä toivottiin, että nämä piirteet esitettäisiin liitteen alussa.

Muut toteuttamattomat ideat (1–3/8)
Haastatteluissa esiintyi myös monia muita muutostarpeita, mutta ne olivat kuitenkin yksittäisten, tai korkeintaan kahden henkilön antamia ehdotuksia. Esimerkiksi kuvauksesta koettiin, että julkaisuun liittyen voisi esittää myös erinäiset julkaisuympäristöt. Jotkut näkivät julkaisutapahtuman iteraatiosta erillisenä prosessina. Osa koki tapahtumien kestojen esittämisen tärkeäksi. Työmääräarviot ja budjetin esittäminen koettiin myös oleelliseksi. Ehdotettiin myös vaihtoehtoiseksi kuvaustavaksi projektin kuvaaminen konkreettisen esimerkin kautta. Koska nämä ehdotukset esiintyivät vain harvoin, ne jätettiin huomioimatta.

TAULUKKO 10 Haastateltujen esittämät muutosehdotukset prosessikuvaukseen

Haastateltu	1	2	3	4	5	6	7	8
Prosessikuvauksen ja selitetekstiliitteen muutosehdotukset	- Esiselvitysvaihe alkuun - Asiakkaalle olennaiset tapahtumat selvemmin esille esim. värein - Julkaisun kehitysjono nimi selvemmäksi esim. ”Valmiit tehtävät” - Testiympäristön voisi esittää - Kehitysjonon korostaminen visuaalisesti - Kehitysjonon tehtävien prioriteetti selkeämmäksi - Aikataulutus ehkä esille - Selitetekstit yleisistä piirteistä tulee näyttää ennen muita tekstejä	- Esiselvitysvaihe alkuun - Jatkuva suunnittelu yhdistetään iteraatiotapahtumaan - Asiakkaalle olennaiset tapahtumat selvemmin esille - Iteraatiotapahtumalle uusi nimi - Julkaisun kehitysjono nimi selvemmäksi esim. ”valmiit tehtävät” - Testiympäristön voisi esittää - Minimituoteajattelu (MVP) tekstiin - Budjetti ja työmääräarvio vähintään tekstiin - Asiakkaalle suunnatun jatku-van raportoinnin tärkeys tekstiin	- Esiselvitysvaihe alkuun - Jatkuva suunnittelu ja julkaisun suunnittelu erilleen prosessista - Asiakkaalle olennaiset tapahtumat selvemmin esille - Julkaisun kehitysjono nimi selvemmäksi esim. ”Valmiit tehtävät” - Kehitysjonojen ulkonäkö yksinkertaisemmaksi - Julkaisun jälkeiset palautteet kehitysjonoon - Tapahtumien kestot voisi esittää - Selitetekstit yleisistä piirteistä tulee näyttää ennen muita tekstejä	- Jatkuvan suunnittelun voi poistaa tai muuttaa - Iteraatiotapahtumalle uusi nimi esim. ”implementaatiovaihe” - Kehitysjonojen ulkonäkö yksinkertaisemmaksi - Osakokonaisuustaso ehkä pois - Aikataulu ehkä priorisoinnin lisäksi - Tapahtumia voi yhdistää tai yksinkertaistaa - Vision voi ehkä poistaa tai muuttaa - Tuotosten kulkua yksinkertaisemmaksi - Iteraatio ja julkaisu erillisiksi prosesseiksi - Voisi esittää myös konkretian kautta - Selitetekstejä yleisistä piirteistä tulee korostaa enemmän	- Jatkuva suunnittelu yhdistetään iteraatiotapahtumaan - Iteraatio ja julkaisu erillisiksi prosesseiksi - Tuotteen kehitysjonon jatkuvan muuttumisen voisi esittää paremmin - Iteraation toistuvuutta voisi korostaa esim. kuvaamalla pyöreänä - Selitetekstit yleisistä piirteistä tulee näyttää ennen muita tekstejä - Tulee selittää miten ja miksi asioita tehdään	- Esiselvitysvaihe alkuun - Jatkuva suunnittelu yhdistetään iteraatiotapahtumaan - Asiakkaalle olennaiset tapahtumat selvemmin esille - Kehitysjonojen ulkonäkö yksinkertaisemmaksi ja selkeämmäksi - Selvemmin esiin jatkuvien muutoksien tapahtuminen kehitysjonoon	- Jatkuvan suunnittelun voisi eriyttää prosessista - Julkaisun kehitysjono nimen voisi muuttaa	- Jatkuvan suunnittelun voisi eriyttää prosessista - Iteraation toistuvuutta voisi korostaa - Selitetekstit yleisistä piirteistä tulee näyttää ennen muita tekstejä

5.3 Luonnosteltuun prosessikuvaukseen toteutetut muutokset

Prosessikuvauksen korjattuun versioon on tehty edellisessä alaluvussa esitetyt muutosehdotukset. Tässä alaluvussa esitellään nämä toteutetut muutokset. Seuraavassa luvussa esitellään näiden muutosten pohjalta tehty lopullinen versio prosessikuvauksesta.

Jatkuva suunnittelu yhdistettiin aikaisempaan iteraatio -tapahtumaan ja samoin tapahtuman nimi muutettiin ”Kehitys ja suunnittelu” -tapahtumaksi. Tällä muutoksella pyritään kuvaamaan paremmin projektin aikaista jatkuvasti tapahtuvaa suunnittelua ja lisäksi korostamaan iteraation varsinasta kehitysvaihetta. Samaiseen tapahtumaan lisättiin uusi nuoli ”Visio tuotteesta” -tuotoksesta, jolla esitetään vision käyttämistä kehityksen ja suunnittelun aikana. Samoin lisättiin uusi tuotosnuoli myös tuotteen kehitysjonoon, jolla esitetään jatkuvan suunnittelun aikaansaamia uusia tehtäviä ja tarkennuksia, joita projektin aikana ilmenee.

Prosessikuvauksen alkuun lisättiin esiselvitys -tapahtuma. Tapahtumalla esitetään projektin alkuvaiheessa tapahtuvaa alustavaa suunnittelua. Tämä on varsinaisesta iteraatiosta erillään ja tapahtuu vain projektin alkuvaiheessa ja tässä tapahtumassa tuotetaan alustavat tuotokset tuotteen kehitysjonolle. Samoin julkaisun kehitysjonon nimi muutettiin valmiisiin ja hyväksyttyihin tehtäviin sekä sen visuaalista ulkoasua muutettiin siten, että se esittää vain tehtäväpinoa eikä varsinaista kehitysjonoa.

Prosessikuvaa pyrittiin yksinkertaistamaan ja selkeyttämään monin tavoin. Tämä toteutettiin käyttämällä värejä tapahtumissa ja tuotoksissa. Kehitysjonoja ja niiden sisältöjä väritettiin eri värein, jotta eri osakokonaisuudet erottuvat paremmin. Tapahtumat väritettiin kahdella eri värillä, jotka osoittavat tapahtuman olennaisuuden asiakkaalle. Tässä käytettiin kahta tasoa, sillä haastattelujen perusteella koettiin, että kaikki tapahtumat ovat ainakin jossain määrin olennaisia asiakkaalle. Samoin tekstillä ”iteraatio” kuvan alareunassa pyrittiin korostamaan iteraation toistuvuutta. Käsitteiden numeroinnista luovuttiin yksinkertaistamisen vuoksi.

Seliteteksteihin lisättiin ja muutettiin uuden prosessikuvauksen mukaisesti uusi tapahtuma ”Esiselvitys” ja muuttunut ”Kehitys ja suunnittelu”. Samoin ”Julkaisun kehitysjono” termistä luovuttiin ja ”Hyväksymättömät ja keskeneräiset tehtävät” -osioon lisättiin uudet käsitteet ”Valmiit” ja ”Uudet” – tehtävät. Seliteteksteistä poistettiin numeroinnit ja ne järjestettiin uudelleen aakkosittain. Selitetekstin tärkeät piirteet siirrettiin selitetekstiliitteen alkuun, sillä ne koettiin kaikkein tärkeimmiksi asioiksi seliteteksteissä.

6 VALMIIN PROSESSIKUVAUKSEN LÄPIKÄYNTI

Tässä luvussa on esitetty tutkimuksessa kehitetyn prosessikuvauksen lopullinen versio. Samoin luvussa esitetään mahdollisia käyttötilanteita prosessikuvauksen käyttöön sekä avataan prosessikuvauksen osat ja niiden merkitykset. Tämän luvun tarkoituksena onkin pystyä vastaamaan suunnittelutieteellisen tutkimusmenetelmän demonstroivaan ja viestiviin vaiheisiin. Valmis prosessikuvaus on esitettynä tämän luvun lopussa kuviossa 3. Tämän lisäksi prosessikuvaukseen liitettävä selitetekstiliite on esitettynä liitteessä 5.

