Emilia Rantonen

Ketterät menetelmät ja työmäärän arviointi ohjelmistoprojekteissa

Tieto- ja ohjelmistotekniikan kandidaatintutkielma

17. toukokuuta 2024

Jyväskylän yliopisto

Informaatioteknologian tiedekunta

Tekijä: Emilia Rantonen

Yhteystiedot: emilia.s.rantonen@student.jyu.fi

Ohjaaja: Annemari Auvinen

Työn nimi: Ketterät menetelmät ja työmäärän arviointi ohjelmistoprojekteissa

Title in English: Agile methods and effort estimation in software projects

Työ: Kandidaatintutkielma

Sivumäärä: 25+0

Tiivistelmä: Ketterät menetelmät, kuten Scrum, ovat yleistyneet ohjelmistoprojekteissa ja niitä käytetään sekä ohjelmistokehityksessä että projektinhallinnassa. Tutkielmassa käsitellään näiden menetelmien periaatteita, keskittyen erityisesti Scrumiin. Lisäksi tarkastellaan työmäärän arvioinnin merkitystä ja yleisimpiä haasteita. Tutkimuksen tavoitteena on valottaa ketterien menetelmien suosiota ja selvittää, miksi työmäärän arviointi on keskeistä onnistuneelle projektinhallinnalle. Ketterien menetelmien etuihin kuuluvat kyky reagoida muutoksiin ja jatkuva yhteistyö asiakkaan kanssa, kun taas työmäärän arvioinnin haasteina ovat usein puutteelliset ja virheelliset arviot.

Avainsanat: Ketterät menetelmät, projektinhallinta, työmäärän arviointi, Scrum, Planning Poker

Abstract: Agile methods, such as Scrum, are widely used in software projects and they are used in for both development and project management purposes. This thesis introduces the principles of these methods, with a particular focus on Scrum. In addition this thesis discusses the importance and common challenges of effort estimation. The purpose of the study is to explore the reasons behind the popularity of these methods and to understand why effort estimation is critical for successful project management. The advantages of agile methods are the ability to respond to changes and continuous collaboration with the customer, while the primary challenges of effort estimation often involves incomplete and inaccurate estimates.

Keywords: Agile Methods, Project Management, Effort Estimation, Scrum, Planning Poker

Sisällys

1 Johdanto

Ohjelmistoprojektien innovaatioprosessien haasteet ovat ohjanneet ketterien menetelmien kehitystä sekä ohjelmistokehityksessä että projektinhallinnassa (Zayat ja Senvar 2020). Poženelin ym. (2024) mukaan kehitykseen on vaikuttanut myös se, että ohjelmistoprojektien epäonnistumisaste on pysynyt korkeana viime vuosikymmenen ajan. Yksi keskeisimmistä syistä projektien epäonnistumiselle on suunnittelemattomat muutokset, jotka ilmenevät projektin aikana. Myös Ionelin (2019) mukaan taloudellisen ja teknisen ympäristön nopeiden muutosten seurauksena organisaatioilla on yhä enemmän painetta kehittää joustavuutta ja tehokkuutta. Tämän seurauksena perinteiset menetelmät alkavat väistyä uusien ketterämpien menetelmien tieltä. Alsaqqa, Sawalha ja AbdelNabi (2020) määrittelevät ketterät menetelmät ohjelmistokehityksen viitekehyksinä, jotka toimivat kattokäsitteenä ohjelmistotuotannolle. Ketterä kehitys on herättänyt laajaa kiinnostusta ja aiheuttanut ajattelutavan muutoksen ohjelmistotekniikassa vakiinnuttaen ketterän lähestymistavan uudeksi standardiksi ohjelmistoprojektien hallinnassa (Poženel ym. 2024).

Ketterien menetelmien kehitystä on ohjannut myös se, että Alsaqqan, Sawalhan ja AbdelNabin (2020) mukaan perinteisiä menetelmiä käyttävissä ohjelmistoprojekteissa on kohdattu haasteita esimerkiksi ylläpidossa sekä muuttuvissa vaatimuksissa. Poženelin ym. (2024) mukaan perinteiset suunnittelumallit, joissa laaditaan kattava suunnitelma alussa ja jatketaan toteutusvaiheeseen, eivät välttämättä ole optimaalisia useimmille ohjelmistoprojekteille. Syy tähän on se, että ne rajoittavat projektitiimin kykyä käsitellä vastaan tulevia odottamattomia haasteita. Tästä syystä monet ohjelmistoyritykset ovat siirtyneet käyttämään ketterää ohjelmistokehitystä.

Tämän tutkimuksen tarkoituksena on syventyä Scrumiin ja sen hyötyihin sekä merkitykseen niin ohjelmistokehityksessä kuin projektinhallinnassa. Lisäksi tutkitaan millainen rooli ja soveltuvuus Planning Pokerilla on työmäärän arvioinnissa ja mitä haasteita tähän liittyy. Scrum on yleisimmin käytetty ketterä menetelmä tuotekehityksessä ja ohjelmistojärjestelmien kehityksessä (Chaudhari ja Joshi 2021). Alsaqqan, Sawalhan ja AbdelNabin (2020) mukaan se on lisäksi vaikutusvaltaisessa roolissa ohjaamassa ketteriä kehitystrendejä. He kertovat Scrumin suosion perustuvan sen yksinkertaisuuteen, keskittyen ohjelmistohallinnan kysymyksiin

teknisten ohjelmistokehityskäytänteiden sijaan. Tämän ansioista sitä voidaan soveltaa mille tahansa toimialalle. Planning Poker on yksi suosituimmista työmäärän arviointimenetelmistä ketterien menetelmien yhteydessä (Poženel ym. 2024). Rajauksella pyritään keskittymään syvällisemmin näiden kahden menetelmän yksityiskohtiin sekä niiden käytön arviointiin ohjelmistoprojekteissa.

Tässä tutkielmassa määritellään aluksi ketterät menetelmät ja esitellään viitekehys Scrum, sekä perehdytään työmäärän arvioinnissa käytettävään Planning Pokeriin. Tämän jälkeen kolmannessa luvussa tarkastellaan projektinhallintaa ketterissä ohjelmistoprojekteissa sekä Scrumin roolia siinä. Neljännessä luvussa keskitytään työmäärän arvioinnin merkitykseen ketterässä ohjelmistokehityksessä, Planning Pokerin käyttöön arviointiprosessissa sekä siihen, miten se soveltuu tähän tarkoitukseen. Neljännessä osassa tarkastellaan lisäksi työmäärän arvioinnin yleisimpiä haasteita.

2 Ketterät viitekehykset ja menetelmät

Ohjelmistokehityksessä ketterät menetelmät perustuvat jatkuvaan synkronointiin nykyisen ohjelmiston kehitysvaiheen kanssa sekä jatkuvaan iterointiin ja testaukseen läpi koko projektin elinkaaren (Srivastava, Bhardwaj ja Saraswat 2017). Ketterien projektien kehityksessä toistetaan iteratiivisia ja inkrementaalisia vaiheita, kunnes päästään ohjelmiston käyttöönottoon saakka (Ciric Lalic ym. 2022). Nämä vaiheet tapahtuvat jatkuvan vuorovaikutuksen avulla koko projektin elinkaaren ajan.

Zayatin ja Senvarin (2020) mukaan ohjelmistokehitystä on ohjannut perinteiset mallit, kuten vesiputousmalli. Perinteisissä malleissa tehtävät ja vastuut on jaettu selkeästi eri rooleille kuten projektipäällikölle tai kehittäjälle. Asiakkaan osallistuminen rajoittuu projektin ensimmäiseen vaiheeseen eli tekniseen vaatimusmäärittelyyn. Perinteisten menetelmien haasteena on suuri määrä kirjallista työtä, sillä kaikki vaiheet tulee virallistaa dokumentoinnin avulla. Dokumentointi on avainasemassa perinteisiä menetelmiä käytettäessä ja sen haastava käytettävyys on yksi näiden menetelmien rajoituksista (Alsaqqa, Sawalha ja AbdelNabi 2020). Merkittävin ero ketterien ja perinteisten menetelmien välillä on muutoksiin sopeutuminen sekä jatkuva yhteys asiakkaan kanssa (Zayat ja Senvar 2020). Ketterien menetelmien automaation puute on haaste, ja se on yleisin syy miksi esimerkiksi vesiputousmalli valitaan niiden sijaan (Srivastava, Bhardwaj ja Saraswat 2017).