Seuraavaksi käsitellään mahdollisia tilanteita, joissa prosessikuvausta voitaisiin hyödyntää. Koska prosessikuvaus on toteutettu asiakkaalle, on sen kohderyhmänä pääosin projektiin asiakasroolissa toimivat henkilöt. Prosessikuvauksessa on kaksi selvää käyttötarkoitusta. Ensinnäkin se toimii projektiin osallistuvan asiakkaan oppaana ketterän ohjelmistokehitysprojektin käytänteisiin ja tapaan toimia, sekä pyrkii varmistamaan asiakkaan riittävän tietotaidon aiheesta. Toiseksi prosessikuvaus varmistaa, että asiakkaalla ja kehitystiimillä on yhteisymmärrys ketterän ohjelmistokehitysprojektin asioista.

Prosessikuvausta voidaan hyödyntää monissa eri vaiheissa. Sitä voidaan hyödyntää silloin kun projekti on alkamaisillaan ja tällöin asiakkaan puolelta osallistuvat henkilöt tulee perehdyttää ketterän ohjelmistokehitykseen. Toisaalta sitä voidaan hyödyntää myös esimerkiksi silloin kun projektin aikana asiakkaan puolelta osallistuu projektin uusi henkilö. Vastaavasti prosessikuvausta voidaan käyttää ohjenuorana koko projektin aikana.

Seuraavaksi prosessikuvaus käydään kokonaisuudessaan läpi ja esitetään miten prosessikuvausta tulisi tulkita. Kun prosessikuvausta luetaan vasemmalta oikealle, havaitaan sen alkavan tuotteen visiosta. Prosessikuvaus esittää, että tuotteen visiosta ammennetaan tietoa niin esiselvitykselle kuin myös myöhemmin iteraatiolle. Ensimmäisenä tapahtumana on esiselvitys, josta koostetaan alustavat ideat tuotteen kehitysjonolle. Tuotteen kehitysjonon ideat on pilkottu osakokonaisuuksiin, joista ne on edelleen pilkottu toiminnallisuuksiin ja tarkempiin tehtäviin. Esiselvityksen jälkeen projekti etenee iteraatiovaiheeseen, joka alkaa iteraation suunnittelulla. Iteraation suunnittelussa ammennetaan tuotteen kehitysjonosta töitä, jotka pilkotaan tarkemmiksi tehtäviksi ja niistä

koostetaan iteraation kehitysjono. Iteraation kehitysjono sisältää siis kaikki ne tehtävät, joita kehittäjien on tarkoitus työstää iteraation kehityksen aikana.

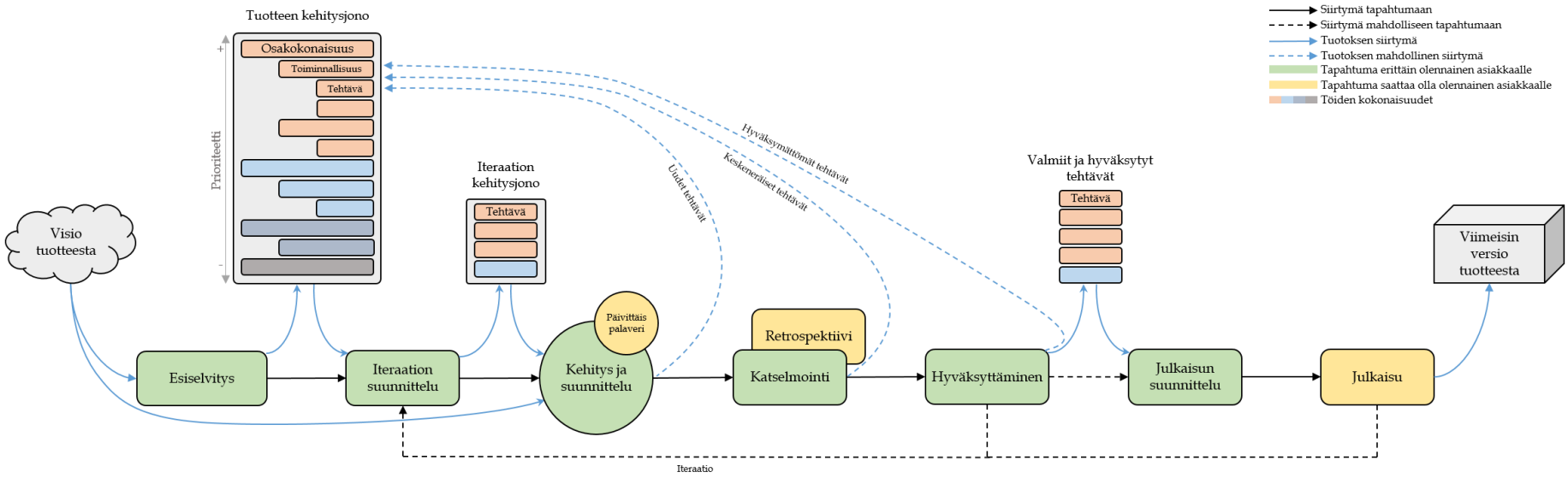
Iteraation suunnittelun jälkeen alkaa kehitys ja suunnittelu -tapahtumavaihe. Tällöin kehitystiimi työstää iteraatiolle valittuja tehtäviään iteraation kehitysjonolta, ja tarkoituksena on saada ne valmiiksi iteraation aikana. Samalla asiakas ja muut projektin henkilöt ideoivat, mitä asioita tuotteen kehitysjonolta vielä puuttuu. Näitä ideoita lisätään uusiksi tehtäviksi tuotteen kehitysjonolle. Kehitystiimi ylläpitää tietoa projektin kulusta järjestetyissä päivittäispalavereissa. Myös asiakas saattaa olla näissä mukana.

Varsinaisen kehitys ja suunnittelu -vaiheen loputtua katselmoidaan asiakkaan kanssa yhdessä, mitä tehtäviä kehityksen aikana saatiin tehtyä, ja mitkä tehtävät jäivät kesken. Keskeneräiset tehtävät palautuvat tuotteen kehitysjonolle odottamaan seuraavaa iteraation suunnittelua. Retrospektiivissä käydään läpi iteraation kulkua ja pyritään reflektoimaan, miten työt ovat sen aikana edistyneet. Asiakas osallistuu mahdollisesti myös tähän tapahtumaan.

Asiakkaan vastuulla on varmistaa ja hyväksyä kehitetyt tehtävät. Näin voidaan varmistaa, että kehitetyt tehtävät vastaavat asiakkaan haluamaa lopputulosta. Jos asiakas on sitä mieltä, että toteutettu tehtävä on tehty väärin tai puutteellisesti, palautuu se takaisin tuotteen kehitysjonolle odottamaan seuraavaa iteraation suunnittelua. Hyväksytyt tehtävät jäävät odottamaan julkaisun suunnittelua.

Hyväksyttämisen jälkeen voidaan mahdollisesti suunnitella seuraavaa julkaisua. Tätä ei välttämättä joka iteraatiossa tapahdu, joten hyväksyttämisen jälkeen voidaan aloittaa myös uusi iteraation suunnittelu. Kuitenkin, jos koetaan, että julkaisu on tarpeellinen, pyritään koostamaan valmiista tehtävistä järkevä kokonaisuus julkaistavaksi. Julkaisun suunnittelun jälkeen kehitystiimi julkaisee uuden version tuotteesta ja tämän jälkeen uusi iteraatio voi alkaa. Uusi iteraatio aloitetaan iteraation suunnittelusta ja siten seuraava iteraatio voidaan aloittaa.

KUVIO 3 Valmis prosessikuvaus



7 POHDINTOJA TUTKIELMASTA

Tässä luvussa käydään läpi yleisesti tutkielmassa esiintyneiden tuotosten merkitystä, tämän tutkielman rajoitteita sekä mahdollisia jatkotutkimusaiheita. Ensimmäisessä alaluvussa käsitellään tutkielmassa kehitettävän ratkaisun tavoitteita niiden toteutumisen kannalta. Toisessa alaluvussa käydään läpi tutkielman tulosten merkitystä ja viimeisessä alaluvussa pohditaan tutkielman rajoitteita sekä jatkotutkimusaiheita.

7.1 Prosessikuvauksen tavoitteiden toteutuminen

Prosessikuvaksen toteuttamisen jälkeen on tärkeää peilata sitä vielä sitä sille asetettuihin tavoitteisiin, jotka ovat esitettynä taulukossa 6. Prosessikuvauksen tuli pohjautua tutkimuksista koostettuun tietoon ja sen tuli esittää ketterän ohjelmistokehityksen tyypillinen prosessi ja vaiheet sekä yleisimmät elementit. Prosessikuvauksen esitystapa tuli rajoittaa siten, että se koostuu vain asiakkaalle olennaisimmista asioista, ja että se on riittävän yksinkertainen sekä helposti ymmärrettävä asiakkaalle.

Jotta näihin tavoitteisiin pystyttiin vastaamaan, prosessikuvauksen kehittämistä toteutettiin tutkimus edellä. Elementtejä ja tietoja, joita käytettiin prosessikuvauksessa, hyödynnettiin tämän tutkielman kirjallisuuskatsauksessa toteutettujen osioiden avulla. Samoin prosessikuvaukseen liitetyt selitetekstit koostettiin tutkimuskirjallisuuteen pohjautuen. Kirjallisuuskatsauksen teoriaosuuden ja toteutettujen analyysien pohjalta pystyttiin prosessikuvauksessa esittämään tyypillisen projektin kulku ja siihen liittyvät vaiheet ja elementit. Nämä rajattiin vielä asiakkaalle olennaisiin asioihin.

Prosessikuvaus pyrittiin toteuttamaan kaikin puolin projektin asiakas huomioiden. Prosessikuvauksessa esitetyt asiat pyrittiin kuvaamaan sellaisella abstraktiotasolla, joka tukee asiakkaan ymmärrystä ketterästä ohjelmistokehitysprojektista. Samoin selitetekstien avulla pyrittiin siihen, että prosessikuvauksen avulla asiakkaan on mahdollista saada riittävä tietämys ketterän ohjelmis-

tokehitykseen liittyvistä tekijöistä. Tutkielmassa toteutetut arviointihaastattelut prosessikuvauksesta tukivat näiden tavoitteiden toteutumista, ja haastattelijoiden palautteiden perusteella pystyttiin prosessikuvausta parantamaan vastaamaan vielä paremmin näihin tavoitteisiin. Nämä haastatteluissa esitettyjen korjausehdotukset sisältävä valmis versio prosessikuvaus pystyykin siten vastaamaan paremmin sille asetettuihin tavoitteisiin ja vastaamaan siten myös sen käyttötarpeeseen.

7.2 Tutkielman tulosten merkitys

Tämän tutkielman varsinaisena lopputuloksena syntyi asiakkaalle suunnattu ketterän ohjelmistokehitysprojektin prosessikuvaus. Tämän prosessikuvauksen myötä tuotettiin tutkielmassa myös yleistys ketterän ohjelmistokehityksen prosessista sekä koostettiin tyypillisimmät elementit liittyen ketterään ohjelmistokehitykseen. Tämä kehitetty prosessikuvaus pyrkii vastaamaan tutkielmassa käsiteltyyn tutkimusongelmaan, jossa asiakkaalla ei välttämättä ole riittävää tietotaitoa ja osaamista ketterästä ohjelmistokehitysprojektista ja näin ollen se hankaloittaa projektin etenemistä ja onnistumista. Tehtyä kattavaa tutkimusta väittämästä ei ole riittävästi, joten tätä väittämää myös varmennettiin tutkielmassa järjestettyjen haastattelujen avulla. Näissä haastatteluissa oltiin yksimielisiä siitä, että ongelma on olemassa. Tästä syystä onkin tärkeää kiinnittää huomiota siihen, miten asiakasta pystyttäisiin perehdyttämään mahdollisimman hyvin projektiin ja sen ketterään luonteeseen.

Tutkimusta tehdessä ja olennaisia tutkimuksia aihealueeseen liittyen etsittäessä huomattiin, että ketterä ohjelmistokehitys on yhä edelleen hyvin abstrakti käsite, eikä siitä tutkimuksissa ole kovinkaan hyvin pyritty tuottamaan yleistyksiä tai kattavampia selityksiä. Tämä tutkielma kuitenkin osoittaa, että ketterän ohjelmistokehityksen yleistäminen, eli yhteisen käsityksen muodostaminen ilmiöstä, on mahdollista, ja että ketterän ohjelmistokehitysprojektin yleisimmät elementit pysytään osoittamaan. Tutkimusten pohjalta pystytäänkin muodostamaan yleistäviä prosessin vaiheita ja löytämään tyypillisimmät ketterään ohjelmistokehitykseen liittyvät elementit. Tämä tutkielma osoittaa, että ilmiön yleistäminen on mahdollista ja myös tarpeellista, sillä vaikka ketteriä menetelmiä on useita, koostuvat ne pääpiirteittäin hyvin samankaltaisista asioista.

Yleisesti tutkimuskirjallisuudessa ei myöskään ole hyvin otettu huomioon asiakkaan näkökulmaa ketterässä ohjelmistokehitysprojektissa. Tämä on omituista, sillä ketterä ohjelmistokehitys kuitenkin hyvin vahvasti korostaa asiakkaan roolia projekteissa. Tästä huolimatta tutkimuskirjallisuudessa on aivan liian vähän kiinnitetty huomiota esimerkiksi siihen, miten asiakasta pystyttäisiin perehdyttämään projektiin mahdollisimman hyvin. Myöskään aiemmissa tutkimuksissa ei kovinkaan paljoa ole näyttöä siitä, miten asiakas toimii ketterässä ohjelmistokehitysprojektissa projektin jäsenenä.

Tutkielmassa toteutetun haastattelun perusteella pystyttiin todentamaan, että asiakkaan roolilla on suuri merkitys projektin kannalta. On enemmän kuin

toivottavaa, että asiakas sitoutuu ja osallistuu projektiin mahdollisimman hyvin. Asiakkaan puutteelliset tietotaidot ketterästä ohjelmistokehityksestä hankaloittavat projektin etenemistä ja tuottavat yleisesti projektille hankaluuksia. Tästä syystä onkin tärkeää taata asiakkaalle riittävä perehdytys ketterään ohjelmistokehitykseen ja siihen, mitä se pitää sisällään. Tämä tutkielma antaakin luodulla prosessikuvauksella tähän oman vastauksensa.

Tässä tutkielmassa kehitetty prosessikuvaus vastaa kahteen tarpeeseen. Ensinnäkin se yleistää ketterän ohjelmistokehityksen nojaamatta liikaa mihinkään tiettyyn ketterään ohjelmistokehitysmenetelmään. Näin ollen se soveltuu projektiin menetelmästä riippumatta, sillä se kuvaa vain tyypillisimmät ketterään ohjelmistokehitykseen liittyvät asiat. Toinen tarve, johon prosessikuvaus vastaa, on se seikka, että prosessikuvaus on toteutettu asiakkaalle. Se siis esittää projektin kulun ja siihen liittyvät elementit siten, että ne ovat asiakkaalle mahdollisimman helpot ja yksinkertaiset ymmärtää. Tässä nimenomaan kiinnitetään huomiota siihen, ettei asiakas välttämättä ole IT-alan ammattilainen toisin kuin projektin osallistuvat kehittäjät.

Tämä prosessikuvaus ja sen arvioiva haastattelu osoittavatkin, että tämänlaisille yleistyksille on käyttöä ja niille on myös tarvetta. Haastatteluissa suurin osa haastateltavista oli sitä mieltä, että tämänkaltaista prosessikuvausta voitaisiin käyttää oikeissa projekteissa ja niistä oletettavasti olisi hyötyä projektin kannalta. Tämän prosessikuvauksen merkitys korostuu varsinkin varsinaisessa työelämässä, jossa prosessikuvausta voidaan käytännössä hyödyntää. Prosessikuvauksesta voidaan esimerkiksi johtaa tarkempia kuvauksia, joissa esitetään asiaa konkreettisesti esimerkkien avulla. Prosessikuvauksella on merkitystä myös tutkimuksen kannalta, sillä se osoittaa prosessikuvausten hyödyllisyyden työkaluna ketterän ohjelmistokehityksen selittämisessä. Samoin se osoittaa yleisesti, miten prosessikuvausta pystytään kohdistamaan tietyille tarvittaville kohderyhmille niiden tarpeet huomioivalla tavalla.

7.3 Tutkielman rajoitteet ja jatkotutkimusaiheet

Tutkielman toteutuksessa esiintyi joitakin rajoitteita. Kattavaa tutkimustietoa ketterän ohjelmistokehityksen yleisimmistä elementeistä tai yleistetystä prosessista ei esiinny kovinkaan kattavasti. Aihealueesta ei löydy koostavia kattavia tutkimuksia tai kvantitatiivisia, systemaattisia tutkimuksia, jotka helpottaisivat käsittämään yleisimmät elementit ja yleistetyn prosessin, esimerkiksi listaamalla tyypilliset tapahtumat. Tämä tuotti ilmiön yleistämisestä hieman hankalampaa.

Parantamisen varaa jäi asiakkaan roolin tutkimiseen ketterässä ohjelmistokehitysprojektissa. Tämän tutkielman kannalta olisi ollut hyödyllistä, että aiemmissa tutkimuksissa käsiteltäisiin asiakasta tutkimuskohteena, sillä etenkin projektin onnistumisen kannalta asiakas on merkittävässä osassa. Tätä pyrittiin tukemaan tutkielmassa toteutettujen haastattelujen avulla.