Ketterät menetelmät mahdollistavat sopeutumisen ohjelmistoprojektien muuttuviin vaatimuksiin ja niiden avulla voidaan tuottaa laadukkaampaa ohjelmistoa asiakastarpeiden mukaisesti (Chaudhari ja Joshi 2021). Myös Ciric Lalicin ym. (2022) mukaan ketterien menetelmien käytöllä on merkittävä vaikutus asiakkaan tyytyväisyyteen. Alsaadin ja Saeedin (2022) mukaan ketterien menetelmien suosio perustuu niiden tarjoamiin etuihin. Menetelmien joustavuus monimutkaisia projekteja kehittäessä auttaa organisaatioita kohti nopeampaa toimitusta pienemmillä kustannuksilla. Myös Alsaqqan, Sawalhan ja AbdelNabin (2020) mukaan tärkeimmät saavutetut edut ovat parempi kommunikaatio ja yhteistyö tiimin kesken. Lisäksi etuna ovat nopeat julkaisut, suunnittelun joustavuus sekä kokonaisvaltaisesti järkevämpi prosessi. Ketterien menetelmien periaatteita voidaan soveltaa myös projektinhallintaan (Hayat ym. 2019). Seuraavaksi esitellään tarkemmin ketterien menetelmien periaatteet ja arvot.

2.1 Ketterien menetelmien periaatteet ja arvot

Ketterät viitekehykset ovat saaneet alkunsa vuonna 2001, kun ohjelmistotekniikan ammattilaiset kokoontuivat ja tuloksena oli ketterä manifesti (Alsaqqa, Sawalha ja AbdelNabi 2020). Ciric Lalicin ym. (2022) mukaan kokoontumisen tarkoituksena oli päästä yhteisymmärrykseen siitä, miten ohjelmistokehitysteollisuudessa voitaisiin tuottaa parempia tuloksia ja ylittää perinteisen ohjelmistokehityksen rajoitukset. Heidän mukaansa tänä päivänä yleisesti vallitsee ajatus, että ketterät menetelmät ovat saaneet alkunsa manifestista. Alsaqqa, Sawalha ja AbdelNabi (2020) määrittävät, että manifestin ja ketterien menetelmien kehityksen alkuperäinen tavoite oli vähentää ohjelmistokehitysprosessin ylimääräistä työtä ja mahdollistaa muutosten hyväksyminen läpi projektin. Menetelmissä pyritään siihen, että muutokset otetaan huomioon ilman, että projekti vaarantuu tai aiheutuu suuri määrä saman työn tekemistä uudelleen. Heidän mukaansa tämä filosofia kattaa alleen neljä arvoa ja kaksitoista periaatetta, jotka tukevat ja muodostavat olennaisen osan ketteryyttä tarjoten samalla perustan ohjelmistokehitysprosessin ohjaamiseen.

Manifestissa määritellyt kaksitoista periaatetta ovat jatkuva toimitus, vaatimusten muutosten hyväksyminen, jatkuva sidosryhmien välinen yhteistyö, yksilöiden motivaatio työskentelyyn, kasvokkaiset keskustelut ja tapaamiset, toimiva ohjelmisto, kestävät kehitystiimit, jatkuva suunnittelu, työn minimoimisen taito, onnistunut arkkitehtuuri sekä tehokkaammat tiimit (Mallidi ja Sharma 2021). Alsaqqan, Sawalhan ja AbdelNabin (2020) mukaan tärkein periaate on jatkuva toimitus, sillä varhainen ja jatkuva ohjelmiston toimittaminen luo luottamuksen asiakkaan ja kehitystiimin välille. Heidän mukaansa muutosten hyväksymisen tarkoituksena on toimia ketterien menetelmien kilpailuetuna. Jatkuvalla sidosryhmien välisellä yhteistyöllä voidaan antaa palautetta puolin ja toisin sekä vastata kehitystiimin kysymyksiin. Alsaqqan, Sawalhan ja AbdelNabin (2020) tutkimuksen mukaan motivoituneet tiimin jäsenet ovat keskeisin menestystekijä, ja ketterät menetelmät luottavat näiden yksilöiden taitoon tehdä asianmukaiset päätökset työnsä suorittamiseksi. He määrittävät kasvokkain tapahtuvan viestinnän olevan tehokkain tapa tiedon välittämiseen tiimin sisällä. Tämä periaate korostaa suoraa ihmiskommunikaatiota ketterässä tiimissä kirjallisten erittelyiden tai suunnitelmien sijasta. Toimiva ohjelmisto on edistyksen ensisijainen mittari asiakkaan näkökulmasta (Alsaqqa, Sawalha ja AbdelNabi 2020). Toimivan ohjelmiston tuottamisen yhteydessä kestävä

kehitys tarkoittaa, että tiimin tulisi ylläpitää jatkuvaa tahtia ja ohjelmiston laatua. Kestävän kehityksen johdosta jatkuva huomio tekniseen laatuun ja hyvään suunnitteluun tehostaa projektin ketteryyttä. Alsaqqan, Sawalhan ja AbdelNabin (2020) mukaan muiden periaatteiden lisäksi projektin yksinkertaisuus on tärkeää, ja sen avulla pyritään tekemään mahdollisimman vähän tarpeetonta työtä. Itseohjautuva tiimi on avain onnistuneeseen arkkitehtuuriin ja yksinkertaisuuden toteuttamiseen. Tiimit suunnittelevat itse, miten käsitellä annettuja vastuita ja rakentavat optimaalisen rakenteen kehitysprosessin aikana. Viimeisenä periaatteena tiimi säännöllisin väliajoin pohtii, miten muuttaa toimintaansa tehokkaammaksi. Perustuen tähän, tiimi mukauttaa ja muuttaa käyttäytymistään tarpeen mukaan (Alsaqqa, Sawalha ja AbdelNabi 2020).

Periaatteiden lisäksi Alsaqqan, Sawalhan ja AbdelNabin (2020) mukaan manifestissa määriteltiin neljä arvoa. Heidän mukaansa ensimmäinen arvo on se, että muodollisten prosessien ja teknisten ympäristöjen merkityksen korostamisen sijaan tulisi keskittyä kommunikaatioon, vuorovaikutukseen sekä osaaviin tiimin jäseniin. He määrittelevät manifestin toisen arvon korostavan toimivan ohjelmiston tärkeyttä kattavan dokumentaation sijaan. Toisen arvon mukaan perinteisissä menetelmissä dokumentaatioon käytetyt resurssit tulisi optimoida esimerkiksi ohjelmiston testaukseen sillä sen avulla saadaan suoraan tietää täyttääkö ohjelmisto vaatimukset vai ei. Kolmas arvo tuo esiin asiakasyhteistyön tärkeyden, jotta muuttuviin vaatimuksiin voidaan vastata missä tahansa projektin vaiheessa. Todellisten asiakastarpeiden täyttämiseksi tulisi muodollisten sopimusten sijaan keskittyä asiakaspalautteeseen, neuvotteluihin ja yhteistyöhön kehitystiimin kanssa. Neljäs arvo perustuu muutoksiin vastaamiseen suunnitelman seuraamisen sijaan asiakastyytyväisyyden saavuttamiseksi. Manifestissa etusija annetaan muutoksiin reagoimiselle kehitysprosessin elinkaaren aikana tiukasti määritellyn suunnitelman noudattamisen sijaan (Alsaqqa, Sawalha ja AbdelNabi 2020). Nämä neljä arvoa luovat perustan ketterälle ohjelmistokehitykselle ja ovat ohjanneet ketterien menetelmien soveltamista käytännön projekteihin. Seuraavassa luvussa käsitellään Scrumia, joka on yksi suosituimmista ketteristä menetelmistä, joka soveltaa näitä arvoja ja periaatteita.