Olemassa olevia prosessikuvauksia olisi voitu tutkia vielä kattavammalla otannalla ja esimerkiksi haastatella asiakkaita ja siten varmistaa prosessikuvauksen toimivuus myös asiakkaan näkökulmasta. Tässä tutkimus perustui asiantuntijoiden arvioihin siitä, mikä on heidän näkemyksensä siitä, mikä on asiakkaalle olennaisinta tietää. Yleisesti prosessikuvausta on kannattavaa testata vielä käytännössä.

Jatkotutkimusaiheina voidaan esittää kehitetyn prosessikuvauksen soveltamista käytäntöön oikeissa projekteissa, joissa esimerkiksi projektin alussa asiakas perehdytettäisiin projektiin kuvausta apuna käyttäen. Toisena mahdollisena jatkotutkimusaiheena voisi olla prosessikuvauksen jatkokehittäminen asiakkaiden antamien palautteiden pohjalta. Kolmantena jatkotutkimusaiheena voisi olla selvitys siitä, ovatko kirjallisuuden perusteella tyypillisimmät elementit myös todellisissa projekteissa kaikkein yleisimpiä. Neljäntenä jatkotutkimusaiheena toivotaan yleisesti lisää tutkimusta asiakkaan roolista ketterässä ohjelmistokehitysprojektissa.

8 YHTEENVETO

Tämän tutkielman tutkimusongelmana oli asiakkaan riittämätön tietämys ketterästä ohjelmistokehitysprojektista ja tästä syystä johtuvasta mahdollisesta projektin onnistumisen hankaloitumisesta. Tutkielman tavoitteena oli selvittää, mitä asioita asiakkaan olisi tärkeää tietää ketterästä ohjelmistokehitysprojektista, jotta asiakas pystyisi osallistumaan projektiin mahdollisimman hyvin, ja näin ollen voitaisiin mahdollistaa projektin onnistuminen. Tämän ongelman pohjalta laadittiin yksi päätutkimuskysymys ja kaksi taustoittavaa tutkimuskysymystä.

- Mitä asioita asiakkaan on tärkeää tietää osallistuessaan ketterään ohjelmistokehitysprojektiin?
    - Mitä vaiheita tyypilliseen ketterään ohjelmistokehitysprojektiin kuuluu?
    - Mitä tyypillisimpiä elementtejä ketterä ohjelmistokehitysprojekti pitää sisällään?

Tutkielma koostuu teoreettisesta ja empiirisestä osiosta. Teoreettisessa osuudessa määriteltiin aihepiiriin liittyvän kirjallisuuden avulla keskeisiä käsitteitä, kuten ohjelmistokehitysprojektiin liittyviä termejä sekä tutkimukseen olennaisesti kuuluvaa ketterän ohjelmistokehityksen käsitettä. Teoriaosuudessa toteutettiin myös kirjallisuuteen pohjautuvat analyysit, joilla vastattiin kahteen taustoittavaan tutkimuskysymykseen. Tutkielman empiirisessä osuudessa puolestaan käsiteltiin tutkielman päätutkimuskysymystä.

Ensimmäiseen taustoittavaan tutkimuskysymykseen vastattiin tutkielman kirjallisuuskatsauksessa. Käytettyjen lähteiden ja olemassa olevien prosessikuvausten pohjalta muodostettiin yleistetty käsitys siitä, millainen tyypillinen ohjelmistokehitysprosessi on, ja mitä vaiheita siihen kuuluu. Näistä muodostettiin tyypillisen ohjelmistokehitysprojektin prosessin vaiheistus. Toiseen taustoittavaan tutkimuskysymykseen vastattiin kirjallisuuskatsauksessa toteutetun analyysin perusteella. Analyysin tuloksena saatiin koostettua 28 tyypillisintä ketterään ohjelmistokehitykseen liittyvää elementtiä, jotka jaoteltiin kolmeen eri ka-

tegoriaan: tapahtumiin ja toimenpiteisiin, ominaispiirteisiin sekä työkaluihin ja teknisiin tekijöihin.

Suunnittelutieteellistä tutkimusmenetelmää hyödyntämällä pyrittiin saamaan vastaus päätutkimuskysymykseen. Menetelmän avulla tyypillisesti suunnitellaan ja ratkaistaan havaittu, tarkemmin määritelty ongelma, jollain konkreettisella tuotoksella, joka tämän jälkeen arvioidaan ja demonstroidaan jollain tavalla. Ongelmaan vastaamiseksi tutkielmassa laadittiin uusi prosessikuvaus. Luonnosteltu prosessikuvaus arvioitettiin alan asiantuntijoiden toimesta haastatteluiden avulla, ja haastatteluissa ilmenneiden palautteiden perusteella prosessikuvauksesta luotiin lopullinen versio. Valmis prosessikuvaus pitää sisällään tyypillisen prosessin ja siihen liittyvät tyypilliset elementit, jotka asiakkaan on syytä tietää osallistuessaan ketterään ohjelmistokehitysprojektiin.

Tämä tutkielma osoittaa, että asiakas on erittäin tärkeässä osassa ketterässä ohjelmistokehitysprojektissa ja asiakkaan riittävä tietotaito aihealueesta on syytä varmistaa jokaisessa projektissa. Tutkielmassa luotu asiakkaalle suunnattu ketterän ohjelmistokehitysprojektin prosessikuvaus on kehitetty siten, että se koostuu asiakkaalle olennaisista elementeistä ja se osoittaa ketterän ohjelmistokehitysprojektin tyypillisen kulun ja siihen liittyvät tyypillisimmät käsitteet. Tällä tavoin projektiin osallistuvalla asiakkaalla on keino saada tietoa juuri niistä asioista, jotka ovat asiakkaalle olennaisia ketterässä ohjelmistokehitysprojektissa ja tällöin myös pyritään varmistamaan se, että niin asiakkaalla kuin kehitystiimillä on yhteisymmärrys projektin kulusta ja siihen liittyvistä käsitteistä.

LÄHTEET

Abrahamsson, P., Salo, O., Ronkainen, J., & Warsta, J. (2017). Agile software development methods: Review and analysis. *arXiv preprint* arXiv:1709.08439.

Ahimbisibwe, A., Cavana, R. Y., & Daellenbach, U. (2015). A contingency fit model of critical success factors for software development projects: A comparison of agile and traditional plan-based methodologies. *Journal of Enterprise Information Management, 28(1)*, 7-33.

Al-Zewairi, M., Biltawi, M., Etaiwi, W., & Shaout, A. (2017). Agile software development methodologies: survey of surveys. *Journal of Computer and Communications, 5(05),* 74-97.

Beck, K. (1999). Embracing change with extreme programming. *Computer, (10),* 70-77.

Beck, K., & Gamma, E. (2000). Extreme programming explained: embrace change. *Addison-Wesley Professional*.

Beck, K., Beedle, M., Van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., ... & Kern, J. (2001). Manifesto for agile software development.

Bhalerao, S., Puntambekar, D., & Ingle, M. (2009). Generalizing Agile software development life cycle. *International journal on computer science and engineering, 1(3),* 222-226.

Chow, T., & Cao, D. B. (2008). A survey study of critical success factors in agile software projects. *Journal of systems and software, 81(6),* 961-971.

Conboy, K. (2009). Agility from first principles: Reconstructing the concept of agility in information systems development. *Information systems research, 20(3),* 329-354.

De Azevedo Santos, M., de Souza Bermejo, P. H., de Oliveira, M. S., & Tonelli, A. O. (2011). Agile practices: An assessment of perception of value of professionals on the quality criteria in performance of projects. *Journal of Software Engineering and Applications, 4(12)*, 700.

De Lucia, A., & Qusef, A. (2010). Requirements engineering in agile software development. *Journal of emerging technologies in web intelligence, 2(3),* 212-220.

Dingsøyr, T., Nerur, S., Balijepally, V., & Moe, N. B. (2012). A decade of agile methodologies. *Journal of Systems and Software, 85(6),* 1213-1221.

Dybå, T., & Dingsøyr, T. (2008). Empirical studies of agile software development: A systematic review. *Information and software technology, 50(9-10),* 833-859.

Elghariani, K., & Kama, N. (2016, August). Review on Agile requirements engineering challenges. In *2016 3rd International conference on computer and information sciences (ICCOINS)* (pp. 507-512). IEEE.

Erickson, J., Lyytinen, K., & Siau, K. (2005). Agile modeling, agile software development, and extreme programming: the state of research. *Journal of Database Management (JDM), 16(4),* 88-100.

Hirsjärvi, S., Remes, P. & Sajavaara, P. (2009). Tutki ja kirjoita (15. uud. p.). Helsinki: Tammi.

Inayat, I., Salim, S. S., Marczak, S., Daneva, M., & Shamshirband, S. (2015). A systematic literature review on agile requirements engineering practices and challenges. *Computers in human behavior, 51,* 915-929.

Lee, G., & Xia, W. (2010). Toward agile: An integrated analysis of quantitative and qualitative field data on software development agility. *MIS Quarterly, 34(1).*

Levy, R., Short, M., & Measey, P. (2015, March). Agile foundations: principles, practices and frameworks. *BCS*.

Lukka, Kari. (2014). ”Kari Lukka: Konstruktiivinen tutkimusote”. *METODIX* (blog). 19. toukokuuta 2014. https://metodix.fi/2014/05/19/lukka-konstruktiivinen-tutkimusote.

McCauley, R. (2001). Agile development methods poised to upset status quo. *ACM SIGCSE Bulletin, 33(4),* 14-15.

Misra, S. C., Kumar, V., & Kumar, U. (2009). Identifying some important success factors in adopting agile software development practices. *Journal of Systems and Software, 82(11),* 1869-1890.

Nasir, M. H. N., & Sahibuddin, S. (2011). Critical success factors for software projects: A comparative study. *Scientific research and essays, 6(10),* 2174-2186.

Nerur, S., Mahapatra, R., & Mangalaraj, G. (2005). Challenges of migrating to agile methodologies. *Communications of the ACM, 48(5),* 72-78.

Ochodek, M., & Kopczyńska, S. (2018). Perceived importance of agile requirements engineering practices–A survey. *Journal of Systems and Software, 143,* 29-43.

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

Pulkkinen, M., & Kapraali, L. (2015, July). Collaborative EA Information Elicitation Method: The IEM for Business Architecture. In *2015 IEEE 17th Conference on Business Informatics* (Vol. 2, pp. 64-71). IEEE.

Ramesh, B., Cao, L., & Baskerville, R. (2010). Agile requirements engineering practices and challenges: an empirical study. *Information Systems Journal, 20(5),* 449-480.

Rodríguez, P., Markkula, J., Oivo, M., & Turula, K. (2012, September). Survey on agile and lean usage in finnish software industry. In *Proceedings of the 2012 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement* (pp. 139-148). IEEE.

Rolland, C. (1998, June). A comprehensive view of process engineering. In *International Conference on Advanced Information Systems Engineering* (pp. 1-24). Springer, Berlin, Heidelberg.

Saaranen-Kauppinen, A. & Puusniekka, A. (2008). KvaliMOTV - Menetelmäopetuksen tietovaranto - Yhteiskuntatieteellinen tietoarkisto Haettu 4.9.2019 osoitteesta https://www.fsd.uta.fi/menetelmaopetus/kvali/L6_3_2.html.

Sidky, A., Arthur, J., & Bohner, S. (2007). A disciplined approach to adopting agile practices: the agile adoption framework. *Innovations in systems and software engineering, 3(3),* 203-216.

Tripp, J. F., & Armstrong, D. J. (2014, January). Exploring the relationship between organizational adoption motives and the tailoring of agile methods. In *2014 47th Hawaii International Conference on System Sciences* (pp. 4799-4806). IEEE.

VersionOne. (2018). ”VersionOne 12th Annual State of Agile Report”. 2018. https://explore.versionone.com/state-of-agile/versionone-12th-annual-state-of-agile-report.

Vlaanderen, K., Jansen, S., Brinkkemper, S., & Jaspers, E. (2011). The agile requirements refinery: Applying SCRUM principles to software product management. *Information and software technology, 53(1),* 58-70.

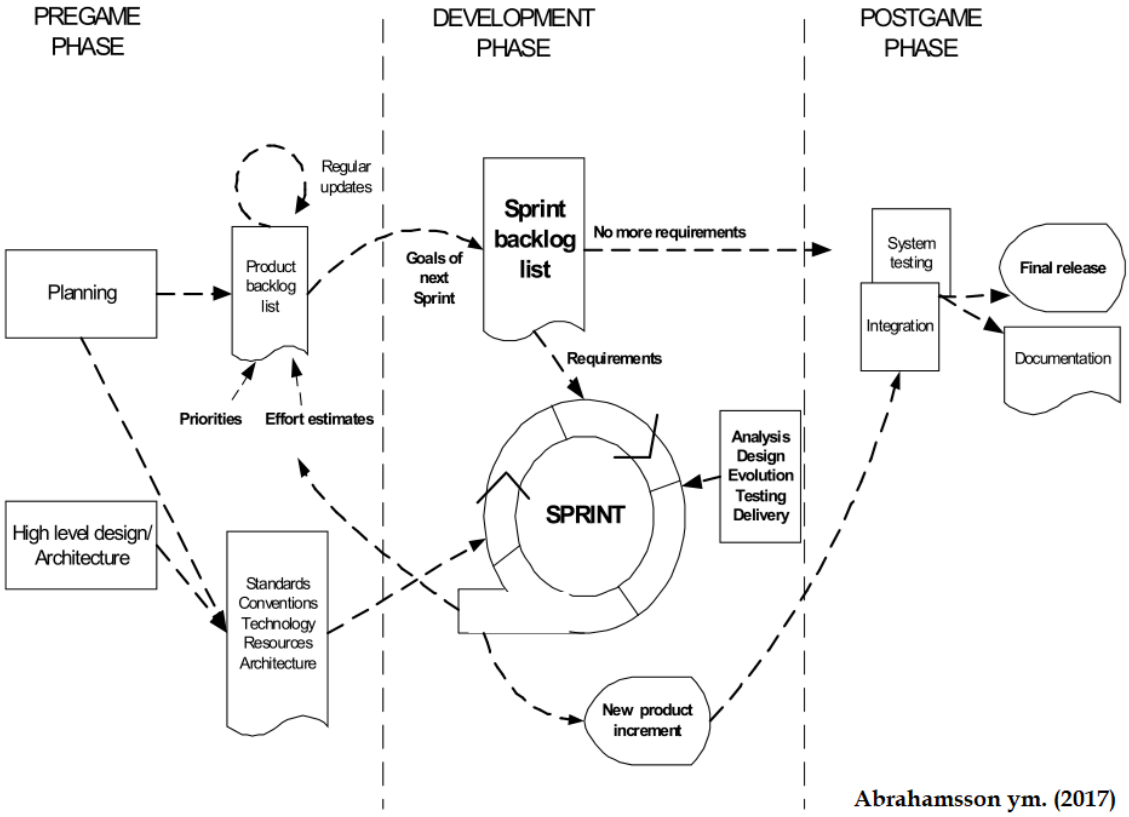
Vähäniitty, J. (2012). *Towards Agile Product and Portfolio Management*. *Aalto University.* https://aaltodoc.aalto.fi:443/handle/123456789/6046.

Wang, X., Conboy, K., & Pikkarainen, M. (2012). Assimilation of agile practices in use. *Information Systems Journal, 22(6),* 435-455.

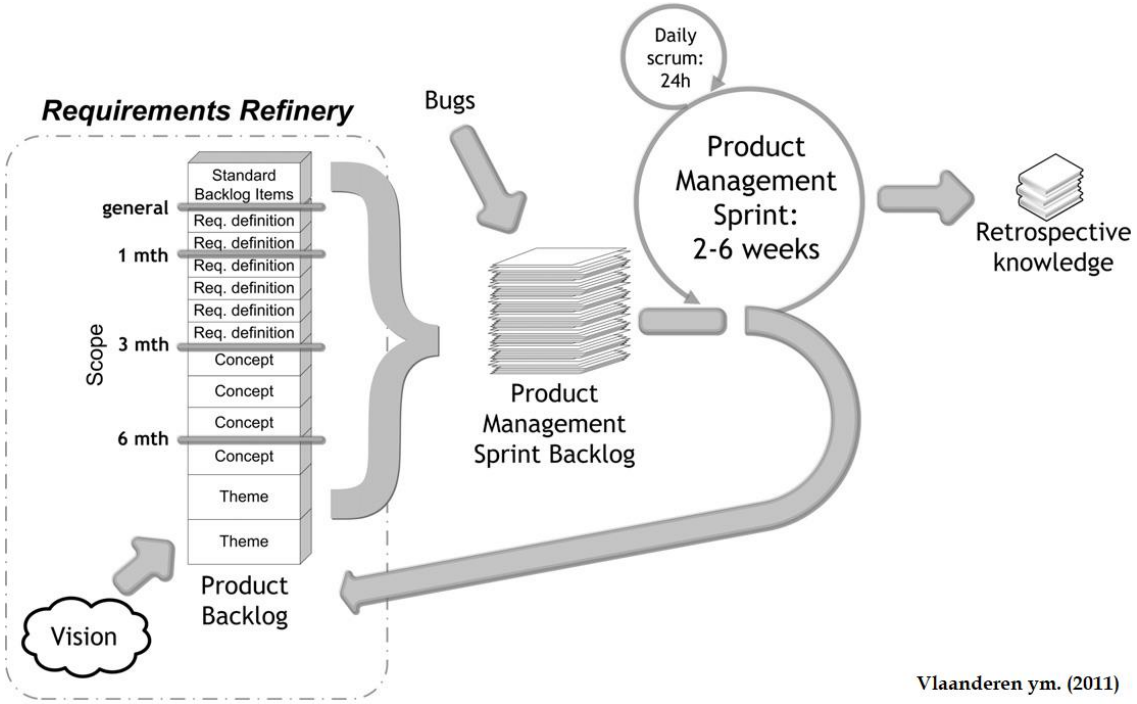
Wiegers, K., & Beatty, J. (2013). Software requirements. *Pearson Education.*

Yang, C., Liang, P., & Avgeriou, P. (2016). A systematic mapping study on the combination of software architecture and agile development. *Journal of Systems and Software, 111,* 157-184.