2.2 Scrum

Scrum on kevyt ja ketterä viitekehys, joka mahdollistaa ohjelmiston ja tuotteen kehitysprosessin ohjauksen ja hallinnan (Srivastava, Bhardwaj ja Saraswat 2017). Sen tavoitteena on mm. parantaa kehitysprosessin tuottavuutta sekä yhteensovittaa yksilöiden ja organisaatioiden tavoitteita. Lisäksi tavoitteena on parantaa projektien ja ohjelmistokehityksen vaiheiden tasoa (Hayat ym. 2019). Scrum mahdollistaa yrityksille nopeampaa etenemistä, muutosten vastaanottamista sekä sopeutumista muuttuviin liiketoiminnan tarpeisiin (Madya, Budiardjo ja Mahatma 2022). Alsaqqan, Sawalhan ja AbdelNabin (2020) tutkimuksessa kerrotaan, että Scrum on tiimipohjainen ketterä menetelmä, joka käyttää lyhyitä aikarajoitettuja iteraatioita, joita kutsutaan sprinteiksi. Heidän mukaansa Scrumin suosio perustuu yksinkertaisuuteen sekä ohjelmistohallinnollisiin kysymyksiin keskittymiseen teknisten ohjelmistokehityskäytänteiden sijaan. Menetelmässä esiintyy myös haasteita kuten se, että Scrum ei määrää tiettyjä työtapoja eikä ota kantaa teknisiin käytäntöihin. Tästä johtuen näiden seikkojen hallinta jää organisaatioiden omalle vastuulle. Luvussa kolme käsitellään enemmän sitä, miten Scrumia hyödynnetään projektinhallinnassa.

Zayat ja Senvar (2020) esittelevät, että Scrumin vaiheet ovat tuotteen kehitysjonon määritys, sprintin suunnittelu, sprintti, sprintin valmistuminen sekä sprintin katselmointi. Tuotteen kehitysjonon määrityksessä käydään läpi asiakkaan vaatimuksien pohjalta luodut käyttäjätarinat ja asetetaan ne järjestykseen perustuen tuotteen tärkeimpiin osiin ja ominaisuuksiin, jotka on löydyttävä tuotteesta. Suunnitteluvaiheessa kehitysjonon kohteiden eli käyttäjätarinoiden työstäminen aloitetaan niiden prioriteetin perusteella ja tehdään päätös siitä, miten kohteet kehitetään. Sprintissä kehitetään valittuja käyttäjätarinoita siten, että sprintin jälkeen ne ovat valmiita ja testattuja toimitettavaksi uutena tuotteen lisäyksenä. Sprinttiin sisältyy tiimin päivittäiset Scrum-palaverit, joissa varmistetaan, että kaikki etenee suunnitellusti. Sprinttiä voidaan ajatella projektina, jossa on tietyt tavoitteet ja aikaraja, jolloin lopussa kehitetyt ominaisuudet ovat valmiita toimitettavaksi. Jokaisen sprintin jälkeen asiakas tai sidosryhmät voivat suorittaa esimerkiksi käyttäjätestausta ja tarkastella täyttyykö heidän vaatimukset. Tämän jälkeen tarkastellaan kulunutta sprinttiä eli sitä mikä toimi ja mikä ei sekä sitä, mikä on tärkeää tulevia sprinttejä varten. Sama sykli toistuu uudelleen, seuraten edellä mainittuja vaiheita. (Zayat ja Senvar 2020).

Scrum-tiimi koostuu tuoteomistajasta (engl. *Product Owner*), kehitystiimistä sekä Scrum Masterista (Zayat ja Senvar 2020). Alsaqqa, Sawalha ja AbdelNabi (2020) kertovat, että tuoteomistaja määrittelee kehitystiimin tavoitteet asiakastarpeiden pohjalta, muuttaen vaaditut ominaisuudet käyttäjätarinoiksi tuotteen kehitysjonoon. Heidän mukaansa Scrum Master on niin sanotusti projektipäällikkö, joka vastaa Scrumin arvojen ja sääntöjen noudattamisesta sekä viestinnästä eri sidosryhmien kanssa. Lisäksi Scrum Masterin rooliin kuuluu kehitysjonon edistymisen seuranta, tiimin toiminnan optimoiminen ja mahdollisten esteiden poistaminen. Kehitystiimi tekee yhteistyötä saavuttaakseen sprinttiin asetetut tavoitteet. Alsaqqan, Sawalhan ja AbdelNabin (2020) mukaan kehitystiimin tehtäviin kuuluu käyttäjätarinoiden analysointi ja niiden vaatimusten pohjalta ohjelmiston suunnittelu, kehitys ja testaus. Tiimin vastuulla on myös työmäärän arviointi, johon Scrumin yhteydessä käytetään usein seuraavassa luvussa esiteltävää Planning Pokeria. Tiimi jakaa keskenään sprintin vastuualueet eikä kehitystiimin henkilöille ole määritelty perinteisiä ohjelmistokehityksen rooleja kuten suunnittelija, testaaja tai kehittäjä (Chaudhari ja Joshi 2021). Koska tiimin jäsenille ei ole selkeästi määriteltyjä rooleja ja vastuualueita, on riskinä, että tiimin jäsenten kesken esiintyy vastuiden laiminlyöntiä (Alsaqqa, Sawalha ja AbdelNabi 2020).

Aziz Butt ym. (2022) tuovat esiin sen, että ketterät menetelmät eivät välttämättä sovellu yhtä tehokkaasti suurille tai laajoille projekteille. Tämän seurauksena on kehitetty uusia sovelluksia niiden käytön tukemiseksi ja mahdollistamiseksi myös suurissa projekteissa. Scrumiin pohjautuen on kehitetty esimerkiksi SoS (Scrum of Scrums) sekä LeSS (Large Scale Scrum). Srivastava, Bhardwaj ja Saraswat (2017) kertovat, että Scrum of Scrumia käytetään organisaatioissa, joissa työskentelee usein erillisiä Scrum-tiimejä rinnakkain. Tässä menetelmässä kokouksia laajennetaan ylemmälle tasolle, jolloin jokaisen Scrum-tiimin päivittäisestä kokouksesta yksi henkilö osallistuu projektin laajemman tason SoS-kokoukseen. Tasoja voi olla useita projektin laajuudesta riippuen. LeSS-menetelmässä useat tiimit suorittavat samaa sprinttiä samanaikaisesti saman tuotteen kehitysjonosta (Almeida ja Espinheira 2021). Srivastavan, Bhardwajin ja Saraswatin (2017) mukaan LeSS:in aikana ohjelmistoa toteutetaan ja tallennetaan sprintin aikana keskitettyyn tallennuspaikkaan. Sprintin jälkeen, testausvaiheessa kaikkien tiimien toteuttamat osat integroidaan keskenään. Tiimien välinen koordinointi tapahtuu päivittäisten Scrum-kokousten ja keskitetyn tuen avulla. LeSS-menetelmä on erityisesti osoittanut sen, että Scrumin skaalautuvuus ei ole ongelma.