LIITE 1 PROSESSIKUVAUKSET SCRUM-MENETELMÄSTÄ

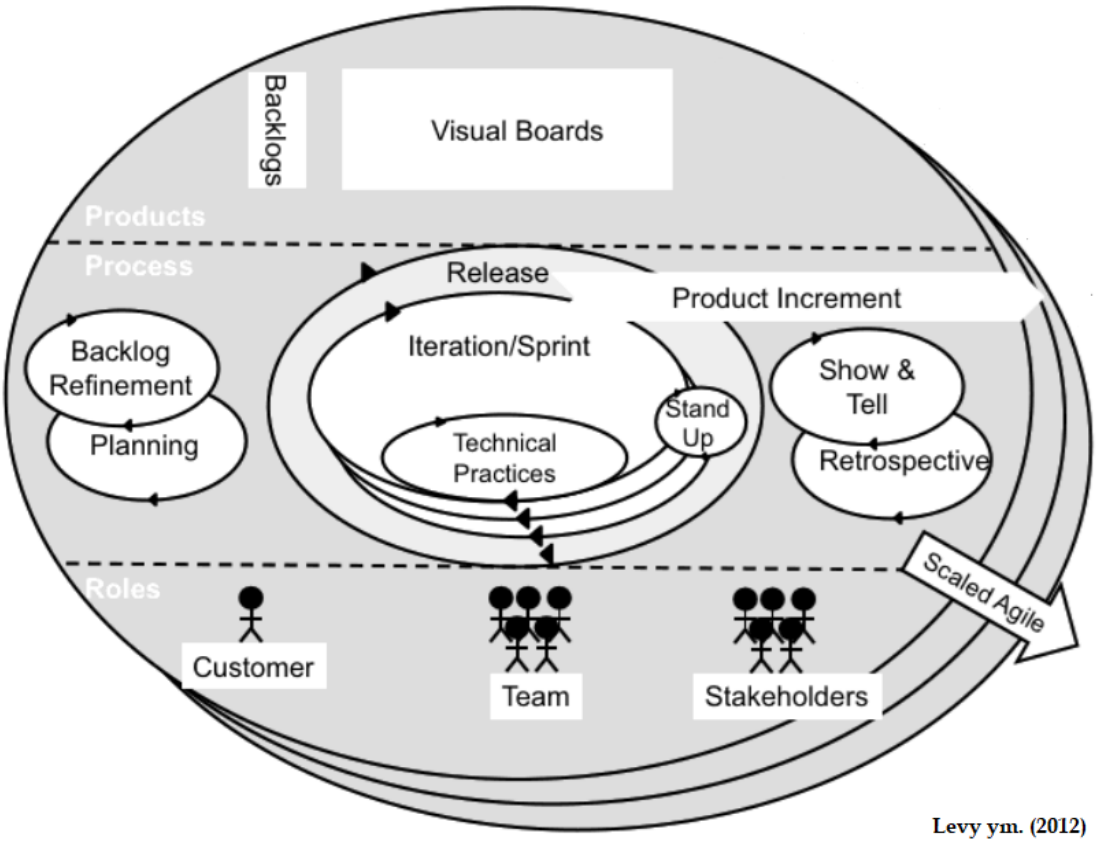


Abrahamsson ym. (2017)

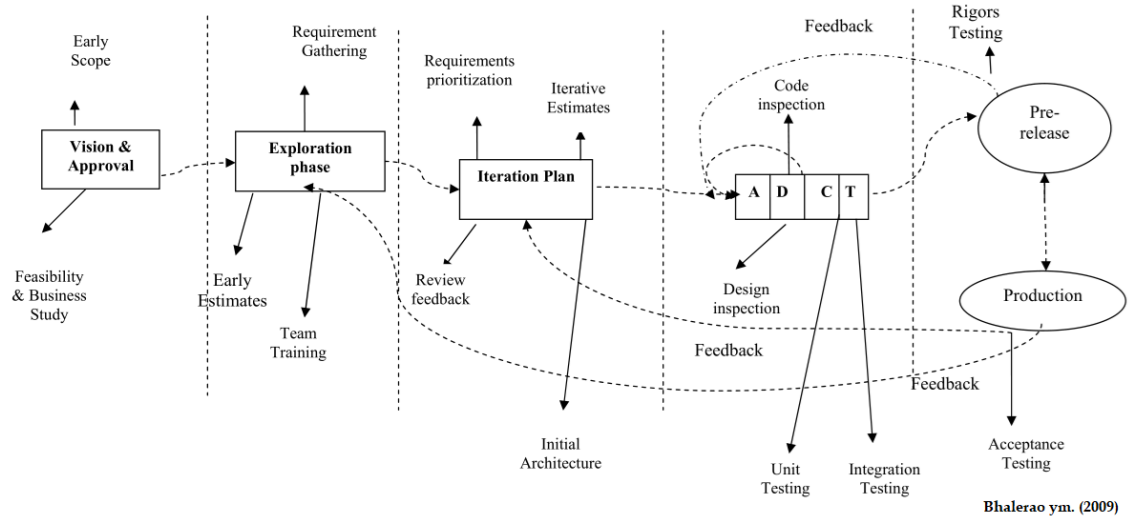


Vlaanderen ym. (2011)

LIITE 2 YLEISTÄVÄT PROSESSIKUVAUKSET KETTERÄSTÄ OHJELMISTOKEHITYSPROJEKTISTA



Levy ym. (2012)



Bhalerao ym. (2009)

LIITE 3 SAATEKIRJE HAASTATTELUUN JA HAASTATTELU-RUNKO

Atte Tuomisto 13.7.2019

HAASTATTELU

SAATEKIRJE

Hei!

Kutsuisin teidät osallistumaan pro gradu –tutkielmaani liittyvään haastatteluun. Tutkielmani tarkoituksena on tuottaa keino, jolla asiakkaalle pystytään osoittamaan olennaisimmat asiat ketterän ohjelmistokehitysprojektin kulusta. Täksi keinoksi tutkielmassani olen luonut asiakkaalle suunnatun ketterän ohjelmistokehityksen prosessikuvauksen. Pitämieni haastattelujen tarkoituksena on saada haastateltavilta palautetta prosessikuvauksestani ja näiden palautteiden perusteella jatkokehittää luomustani. Haastattelu kestää 30-45 minuuttia ja se äänitetään tämän tutkimuksen tarpeita varten. Äänitallenteet hävitetään tutkimuksen valmistuttua.

Tämän koostamani prosessikuvauksen tarkoituksena on pystyä osoittamaan projektiin osallistuvalle asiakkaalle riittävä perustieto ketterän ohjelmistokehitysprojektiin liittyvistä asioista. Asiakkaan perustieto aiheesta on tärkeää, sillä asiakkaan osallistuminen projektiin on hyvin olennaisessa osassa ja suuri tekijä projektin onnistumisen kannalta. Luomani prosessikuvaus pyrkiikin juuri siihen, että sen avulla asiakas pystyy omaksumaan mahdollisimman helposti ketterän ohjelmistokehitysprojektin kulun sekä siihen kuuluvat yleisimmät käytännöt. Tällöin asiakas pystyy jo projektin alussa hahmottamaan, miten projekti yleisesti etenee.

Toivon, että ilmoitatte sähköpostin kautta, mikäli pystytte osallistumaan haastatteluun. Kiitos jo etukäteen!

Ystävällisin terveisin,
Atte Tuomisto
atte.e.e.tuomisto@student.jyu.fi

Atte Tuomisto 13.7.2019

HAASTATTELURUNKO

HAASTATELTAVAN TAUSTATIEDOT

1. Mikä on työnimikkeenne?
2. Lyhyesti, mitä eri työtehtäviä teette tällä hetkellä?
3. Kuinka kauan olette olleet töissä IT-alalla?
4. Oletteko työtehtävissänne tekemisissä yrityksenne asiakkaiden kanssa ja yleensä minkä toimialojen?
5. Osallistutteko tällä hetkellä johonkin kehitysprojektiin, jota toteutetaan ketterin ohjelmistokehitysmenetelmin tai piirtein? Minkä kokoluokan projektista on kyse henkilömäärältään ja mikä on asiakkaan toimiala?