2.3 Planning Poker

Planning Poker on yhteisymmärrykseen perustuva arviointimenetelmä, jossa käyttäjätarinat arvioidaan tuotteen kehitysjonosta yksitellen. Menetelmässä keskustelu on avainasemassa ja se toimii mahdollistajana menetelmän onnistuneelle käytölle (Poženel ym. 2024). Naikin ja Divyan (2022) mukaan Planning Poker on laajasti käytetty työmäärän arviointimenetelmä, jonka avulla voidaan luoda arvio siitä mahtuuko jokin ominaisuus sprinttiin mukaan. He kertovat menetelmää käytettävän usein osana Scrum viitekehystä, jossa käyttäjätarinoiden vaatima työmäärä määritetään keskimääräisesti pokerikorttien arvioiden perusteella. Jokaiselle käyttäjätarinalle annetaan sen vaatimaa työmäärää kuvaava arvo, mitä kutsutaan tarinapisteeksi (Alsaadi ja Saeedi 2022). Pokerikorttien sisältämät arvot tarkoittavat tarinapisteitä. Naikin ja Divyan (2022) mukaan menetelmän avulla tiimit arvioivat ominaisuuden kehitykseen vaadittavan työmäärän ilman muiden tahojen vaikutusta. Tämä tekee työmäärän arvioinnista ennustettavaa ja johdonmukaista. Menetelmä myös parantaa yleisesti työmäärän arvioinnin lopputulosta, sillä se tuo yhteen erilaisia näkökulmia. Sen avulla voidaan saada luotettavia arvioita erityisesti silloin, kun mukana on asiantuntevia toimialaosaajia (Poženel ym. 2024).

Naik ja Divya (2022) mukaan Planning Poker on kasvattanut suosiotaan nykyisillä markkinoilla, joissa kilpailu on kovaa. Markkina-arvo on kasvanut erityisesti ajan säästämisen ja työn optimoinnin ansiosta. Nämä ominaisuudet johtavat tiimejä oikeaan suuntaan parhaiden mahdollisten tulosten saavuttamiseksi. Työkalu mahdollistaa tehokkaat työmäärän arviointisessiot tiimeille, jonka jäsenet eivät toimi samassa sijainnissa. Esimerkiksi Scrum Master voi kirjautua järjestelmään ja luoda uuden työmäärän arviointisession ja lähettää kutsulinkin tiimin jäsenille. Naik ja Divya (2022) määrittelevät yhdeksi tämän menetelmän vahvuuksista sen, että jokainen tiimin jäsen tekee oman arvion ennen kuin tietää muiden tekemiä arvioita, jolloin niin sanottu ’ankkurointi’ ilmiö vähenee. Heidän mukaansa ankkurointia esiintyy usein tiimityöskentelyssä tarkoittaen sitä, että yhden henkilön tekemä arviointi vaikuttaa muiden tekemiin arvioihin, jolloin arvioinnin tarkkuus sekä todenmukaisuus vähenee. Seuraavassa luvussa käsitellään projektinhallintaa ketterissä ympäristöissä, sekä Scrumin periaatteiden hyödyntämistä ohjelmistoprojektien hallintaan.

3 Projektinhallinta ketterissä ympäristöissä

Projektinhallinta tarkoittaa suunnittelua, organisointia ja seurantaa. Lisäksi se on kaikkien projektin osa-alueiden hallintaa, jossa varmistetaan projektin tavoitteiden saavuttaminen sovitussa aikataulussa, budjetissa ja määritettyjen suorituskriteerien puitteissa (Venczel, Berényi ja Hriczó 2021). Hayatin ym. (2019) mukaan projektinhallinnalla on keskeinen rooli ohjelmistoteollisuudessa. Heidän mukaan ohjelmistoprojektien onnistumista mitataan kolmoisrajoitteiden eli ajan, kustannuksien ja laajuuden avulla. Nämä ovat suoraan riippuvaisia projektin vaatimusten kanssa. Tehokkaan projektinhallinnan tärkeys korostuu, jotta ohjelmistotuotteen onnistunut toimitus varmistetaan. Chaudhari ja Joshi (2021) kertovat asiakastyytyväisyyden hallinnan olevan helpompaa ketterissä projekteissa, sillä asiakkaan vaatimuksia otetaan huomioon jokaisessa iteraatiossa. Heidän mukaansa myös ohjelmiston laadunhallinta helpottuu iteraatioiden ansiosta. Iteratiivisen prosessin avulla mahdollistetaan jatkuva testaus ja validoinnin hallinta. Jatkuva asiakkaan muuttuviin vaatimuksiin reagoiminen on yhtä aikaa ketterien menetelmien suurin vahvuus, mutta myös riski sillä työmäärä voi kasvaa suureksi ja projekti voi laajentua alkuperäisen määrittelyn ulkopuolelle (Aziz Butt ym. 2022).

Perinteisesti johtoryhmät seuraavat projektin etenemistä suunnitellakseen ja hallitakseen ohjelmiston kehitystä, testausta ja toimittamista aikataulun puitteissa (Mallidi ja Sharma 2021). Alsaadi ja Saeedi (2022) määrittävät ketterien kehitystiimien olevan itseohjautuvia, mikä tarkoittaa, että tiimit ohjaavat koko kehitysprosessia suunnittelusta käyttöönottoon saakka. Ketterä kehitystiimi on monitoiminnallinen, ja sillä on tarvittava osaaminen toimittaa iteraatioiden vaatimukset sisältäen kaikki kehitysvaiheet kuten suunnittelu, kehitys ja testaus. Thesingin, Feldmannin ja Burchardtin (2021) mukaan projektitiimi kehittää suunnitelmaa vaiheittain ja välituloksia koordinoidaan asiakkaan kanssa lyhyissä sykleissä. Heidän mukaan perinteisessä projektinhallinnassa puolestaan on periaatteena kokonaisvaltainen ennakkosuunnittelu, vakaus ja pitkän aikavälin näkökulma. Perinteisissä menetelmissä myös projektin laajuus on tiedossa ennustettavalla kehityksellä. Ketterässä projektinhallinnassa suunnittelu on vaiheittaista, jatkuvaa ja joustavaa. Lisäksi se on lyhyeen aikaväliin kohdistettua perustuen pitkän aikavälin visioon. Ketterässä projektinhallinnassa ei keskitytä laajaan ennakkosuunnitteluun tai suunnitelmaa seuraavaan tarkkaan toteutukseen (Thesing, Feldmann

ja Burchardt 2021).

Työmäärän arviointi on oleellinen osa ketterien projektien hallintaa ja iteraatioiden suunnittelua, ja sitä käsitellään tarkemmin luvussa neljä. Toinen asia mitä Alsaadin ja Saeedin (2022) mukaan iteraatioiden suunnitteluvaiheessa käsitellään työmäärän arvioinnin lisäksi on tiimin velositeetti eli nopeus. Tiimin velositeetti tarkoittaa sitä määrää tarinapisteitä, joita tiimi voi suorittaa yhden iteraation aikana. Velositeetin perusteella voidaan arvioida tarvittavien iteraatioiden määrää koko projektin suorittamiseksi. Velositeettiin vaikuttavat useat tekijät kuten tiimin kokoonpano, projektin ala sekä käytetyt työkalut ja teknologiat (Alsaadi ja Saeedi 2022).

Alsaadin ym. (2021) mukaan ketterien menetelmien yleistyminen on samaan aikaan lisännyt tehokkaan projektinhallinnan ja suunnittelun haasteita. Nämä haasteet ovat merkittävimpiä tiimeille, joissa on kokemattomia jäseniä, sillä vaatimukset muuttuvat jatkuvasti ja työn eteneminen riippuu jäsenten tehokkuudesta. Myös Marnadan ym. (2022) mukaan mahdollisia riskejä esiintyy jokaisessa sprintissä. Näitä ovat esimerkiksi projektin laajuuden kontrolloimaton kasvu, epärealistiset odotukset sekä yhteistyön ja kommunikaation puute. Chaudhari ja Joshi (2021) määrittelevät ketterien menetelmien haasteita projektinhallinnan näkökulmasta. Haasteena on esimerkiksi suunnittelun laadun kärsiminen nopean kehityksen saavuttamiseksi. Ohjelmiston vaatimusten yksityiskohtainen tarkastelu vaatii paljon aikaa, johon ei aina ole mahdollisuutta. Aziz Butt ym. (2022) puolestaan esittelevät ihmisiin ja rooleihin liittyviä riskejä. Heidän mukaan kehitystiimiltä vaaditaan erityisen paljon ketterissä projekteissa, sillä heidän tulee olla taitavia kehittäjiä ja tiimityöskentelijöitä. Tämän lisäksi heidän tulee omata valmiudet vastuunottokykykyyn, suunnitteluun, ongelmanratkaisuun sekä muutoksiin sopeutumiseen. Myöskään tiivistä kommunikaatiota johdon ja tiimien välillä ei aina ole helppoa toteuttaa (Chaudhari ja Joshi 2021). Alsaadin ym. (2021) mukaan voidaan todeta ketterien projektien hallinnan sisältävän haasteita sekä riskejä ketterien menetelmien tehokkaasta ja toimivasta luonteesta huolimatta. Seuraavaksi käsitellään sitä, miten Scrumia sovelletaan ohjelmistoprojektien hallintaan, sillä ketterät periaatteet vaikuttavat myös projektinhallinnassa.

3.1 Scrum projektinhallinnassa

Scrum keskittyy päivittäiseen projektinhallintaan (Hayat ym. 2019). Aziz Buttin ym. (2022) mukaan Scrumissa toteutettavia lyhyitä sprinttejä on helppoa hallita, sillä sprintin jokaisena päivänä pidetään kokous hallinnan edistämiseksi. Kokouksessa käydään läpi ajankohtaiset asiat ja varmistetaan, että sprintin tavoitteet edistyvät. Näissä on kuitenkin haasteena se, että kaikki kehitystiimin jäsenet eivät osallistu kokouksiin tai eivät ole aktiivisia niissä. Jokaisen sprintin jälkeen tulos esitellään asiakkaalle palautteen saamista varten. Jos muutospyyntöjä esiintyy ne pyritään huomioimaan ja sijoittamaan seuraavaan sprinttiin. Muutospyyntöihin välittömästi reagoiminen on yksi Scrumin projektinhallinnan perusta (Aziz Butt ym. 2022). Projektinhallinta Scrumissa on läpinäkyvää, sillä tiimillä on näkyvyys erilaisten kokousten kautta esimerkiksi viestintään ja tuoteomistajalta saatuun palautteeseen kehitysprosessin aikana (Alsaqqa, Sawalha ja AbdelNabi 2020).

Tuotteen kehitysjonon hallinta on tuoteomistajan vastuulla. Zayatin ja Senvarin (2020) mukaan tuoteomistajan rooli projektinhallinnassa on asiakasvaatimusten hallinta eli toisin sanoen tuotteen kehitysjonon hallinta. Tämä tarkoittaa kehitysjonon kohteiden määrittämistä käyttäjätarinoiksi asiakasvaatimusten perusteella sekä niiden järjestelemistä projektin tavoitteita parhaiten palvelevalla tavalla. Tarvittavat muutokset tulisi siis aina esittää tuoteomistajalle. Lisäksi tuoteomistajan vastuulla projektinhallinnassa on tuotteen kehitysjonon selventäminen kehitystiimille ja toteutettavien käyttäjätarinoiden hallinnointi. Projektin menestymisen takaamiseksi kaikkien tulee kuunnella tuoteomistajan päätöksiä, jotka on tehty tuotteen kehitysjonon sen hetkisen tilanteen perusteella. Zayat ja Senvar (2020) esittelevät Scrum Masterin roolin, jonka tehtävänä on vastata Scrum viitekehyksen soveltamisesta projektiin. Hänen tulisi työskennellä vain yhden projektin parissa kerrallaan voidakseen antaa tälle täyden huomionsa. Vaikka Scrum Masterilla ei ole johtavaa roolia, hänen panoksensa on tärkeä projektin tehokkuuden lisäämiseksi (Zayat ja Senvar 2020). Sprinttien työmäärän arvioinnin hallinta on kriittistä Scrum Masterille, jotta tuote saadaan toimitettua ajoissa mahdollisimman korkealla laadulla (Mallidi ja Sharma 2021). Työmäärän arviointia ja sen merkitystä käsitellään tarkemmin luvussa neljä.

Hayatin ym. (2019) mukaan ketterien ohjelmistoprojektien haasteena on ajan ja kustannusten hallinta, erityisesti samanaikaisesti. He kertovat Scrumin mahdollistavan projektien kus-

tannustehokkaan tuotettavuuden, sillä etukäteen määritellyt sprintit auttavat aikataulun sekä budjetin hallinnassa. Kuitenkin Ionelin (2019) tutkimuksen mukaan Scrumia käyttävissä projekteissa ei ole laajaa näkyvyyttä koko projektin tilanteeseen sprinttien ulkopuolella, minkä seurauksena on vaikeaa arvioida etukäteen koko projektin kestoa tai kustannuksia. Tämä voi muodostua rajoitukseksi Scrumin käytölle erityisesti tilanteissa, joissa ulkoiset asiakkaat kilpailuttavat potentiaalisia toimittajia tarjouskilpailun avulla. Srivastava, Bhardwaj ja Saraswat (2017) kertovat, että projektin etenemisen tehokkuudessa esiintyy haasteita, sillä tiimeille annetaan usein vain peruskoulutus Scrumista. Näiden tiimien tehokkuus on heikompaa verrattuna hyvin koulutettuun Scrum-tiimiin. Heidän mukaansa tätä voidaan kehittää tulevaisuudessa kouluttamalla organisaation yksiköitä ymmärtämään Scrumin toimintaperiaate sekä näyttää esimerkkejä projekteista, jotka ovat saaneet huipputuloksia Scrumin avulla (Srivastava, Bhardwaj ja Saraswat 2017). Ionelin (2019) mukaan Scrumin käytössä on haasteena on tuotteen kehitys ulkoiselle asiakkaalle, jolloin asiakkaalta vaaditaan paljon sitoutumista. Asiakkaan esimerkiksi tulee olla säännöllisesti saatavilla testaamaan kehitettyä tuotetta sekä antamaan palautetta. Scrumissa asiakkaan näkemyksellä on suuri merkitys kehitysprosessiin sekä lopputulokseen. Voidaankin todeta, että yksi Scrumin haasteista on samalla sen vahvuus eli asiakkaan osallistuminen kehitysprosessiin (Ionel 2019).

4 Työmäärän arviointi ketterissä projekteissa

Ketterissä ohjelmistoprojekteissa työmäärän arviointia tehdään projektin elinkaaren aikana eri tasoilla, kuten tarjous-, julkaisu- ja sprinttitasolla (Mallidi ja Sharma 2021). Arviointiprosessissa pyritään ennustamaan vaadittavaa työmäärää. Näiden pohjalta voidaan muodostaa arvio projektiin tarvittavista resursseista (Ramessur ja Nagowah 2020). Projektin johtaja tekee yhteistyötä tiimin kanssa työmäärän arvioinnin ohessa (Aziz Butt ym. 2022). Poženelin ym. (2024) mukaan ryhmäarviointimenetelmiä, kuten Planning Pokeria, käytetään paljon. Ne osallistavat koko tiimin arviointiprosessiin ja yhdistävät prosessissa useita asiantuntijanäkemyksiä. Asiantuntijapohjaisissa menetelmissä tiimin jäsenet osallistuvat työmäärän arviointiin perustelemalla arvioita omalla kokemuksellaan. Arvoinnissa hyödynnetään tietoa projektin vaatimuksista, koosta sekä käytettävissä olevista resursseista ja työkaluista (Alsaadi ja Saeedi 2022). Tällaisissa arviointimenetelmissä avoimella keskustelukulttuurilla ja viestinnällä on tärkeä merkitys. Niitä tuetaan esimerkiksi päivittäisillä kokouksilla, joissa varmistetaan, että tiimillä on selkeä ymmärrys sekä etenemissuunnitelmasta ja aikataulusta. Nämä asiat käydään läpi, jotta vaadittavien ominaisuuksien toimitus varmistetaan (Mallidi ja Sharma 2021).