TEEMA 1: ONGELMA

6. Onko teidän mielestänne ohjelmistokehitysprojektien asiakkailla yleensä riittävästi osaamista tai tietoa siitä, miten ohjelmistoja kehitetään ketterin menetelmin?
7. Oletteko joutuneet opastamaan asiakasta ketterästi toteutettavan projektin toteuttamiseen ja etenemiseen liittyen työtehtävissänne?
8. Onko teillä esiintynyt ongelmia sen suhteen, ettei asiakkaalla ole riittävästi tietoa ketterän ohjelmistokehityksen käytänteistä tai piirteistä?
9. Onko asiakasyrityksen kokoluokalla tai toimialalla yleensä merkitystä siinä, kuinka hyvin he entuudestaan tietävät ketterästä ohjelmistokehitysestä?

TEEMA 2: PROSESSIKUVAUS

10. Puuttuuko prosessikuvauksesta asiakkaan kannalta joitain olennaisia tapahtumia? Voisiko niitä parantaa tai täsmentää?
11. Puuttuuko prosessikuvauksesta asiakkaan kannalta joitain olennaisia tuotoksia? Voisiko niitä parantaa tai täsmentää?
12. Puuttuuko prosessikuvauksesta asiakkaan kannalta joitain olennaista? Voisiko niitä parantaa tai täsmentää?

TEEMA 3: SELITETEKSTIT

13. Koetko selitetekstit hyödyllisenä ja tarpeellisena osana prosessikuvausta?
14. Koetko ylimääräiset käsitteet tarpeellisiksi, onko niistä jotain huomioita?
15. Onko seliteteksteistä parannettavaa tai täsmennettävää?

TEEMA 4: YLEINEN ARVIO

16. Kuinka hyvin tuotos osoittaa tyypillisen ketterän ohjelmistokehitysprojektin kulkua ja vastaako se reaalimaailmaa?
17. Kuinka hyvin tuotos onnistuu kuvaamaan ketterää ohjelmistokehitystä asiakkaalle suunnatusta näkökulmasta?
18. Onko prosessikuvaus asiakkaalle riittävän yksinkertainen ja helposti ymmärrettävissä?
19. Voisiko tätä tuotosta tai tämän tuotoksen pohjalta tehtyä tuotosta käyttää hyödyksi oikeissa projekteissa, joissa asiakas tulisi saada perehdytettyä ketterään ohjelmistokehitykseen?

LIITE 4 PROSESSIKUVAUKSEN SELITETEKSTIT

Käsite	Määritelmä
Asiakkaalle suunnattu ketterän ohjelmistokehitysprojektin prosessikuvaus	Tämän prosessikuvauksen tarkoituksena on kuvata yleistetty ketterän ohjelmistokehitysprojektin kulku, ja siihen liittyvät tärkeimmät tapahtumat ja termit asiakkaalle suunnatusta näkökulmasta. Prosessikuvaus ja siihen liittyvät selitetekstit ovat laadittu siten, että niiden avulla asiakas pystyy helposti ja nopeasti käsittämään ketterään ohjelmistokehitykseen liittyvät käsitteet ja tärkeimmät muut asiat.
Ketterä ohjelmistokehitys (eng. agile software development)	Ketterä ohjelmistokehitys on ajatusmalli, jonka pohjalta toteutetaan ohjelmistokehitysprojekteja. Se pohjautuu iteratiivislähtöiseen tapaan kehittää ohjelmistoja osa kerrallaan. Vaatimukset ja niiden ratkaisut kehittyvät ja muuttuvat jatkuvasti yhdessä asiakkaan ja kehitystiimin kanssa ja se korostaa tiivistä yhteistyötä ja jatkuvaa pienten osien tuotantoon vientiä.
Jatkuva suunnittelu	Ketterän ohjelmistokehityksen luonteeseen kuuluu, että projektin asioita suunnitellaan jatkuvasti. Tähän suunnitteluun kuuluu niin projektin lyhyen kuin pitkän aikavälin tarpeet. Ketterässä ohjelmistokehityksessä vallitsee käsitys, ettei projektin alkuvaiheessa voida ennustaa kaikkia projektiin liittyviä tarpeita, vaan ne tarkentuvat koko projektin ajan. Näin ollen projektin tuotteeseen, toimintatapoihin ja muihin seikkoihin tehdään tarkennuksia jatkuvasti ja tarkoituksena on hakea projektissa mahdollisimman optimaalista tilannetta niin projektin työtavoissa kuin myös projektin lopputuloksessa.
Kasvokkainen vuorovaikutus	Ketterässä ohjelmistokehityksessä kommunikointi on suuressa arvossa. Varsinkin kasvokkain käytyä kommunikointia pidetään kaikkein arvokkaimpana kommunikoinnin muotona. Tästä syystä sitä pyritäänkin käymään mahdollisimman paljon.
Kehittyvät ja muuttuvat vaatimukset	Ketterässä ohjelmistokehityksessä projektin aikana tapahtuvia muutoksia ajatellaan täysin luonnollisina ja hyväksyttävinä. Projektin kokonaisuus ja siihen kuuluvat vaatimukset tarkentuvat koko projektin ajan. Vaatimuksia kehitetään, muutetaan, poistetaan ja lisätään jatkuvasti. Nämä muutokset vaikuttavat projektiin ja niiden mukana tehdään korjaavia toimenpiteitä esimerkiksi kustannusarvoihin.
Kestävä työtahti	Ketterässä ohjelmistokehityksessä pyritään sellaiseen työtahtiin, jossa jokaisella projektiin osallistuvalla on kohtuullinen määrä töitä. Projektin työkuorma suunnitellaan aina siten, että työkuorma on silloisiin resursseihin sopiva. Tarkoituksena projektissa on löytää kehitystahti, jota pystytään toteuttamaan ilman, että kukaan projektiin osallistuvista henkilöistä kuormittuu liikaa. Tämä asia on myös asiakkaan vastuulla ja on ymmärrettävä, että esimerkiksi liian tiukat aikataulut saattavat kuormittaa liikaa kehitystiimiä.
Lyhyet iteraatiot ja julkaisut	Projektissa pyritään pitämään iteraatiot mahdollisimman lyhyinä, jotta muutoksiin pystytään reagoimaan nopeasti. Samoin tarkoituksena on tuottaa mahdollisimman usein julkaistavia toiminnallisuuksia, jotta asiakkaalle voidaan tuottaa järjestelmästä toimivia osakokonaisuuksia jo projektin varhaisessa vaiheessa.
Osapuolten tiivis osallistuminen	Ketterään ohjelmistokehitykseen kuuluu hyvin olennaisesti kaikkien osapuolten osallistuminen projektiin. Etenkin asiakas on jatkuvasti kehitystiimin kanssa vuorovaikutuksessa. Tämän lisäksi esimerkiksi kehitettävän tuotteen loppukäyttäjien mielipiteet ovat tärkeitä ottaa huomioon. Tiiviillä osallistumisella tuotteesta saadaan mahdollisimman hyvä lopputulos, joka vastaa niin asiakkaan kuin loppukäyttäjien tarpeita.