Alsaadin ja Saeedin (2022) mukaan ketteriä menetelmiä kuten Scrumia käytettävissä projekteissa projektin vaatimukset esitetään käyttäjätarinoina. Tarinapiste on yleisin mittayksikkö yksittäisen käyttäjätarinana työmäärän arvon kuvaamiseen. Käyttäjätarinoita arvioidaan eri tekijöiden mukaan ja arvioiden perusteella päätetään, mitkä käyttäjätarinat otetaan sprinttiin mukaan. Tavoitteena on, että ne ovat toimitettuna seuraavan sprintin loppuun mennessä. Ketterissä projekteissa arvioidaan jokaisen sprintin vaadittavaa työmäärää sen sijaan, että arvioitaisiin koko projektiin vaadittava työmäärä (Alsaadi ja Saeedi 2022). Tiimit pyrkivät tuottamaan tarkkoja arvioita saavuttaakseen projektin onnistuneen toteutuksen (Alsaadi ym. 2021).

Työmäärän arvioinnilla on keskeinen rooli ohjelmistokehityksen projekteissa, sillä se vaikuttaa suoraan kustannusten, resurssien ja aikataulujen hallintaan, mikä on olennaista liiketoiminnan kannalta (Naik ja Divya 2022). Ramessurin ja Nagowahin (2020) mukaan arvioinnin tarkkuus vaikuttaa suuresti realististen suunnitelmien luomiseen ja sopimusten täyttämiseen.

Tämän lisäksi projektin onnistuminen riippuu arvioinnin tarkkuudesta. Työmäärää arvioidessa ketterässä ohjelmistokehityksessä tulee ottaa huomioon prosessin dynaaminen luonne. Sen vuoksi on haastavaa seurata, ylläpitää ja määrittää vaadittavaa työmäärää toivotun lopputuloksen saavuttamiseksi (Poženel ym. 2024). Arviointimenetelmien soveltaminen voi olla haastavaa, sillä projektin vaatimuksien määrää säädellään jatkuvasti ja uusia ominaisuuksia voi nousta esiin projektin myöhemmissä vaiheissa (Ramessur ja Nagowah 2020). Poženelin ym. (2024) mukaan perinteiset projektin alussa määritellyt suunnittelu- ja arviointimallit eivät ole usein optimaalisia ohjelmistoprojekteille. Ne rajoittavat mahdollisuuksia käsitellä kohdattuja haasteita. Seuraavaksi esitellään miten Planning Poker soveltuu työmäärän arviointiin ja mitä haasteita tähän liittyy.

4.1 Planning Poker työmäärän arvioinnissa

Planning Poker -menetelmää käytetään käyttäjätarinoiden arvioimiseen työmäärän arviointisessioissa, joissa kehitystiimi arvioi käyttäjätarinoiden vaativuuden yksitellen (Poženel ym. 2024). Poženelin ym. (2024) mukaan Planning Poker -menetelmässä työmäärän arviointiprosessi alkaa, kun ensimmäinen käyttäjätarina otetaan tuotteen kehitysjonosta. Tuoteomistaja esittelee kyseisen käyttäjätarinan vaatimukset, ja tiimin jäsenet voivat esittää kysymyksiä sekä keskustella ominaisuuden yksityiskohdista. Tiimin jäsenten ei tulisi keskustella suoraan arvioista, vaan esittää omat arvionsa vasta samanaikaisesti. Jos arviot poikkeavat merkittävästi toisistaan, tiimin tulee keskustella ja perustella arvioitaan, minkä jälkeen esitetään uudet arviot. Tätä toistetaan, kunnes saavutetaan yhteisymmärrys, jolloin yksittäisen käyttäjätarinan arviointi on valmis. Samaa prosessia toistetaan kunnes tuotteen kehitysjonon kaikki käyttäjätarinat on käyty läpi (Poženel ym. 2024).

Naikin ja Divyan (2022) mukaan menetelmässä iteraation työmäärän määritys perustuu Fibonaccin lukujonoon, jossa kaksi edellistä lukua summaamalla saadaan jonon seuraava luku. Fibonaccin lukujonon luvut tarkoittavat tarinapisteitä. Tarinapisteissä esimerkiksi yksi tarkoittaa hyvin pientä tehtävää ja 20 selkeästi suurta tehtävää. Heidän mukaansa jokaiselle lukuarvolle on määritetty arviointikortti, joka edustaa tiettyä työmäärän vaativuutta. Kortit paljastetaan samanaikaisesti kierroksen jälkeen, kun kehitettävään ominaisuuteen liittyvä keskustelu on käyty. Heidän mukaansa menetelmä on hyvä työmäärän arviointiin, sillä se pe-

rustuu keskiarvoihin ja antaa kuvan tiimin yhteisymmäryksessä tehdystä arviosta. Planning Poker -työkalussa on etuna esimerkiksi helppo käytettävyys yhdessä monien yleisimpien teknologioiden kanssa sekä käyttäjäystävällinen käyttöliittymä (Naik ja Divya 2022).

Planning Poker -menetelmään liittyy myös haasteita. Yksi merkittävimmistä rajoituksista on se, että menetelmä perustuu täysin tiimin kokemukseen ja heidän tekemiinsä arvioihin. Toisin sanoen se ei ota huomioon kokemattomia tiimejä, sillä se ei tarjoa apua ja ohjausta käyttäjätarinan pistemäärän tarkkaan arvioimiseen (Alsaadi ym. 2021). Menetelmän haasteena on myös se, että arvoiden muodostaminen voi olla aikaavievää ja tehdystä arviosta voi olla vaikea päästä yhteisymmärrykseen (Poženel ym. 2024). Alsaadin ym. (2021) mukaan tiimin jäsenten tulisi jakaa osaamistaan. Tämä voi tarkoittaa teknistä osaamista tai kokemusta vastaavista projekteista. Myös projektin dokumentaatioita tulisi hyödyntää. Tällöin arvioinniprosessista saadaan sujuva ja tulokseksi saadaan mahdollisimman tarkka arvio. Heidän mukaansa arviointiprosessin onnistunut toteutus voi olla haasteellista, jos tiimin jäsenet työskentelevät etänä eri aikavyöhykkeillä ja fyysisesti eri paikoissa. Tiimin hajautuneisuus voi johtaa siihen, että tiimin keskeinen tiedonjako ole sujuvaa. Johdonmukainen kommunikaatio edesauttaa hyvien arviointien tekemistä (Alsaadi ym. 2021).

Alsaadin ja Saeedin (2022) tutkimuksen mukaan yksi asiantuntijapohjaisten haasteista on epätarkat arviot, jotka johtuvat tiimin jäsenten kokemattomuudesta. Kokemattomilla jäsenillä ei ole tarpeeksi tietoa ja ymmärrystä projektin tehtävistä tai siitä miten työmäärää arvioidaan. Alsaadi ja Saeedi (2022) mainitsevat esimerkkinä vertailun asiantuntijoiden ja opiskelijoiden arvoiden eroista. Vertailussa arviot tehtiin käyttäen Planning Poker -menetelmää. Vertailun perusteella kokeneiden asiantuntijoiden arviot olivat huomattavasti tarkempia (Alsaadi ja Saeedi 2022). Tiimin asiantuntemus ketteristä menetelmistä ja aiempi kokemus ketterissä tiimeissä toimimisesta ovat ratkaisevassa roolissa. Kokenut ja pidempään yhdessä toiminut tiimi pystyy suoriutumaan arvioinnista tehokkaammin ja tarkemmin kuin kokematon tiimi. Haasteista huolimatta menetelmä on saavuttanut suosiota nykyisten markkinoiden kilpailussa aikaa säästävänä työn optimointivälineenä, joka ohjaa tiimiä oikeaan suuntaan, jotta tiimi voi saavuttaa parhaan mahdollisen tuloksen (Naik ja Divya 2022). Seuraavassa luvussa käsitellään työmäärän arviointiin liittyviä haasteita. Osa näistä haasteita esiintyy myös Planning Pokeria käyttäessä.

4.2 Työmäärän arvioimisen haasteet

Useat ketterät ohjelmistoprojektit ylittävät budjetin puutteellisten arvioiden vuoksi. Puutteelliset arviot aiheuttavat epätarkkaa suunnittelua ja vaikeuttavat ohjelmiston kehitystä. Nämä arviot voivat johtaa keskeneräisen tuotteen toimittamiseen asiakkaalle ja edelleen tyytymättömään asiakkaaseen (Mallidi ja Sharma 2021). Virheellinen työmäärän arviointi voi johtaa edellisen sprintin käyttäjätarinoiden suorittamiseen seuraavassa sprintissä. Tämän seurauksena voi olla budjetin ylitys, viivästynyt toimitus ja virheelliset aikatauluarviot (Madya, Budiardjo ja Mahatma 2022). Sudarmaningtyasin ja Mohamedin (2021) mukaan ohjelmistokehityksessä kustannusten arviointi on tärkeää budjetin ylittämisen välttämiseksi. Heidän mukaansa projektin kustannukset perustuvat arvioituun työmäärään, joten tarkka työmäärän arviointi on kriittinen osa ohjelmistoprojektia.

Jos työmäärän arviointi on ollut puutteellista tai virheellistä, joutuu esimerkiksi tuotteenomistaja tai Scrum Master tekemään kompromisseja suunnittelussa (Sudarmaningtyas ja Mohamed 2021). Käytännössä he joutuvat tekemään päätöksen siitä toimitetaanko lopputuote asiakkaalle huonommalla laadulla vai keskeneräisenä (Mallidi ja Sharma 2021). Alsaadin ym. (2021) mukaan tämä voi lisää painetta tiimin jäsenille, johtuen johdon motivaatiosta projektin toimittamiseen pysyen tietyssä aikataulussa ja budjetissa. Tiimin jäsenet saattavat pelätä myöhästyneitä toimituksia, mikä voi johtaa käyttäjätarinoiden vaatiman työmäärän aliarvioimiseen. Työmäärän arvioinnin haasteet ovat moninaisia ja ne voivat vaikuttaa merkittävästi projektin etenemiseen sekä lopputulokseen.

Myös Mallidi ja Sharma (2021) esittelevät tutkimuksessaan yleisiä haasteita ketterien projektien työmäärän arvioinnissa. Ensimmäisenä haasteena on arvioinnin vaativuus, jolloin tiimit eivät tiedä miten määrittää tarinapisteitä tuotteen kehitysjonon kohteille. Tämä liittyy edellisessä luvussa käsiteltyyn asiantuntijuuteen, eli tiimin jäsenillä tulee olla tarpeeksi kokemusta ja tietämystä niin ketteristä menetelmistä, kuin ohjelmistokehityksestä, pystyäkseen arvioimaan tuotteen kehitysjonon kohteita tarinapisteillä. Myös Alsaadin ym. (2021) mukaan kokemattomat tiimin jäsenet saattavat arvioida käyttäjätarinoita virheellisesti, johtuen heidän heikosta taustastaan tärkeillä osa-alueilla ohjelmistokehityksen saralla. Tämä tarkoittaa sitä, että tiimin jäsenten tulisi ymmärtää sprinttien keskeiset teemat tai toimintaympäristöt, jotta he voivat arvioida ja suorittaa tehtäviä tehokkaasti sprinttien aikana. Jos asiantuntijajäseniä

puuttuu arviointisessiosta, päätöksentekoprosessiin vaikuttaa merkittävästi kokemattomien jäsenten tekemät arviot. Tämän johdosta tiimit saattavat arvioita tehdessään pidentää tehtävien suorittamiseen tarvittavaa aikaa, mikä johtaa vaaditun työmäärän yliarviointiin. Myös Mallidin ja Sharman (2021) mukaan sprinttien työmäärää saatetaan yliarvioida, mikäli vastaavaa työtä ei ole aiemmin tehty. Heidän mukaansa kehittäjät usein arvioivat tarinapisteet liian korkeiksi työskennellessään esimerkiksi uuden teknologian parissa.

Mallidin ja Sharman (2021) mukaan toisena haasteena on se, että ei ole olemassa täysin standardoituja arviointimenetelmiä. Tarinapistein tehdyt arviot ovat aina suhteellisia määritelmän vaihdellessa projektista toiseen. Haastetta voi lisätä se, että aiemmista projekteista ei ole saatavilla tietoa, joka olisi vertailukelpoista nykyisen projektin koon, työmäärän ja velositeetin suhteen. Tämä tarkoittaa, että vastaavia tietoja ei ole dokumentoitu aiemmissa projekteissa, jotta niitä voitaisiin käyttää vertailupohjana samankaltaisten tehtävien suorittamiseen. Myös liian laajat käyttäjätarinat tuotteen kehitysjonossa vaikeuttavat työmäärän arviointia (Mallidi ja Sharma 2021), mitä pienempiin paloihin tehtävät on pilkottu, sitä helpompi niille on määrittää paikkansapitävät tarinapisteet. Vaihtelevat arviontimenetelmät, työmäärän jatkuva lisääntyminen ja laajojen käyttäjätarinoiden hallinta tuovat omat haasteensa arviointiprosessiin.

5 Yhteenveto

Tutkielmassa tehtiin katsaus ketterien menetelmien käyttöön ohjelmistoprojekteissa sekä tutustuttiin tarkemmin projektinhallintaan ja työmäärän arviointiin. Ketterien menetelmien suosio on kasvanut ohjelmistoprojekteissa korvaten perinteisiä malleja kuten vesiputousmallin. Merkittävimpiä eroja perinteisten ja ketterien menetelmien välillä ovat ne, että perinteisissä malleissa asiakkaan osallistuminen rajoittuu vain suunnitteluvaiheeseen sekä muutoksiin on lähes mahdotonta reagoida projektin myöhemmissä vaiheissa. Ketterät menetelmät perustuvat lyhyempiin iteraatioihin, jolloin vaatimusmäärittelyä, asiakkaan kanssa keskustelua ja projektinhallintaa voidaan tehdä läpi koko projektin. Ketterien menetelmien yksi keskeisimmistä ominaisuuksista sekä vahvuuksista on kyky sopeutua muutoksiin.

Tutkielmassa esiteltiin Scrumia menetelmänä yleisellä tasolla, sekä sen sovellusta ohjelmistoprojektien hallintaan. Scrumin vahvuudet ohjelmistokehityksessä ovat kehitysprosessin tuottavuus, nopean etenemisen mahdollistaminen sekä muutosten huomioiminen projektin jokaisessa vaiheessa. Tämän johdosta se on saavuttanut suositun aseman sekä ohjelmistokehityksessä että projektinhallinnassa. Sen merkittävimmät vahvuudet ovat tuottavuus sekä asiakastyytyväisyys. Scrumin etuna on lisäksi skaalautuvuus laajempiin projekteihin, sillä skaalautuvuuden puute usein esiintyy haasteena ketterissä menetelmissä. Nämä edut selittävät Scrumin laajaa käyttöastetta sekä suosiota. Projektinhallinnan vahvuuksia ovat lyhyiden muutaman viikon pituisten sprinttien helppo hallinta sekä selkeät roolit, kuten kehitysjonon hallinta tuotteen omistajana vastuulla tai sprinttien työmäärän arvioinnin hallinta Scrum Masterin vastuulla. Scrum antaa organisaatioille selkeän viitekehyksen, jonka mukaan toimittaessa saavutetaan hyviä tuloksia tuotteen laadusta asiakastyytyväisyyteen. Scrumin onnistunut käyttö kuitenkin vaatii asiakkaalta paljon sitoutumista, mikä voidaan todeta sekä heikkoudeksi että vahvuudeksi samaan aikaan.