Käsite	Määritelmä
Esiselvitys	Projektin alussa määritellään hyvin karkeasti projekti tarpeita ja kehitettävän tuotteen osakokonaisuuksia. Tarkoituksena on luoda yleinen kuva siitä, mitä projektissa tehdään ja toteuttaa alustavia arvioita esimerkiksi projektin budjetista ja kestosta sekä mahdollisesti tutustuttaa osallistuvia henkilöitä projektiin.
Hyväksyttäminen (eng. acceptance)	Asiakkaalla on lopullinen vastuu kehitettävien toiminnallisuuksien hyväksynnässä. Asiakas hyväksyy iteraation aikana kehitetyt tehtävät, eli esimerkiksi jonkin toiminnallisuuden järjestelmässä, jotta se voidaan todeta valmiiksi. Ilman asiakkaan hyväksyntää, ei kehitettyä toiminnallisuutta julkaista. Hylättyyn toiminnallisuuteen tehdään asiakkaan vaatimat muutokset, jotta se täyttää sille annetut vaatimukset ja siten voidaan hyväksyä.
Iteraatio (eng. iteration)	Valmiiksi määritetyn pituinen ajanjakso, jonka aikana kehittäjätiimin tarkoituksena on työstää ajanjaksolle valittuja tehtäviä tiettyjen tuotteen toiminnallisuuksien edistämiseksi. Ajanjakson aikana on tarkoitus saada iteraatiolle valitut tehtävät valmiiksi ja mahdollisesti julkaisukelpoisiksi. Asiakkaan velvollisuutena on olla iteraation aikana käytettävissä, sillä kehittäjät voivat tarvita asiakkaan mielipiteitä tai täsmennyksiä iteraation tehtäviin.
Iteraation suunnittelu (eng. iteration planning)	Iteraatiota edeltävä tapahtuma, jossa määritellään mitä kehitettäviä tehtäviä tuotteen kehitysjonosta toteutetaan seuraavassa iteraatiossa. Nämä toiminnallisuuksista ja muista vaatimuksista pilkotut työtehtävät siirretään iteraation kehitysjonolle ja kullekin työtehtävälle määritetään tekijä kehitystiimistä. Iteraatiolle valitaan vain se määrä tehtäviä, jotka on mahdollista toteuttaa iteraation aikana. Projektin kehitystiimi käy läpi yhdessä asiakkaan kanssa mitä seuraavaksi halutaan kehittää, eli miten työtä priorisoidaan. Iteraatioita voidaan suunnitella myös muulloinkin kuin iteraation alussa.
Julkaisu (eng. release)	Julkaisun suunnittelussa päätetyn kokonaisuuden mukaisesti tuotteesta toteutetaan uusi versio, joka sisältää silloiseen versioon tarkoitetut valmiit sekä hyväksytyt toiminnallisuudet. Julkaisu voidaan toteuttaa iteraation loputtua tai muuna ajanjaksona ja varsinaisen julkaisun toteuttaa kehitystiimi.
Julkaisun suunnittelu (eng. release planning)	Kun kehitetyt toiminnallisuudet valmistuvat ja ne ovat hyväksytetty, voidaan suunnitella, milloin ne aiotaan julkaista. Suunnittelua voi toteuttaa kunkin iteraation loppuvaiheilla tai muuna ajanjaksona. Toiminnallisuuksien julkaisuja suunnitellaan sekä priorisoidaan koko ajan ja asiakas yhdessä kehittäjätiimin kanssa pyrkii muodostamaan järkeviä kokonaisuuksia uusista toiminnallisuuksista, jotka julkaistaan seuraavaksi. Mahdollisia julkaisuja voidaan myös suunnitella pidemmällä aikavälillä ja niiden tarkemmat ajankohdat ja sisältö tarkentuvat projektin aikana.
Katselmointi (eng. review)	Iteraation loppuvaiheen tapahtuma, jossa kehitystiimi esittää asiakkaalle mitä iteraation aikana tehtävistä saatiin tehtyä ja mitkä tehtävät jäivät tekemättä. Näissä tapahtumissa asiakkaan on mahdollista nähdä ja mahdollisesti kokeilla uusia kehitettyjä toiminnallisuuksia ja monesti hyvin varhain projektin alussa projektissa pyritään siihen, että asiakkaalle olisi mahdollisimman nopeasti jotain konkreettista näytettävää kuten esimerkiksi prototyyppiä. Näillä katselmoinneilla varmistetaan, että iteraation aikana kehitetyt lopputulokset vastaavat asiakkaan visiota tuotteesta.
Kehitysjono (eng. backlog) (tuotteen kehitysjono) (iteraation kehitysjono)	Kehitysjonot ovat jäsenneltyjä listoja siitä, mitä asioita kehitettävään tuotteeseen tulee toteuttaa. Kehitysjonojen tarkoituksena on jatkuvasti ylläpitää tietoa siitä, mitä projektissa on jo tehtynä ja mitä on vielä tekemättä. Samoin kehitysjono on ensisijainen paikka kaikelle dokumentoinnille, joita kuhunkin osakokonaisuuteen kuuluu. Kehitysjonojen avulla myös asiakkaalle pysyy läpinäkyvänä missä vaiheessa projektin työt ovat. Samoin asiakas pystyy tarkentamaan kehitysjonossa olevien asioiden vaatimuksia tai selittämään tarkemmin mitä jossain toiminnallisuudessa tarkoitetaan. Kehitysjonoja on erilaisia, joista yleisiä ovat tuotteen kehitysjono ja iteraation kehitysjono.

	Tuotteen kehitysjono on lista kaikista kehitettävistä asioista ja vaatimuksista tuotteelle. Iteraation kehitysjono sisältää tuotteen kehitysjonosta valitut tehtävät, joita on tarkoitus toteuttaa valitun iteraation aikana.
Uudet, keskeneräiset ja hyväksymättömät tehtävät (eng. unfinished and unaccepted tasks)	Iteraation loputtua iteraatiossa työstetyt tehtävät valmistuvat tai jäätävät kesken. Tällöin keskeneräiset tehtävät palautetaan takaisin tuotteen kehitysjonolle odottamaan seuraavaa iteraation suunnittelua. Toisinaan tehtävän valmistuttua asiakas ei välttämättä hyväksy kehitetyn tehtävän lopputulosta. Kehitetty asia on esimerkiksi ymmärretty väärin ja se ei vastaa sitä mitä asiakas on sen halunnut olevan. Tällöin tehtävät palautetaan tuotteen kehitysjonolle odottamaan seuraavaa iteraation suunnittelua.
Osakokonaisuus (eng. epic) Toiminnallisuus (eng. feature) Tehtävä (eng. task)	Projektissa toteutettavat asiat on jaoteltu eri tasoisiin kokonaisuuksiin. Asiakkaan silloisesta visiosta on suunnittelemalla pilkottu tuotteeseen tarvittavat asiat osakokonaisuuksiin. Nämä osakokonaisuudet yhdessä muodostavat projektissa kehitettävän tuotteen. Osakokonaisuudet koostuvat yleensä useammista toiminnallisuuksista ja toiminnallisuudet vuorostaan koostuvat monista tehtävistä. Nämä eri tasoiset vaatimukset tarkentuvat, muovaantuvat koko projektin ajan.
Priorisointi (eng. priorization)	Projektin osakokonaisuuksia ja niiden sisältämiä tehtäviä priorisoidaan projektissa jatkuvasti. Asiakas on vahvasti mukana priorisoinnissa, sillä yleensä asiakkaalla on parhain käsitys siitä, mitkä asiat ovat tuotteessa kaikkein tärkeimmässä asemassa. Kehitystiimi on myös vahvasti mukana priorisoinnissa. Priorisointiin vaikuttaa moni asia kuten esimerkiksi kuinka työläs kehitettävä toiminnallisuus on toteuttaa tai kuinka tärkeää kyseinen toiminnallisuus on saada mahdollisimman pian valmiiksi.
Päivittäispalaveri (eng. daily meeting)	Kehittäjätiimin iteraation aikainen päivittäinen lyhyt tapaaminen, jonka tarkoituksena on saada kaikki tiimin osallistujat ajan tasalle projektin silloisesta kulusta ja siitä mitä kullakin tiimin jäsenellä on työn alla.
Retrospektiivi (eng. retrospective)	Iteraation loppuvaiheen tapahtuma, jossa käsitellään iteraation onnistumisia sekä epäonnistumisia ja pyritään kehittämään toimintaa niiden mukaan. Samoin pyritään pohtimaan, onko iteraatiolla ollut esimerkiksi liian suuri tai pieni työkuorma ja onko työtahti kestävä. Yleensä kehittäjätiimin keskeinen tapahtuma, mutta asiakas voi olla mahdollisesti myös mukana tapahtumassa.
Tuote (eng. product)	Projektissa tuotettava asiakkaan tilaama lopputulos. Yleisesti se voi olla mikä tahansa projektissa tuotettava asia esimerkiksi tietojärjestelmä tai palvelu.
Viimeisin versio tuotteesta (eng. latest version of the product)	Ketterässä ohjelmistokehityksessä projektissa kehitettävää tuotetta kehitetään iteratiivisesti pala kerrallaan. Tämä tarkoittaa sitä, että iteraation tavoitteena on usein, että sen loputtua pystyttäisiin julkaisemaan jotain toimivaa tuotosta kehitettävästä tuotteesta. Tällä tavoin asiakas saa mahdollisimman nopeasti käyttöönsä jotain konkreettista ja näin ollen tuotetta voidaan alkaa osittain käyttää.
Visio tuotteesta (eng. product vision)	Tuotteen visio on jatkuvasti kehittyvä käsitys projektissa kehitettävästä tuotteesta ja sen ihanteellisesta lopputuloksesta. Visio on pääosin projektin asiakkaalla sekä muilla liittyvillä sidosryhmillä. Visio käsittää käsityksen siitä mitä arvoa kehitettävä tuote tuo, kenelle se on osoitettu ja mitä liiketoimintamahdollisuuksia sillä on. Visio muuttuu projektin aikana paljon. Joitain ideoita jätetään toteuttamatta tai joitain asioita lisätään, vaikkei alkuperäinen ajatus tuotteesta sisältänytkään sitä. Alkuperäinen visio lopullisesta tuotteesta ei siis välttämättä vastaa varsinaista lopullista tuotetta.