Tutkielmassa käsiteltiin työmäärän arvioinnin merkitystä ketterissä projekteissa sekä esiteltiin usein Scrumin yhteydessä käytettävää työmäärän arviointimenetelmää Planning Pokeria. Työmäärän arviointi on kriittistä projektin onnistumisen kannalta, sillä se vaikuttaa aikataulussa ja budjetissa pysymiseen. Se vaikuttaa myös suoraan projektin kustannusten, resurssien ja saatavuuden hallintaan. Ketterissä menetelmissä työmäärä voi kasvaa helpos-

ti suureksi, sillä vaatimusmäärittelyä tehdään ennen jokaista sprinttiä. Työmäärän hallintaan on hyvät edellytykset ketterissä projekteissa, sillä arviointia tehdään jatkuvasti. Tästä huolimatta arviointiin liittyy myös haasteita. Merkittävimmät haasteet ovat puutteelliset arviot, jotka mahdollisesti lisäävät työmäärää tulevissa sprinteissä. Tämän haasteen ehkäisemiseksi on tärkeää panostaa tarkkoihin ja todenmukaisiin arvioihin. Planning Poker -menetelmän käyttö edesauttaa hyvien arvioiden tekemistä, sillä se pohjautuu asiantuntijoiden tekemiin arvioihin. Menetelmä sopii ketterien menetelmien yhteyteen, sillä asiantuntijoiden on helppo arvioida yksittäisiä käyttäjätarinoita ennen jokaista sprinttiä. Työmäärän arvioinnissa esiintyvät haasteet, yleisesti sekä Planning Poker menetelmässä, korostavat tiimin kokemuksen ja osaamisen merkitystä, tarvetta joustavuudelle arviointiprosessissa sekä vaatimusta jatkuvasta kehittymisestä ja virheistä oppimisesta. Kehittyneet arviointimenetelmät sekä sitoutunut ja kokenut tiimi ovat avainasemassa projektin onnistumisen kannalta. Näiden tekijöiden avulla voidaan paremmin hallita ja ratkaista työmäärän arvioinnin haasteita.

Tutkielman toteuttamisessa on hyödynnetty laajasti eri tutkimuksia, ja yhdistelty näiden havaintoja toisiinsa. Tämän tutkielman aihepiireistä on ollut saatavilla runsaasti tuoretta tutkimustietoa, joten tähän tutkielmaan hyödynnetty aineisto, yleisesti ketteristä menetelmistä, Scrumista ja työmäärän arvioinnista on ollut tuoretta. Tutkimukseen käytetyn aineiston hyvän saatavuuden ansiosta on voitu laajasti esitellä ketterien menetelmien taustaa ja periaatteita sekä keskittyä tarkasti tiettyihin menetelmiin. Tämän ansiosta myös menetelmien merkityksen tutkiminen sekä arvioiminen on ollut mahdollista.

Lähteet

Almeida, Fernando ja Eduardo Espinheira. 2021. “Large-scale agile frameworks: a comparative review”. *Journal of Applied Sciences, Management and Engineering Technology* 2 (1): 16–29.

Alsaadi, Bashaer ja Kawther Saeedi. 2022. “Data-driven effort estimation techniques of agile user stories: a systematic literature review”. *Artificial Intelligence Review* 55 (7): 5485–5516. https://doi.org/10.1007/s10462-021-10132-x.

Alsaadi, Bushra, Bashaer Alsaadi, Mashaal Alfhaid, Athir Alghamdi, Nedaa Almuallim ja Bahjat Fakieh. 2021. “Scrum Poker Estimator: A Planning Poker Tool for Accurate Story Point Estimation”. *International Journal of Computer Information Systems and Industrial Management Applications* 13:12–12.

Alsaqqa, Samar, Samer Sawalha ja Hiba AbdelNabi. 2020. “Agile software development: Methodologies and trends.” *International Journal of Interactive Mobile Technologies* 14 (11). https://doi.org/10.3991/ijim.v14i11.13269.

Aziz Butt, Shariq, Gabriel Piñeres-Espitia, Paola Ariza-Colpas ja Muhammad Imran Tariq. 2022. “Project management issues while using agile methodology”. Teoksessa *International Conference on Lean and Agile Software Development,* 201–214. Springer. https://doi.org/10.1007/978-3-030-94238-0_12.

Chaudhari, Ashvini R. ja Shashank D. Joshi. 2021. “Study of effect of Agile software development Methodology on Software Development Process”. Teoksessa *2021 Second International Conference on Electronics and Sustainable Communication Systems (ICESC),* 1–4. https://doi.org/10.1109/ICESC51422.2021.9532842.

Ciric Lalic, Danijela, Bojan Lalic, Milan Delić, Danijela Gracanin ja Darko Stefanovic. 2022. “How project management approach impact project success? From traditional to agile”. *International Journal of Managing Projects in Business* 15 (3): 494–521. https://doi.org/10.1108/IJMPB-04-2021-0108.

Hayat, Faisal, Ammar Ur Rehman, Khawaja Sarmad Arif, Kanwal Wahab ja Muhammad Abbas. 2019. “The influence of agile methodology (Scrum) on software project management”. Teoksessa *2019 20th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD),* 145–149. IEEE.

Ionel, Năftănăilă. 2019. “Critical analysys of the Scrum project management methodology”. *ANALELE UNIVERSITĂŢII DIN ORADEA* 436:436–442.

Madya, Gusti Raditia, Eko K. Budiardjo ja Kodrat Mahatma. 2022. “PREP: A Post-Requirements Effort Estimation Method in Scrum’s Sprint Grooming”. Teoksessa *2022 International Conference on Data and Software Engineering (ICoDSE),* 132–137. https://doi.org/10.1109/ICoDSE56892.2022.9972012.

Mallidi, Ravi Kiran ja Manmohan Sharma. 2021. “Study on agile story point estimation techniques and challenges”. *Int. J. Comput. Appl* 174 (13): 9–14.

Marnada, Primadhika, Teguh Raharjo, Bob Hardian ja Adi Prasetyo. 2022. “Agile project management challenge in handling scope and change: A systematic literature review”. *Procedia Computer Science* 197:290–300. https://doi.org/10.1016/j.procs.2021.12.143.

Naik, Nayana Gajanan ja T Divya. 2022. “Planning poker tool for story point estimation”. *RV Journal of Science Technology Engineering Arts and Management* 3:82–92.

Poženel, Marko, Luka Fürst, Damjan Vavpotič ja Tomaž Hovelja. 2024. “Agile Effort Estimation: Comparing the Accuracy and Efficiency of Planning Poker, Bucket System, and Affinity Estimation methods”. *arXiv preprint arXiv:2401.16152,* https://doi.org/10.1142/S021819402350064X.

Ramessur, Melvina Autar ja Soulakshmee Devi Nagowah. 2020. “Factors affecting sprint effort estimation”. Teoksessa *Advanced Computing and Intelligent Engineering: Proceedings of ICACIE 2018, Volume 2,* 507–518. Springer.

Srivastava, Apoorva, Sukriti Bhardwaj ja Shipra Saraswat. 2017. “SCRUM model for agile methodology”. Teoksessa *2017 International Conference on Computing, Communication and Automation (ICCCA),* 864–869. https://doi.org/10.1109/CCAA.2017.8229928.

Sudarmaningtyas, Pantjawati ja Rozlina Mohamed. 2021. “A review article on software effort estimation in agile methodology”. *Pertanika Journal of Science & Technology* 29 (2): 837–861. https://doi.org/10.47836/pjst.29.2.08.

Thesing, Theo, Carsten Feldmann ja Martin Burchardt. 2021. “Agile versus waterfall project management: decision model for selecting the appropriate approach to a project”. *Procedia Computer Science* 181:746–756. https://doi.org/10.1016/j.procs.2021.01.227.

Venczel, TB, László Berényi ja Krisztián Hriczó. 2021. “Project management success factors”. Teoksessa *Journal of Physics: Conference Series,* 1935:012005. 1. IOP Publishing. https://doi.org/10.1088/1742-6596/1935/1/012005.

Zayat, Wael ja Ozlem Senvar. 2020. “Framework study for agile software development via scrum and Kanban”. *International journal of innovation and technology management* 17 (04): 2030002. https://doi.org/10.1142/S0219877020300025.