Emmi Laukkarinen

SCRUMIN HAASTEET TIETOJÄRJESTELMÄKEHITYSPROJEKTEISSA

JYVÄSKYLÄN YLIOPISTO
INFORMAATIOTEKNOLOGIAN TIEDEKUNTA
2018

TIIVISTELMÄ

Laukkarinen, Emmi
Scrumin haasteet tietojärjestelmäkehitysprojekteissa
Jyväskylä: Jyväskylän yliopisto, 2018, 34 s.
Tietojärjestelmätiede, kandidaatintutkielma
Ohjaaja: Seppänen, Ville

Scrum ja muut ketterän kehittämisen menetelmät ovat yleistyneet järjestelmäkehitysmenetelminä 2000-luvun alusta lähtien. Yleisesti ketteryyden ja ketterien menetelmien eduiksi koetaan etenkin joustavuus, läpinäkyvyys ja kommunikaation korostaminen kehitysprojekteissa. Systemaattisena kirjallisuuskatsauksena toteutetussa tutkielmassa esitellään tietojärjestelmäkehitysprojekteissa esiintyviä haasteita, kun kehitysmenetelmänä käytetään Scrum-menetelmää. Tutkielmassa pyritään vastaamaan tutkimuskysymykseen: Mitkä ovat Scrum-menetelmän haasteet tietojärjestelmäkehitysprojekteissa. Haasteet on luokiteltu seitsemään ryhmään, jotka ovat kehitystiimi, asiakas, dokumentaatio, työskentely-ympäristö, Scrumin suorittaminen ja tehokkuus, vaatimukset ja järjestelmä, sekä mittaaminen ja arviointi. Jokaisessa ryhmässä esitellään kyseiseen osa-alueeseen liittyen lähdetutkimuksissa yleisimmin nousseet haasteet, sekä pyritään esittämään ratkaisumalleja haasteiden ehkäisemiseksi. Tutkielman tuloksista käy ilmi, että useiden haasteiden taustalla vaikuttaa Scrum-menetelmän puutteellinen tai heikko tuntemus, jonka vuoksi menetelmä ei anna odotettua lisäarvoa projekteille. Tutkielman tuloksia voidaan käyttää Scrum-menetelmän toteuttamisen tukena haasteiden ymmärtämisen näkökulmasta.

Asiasanat: scrum, tietojärjestelmien kehittäminen, ketterä kehittäminen, projektinhallinta

ABSTRACT

Laukkarinen, Emmi
Scrum's Challenges in Information System Development Projects
Jyväskylä: University of Jyväskylä, 2018, 34 pp.
Information Systems, Bachelor's Thesis
Supervisor: Seppänen, Ville

Scrum and other agile development methods have become more common as system development methods since the early 2000s. In general, the agility and agile methods are characterized by flexibility, transparency and emphasis on communication in development projects. The problems of information system development process when using the Scrum method are presented in this systematic literature review. The thesis aims to answer the research question: What are the challenges of the Scrum method in information system development projects. Challenges are categorized into seven categories: Development Team, Client, Documentation, Working Environment, Scrum Performance and Efficiency, Requirements and System, and Measurement and Evaluation. Each group presents the most commonly encountered challenges in the area in question and aims to present solutions to the challenges. The results of the study show that many challenges are underpinned by the inadequate or weak knowledge of the Scrum method, which means that the method does not give the expected added value for the projects. The results of this thesis can be used to support the implementation of the Scrum method from a perspective of understanding the challenges.

Keywords: scrum, software development, agile development, project management

KUVIOT

SISÄLLYS

1 JOHDANTO

Ketterällä kehittämisellä tarkoitetaan menetelmää tietojärjestelmien kehittämiselle, eli menetelmää joko kokonaan uuden järjestelmän luomiselle tai jo olemassa olevan järjestelmän jatkokehittämiselle. Ketterän kehittämisen perusajatus on tehostaa kehitystiimin toimintaa muun muassa tekemällä päätöksenteosta ja viestinnästä tasapuolista ja osallistavaa, sekä pyrkiä hyödyntämään tiimin jäsenten vahvuuksia vaiheittain tapahtuvassa järjestelmäkehityksessä. (Cockburn & Highsmith, 2001).

Ketterillä menetelmillä ja ketterällä kehittämisellä pyritään vastaamaan järjestelmäkehityksen haasteisiin, jotka liittyvät laajojen järjestelmien sekä kehitysprojektien toteuttamiseen. Järjestelmäkehityksen haasteena on usein kehitettävän järjestelmän vaatimusten muuttuminen projektin aikana, mikä tekee kehitysprosessin suunnittelusta ongelmallista. Ketterät järjestelmäkehitysmenetelmät pyrkivätkin tekemään järjestelmäkehityksestä joustavampaa ja mukautuvampaa mahdollistamalla kattavan vaatimusten suunnittelun ja määrittelyn, tehokkaan ohjelmistokehityksen, todellista käyttöympäristöä mukailevan testauksen sekä jatkuvan yhteydenpidon asiakkaan kanssa. (Jiang & Eberlein, 2009).

Tämän kandidaatintutkielman tarkoituksena on vastata kysymykseen, mitä haasteita Scrum-menetelmään liittyy ketterissä tietojärjestelmäkehitysprojekteissa. Tutkielmassa perehdytään aluksi luvussa 2 ”Ketterä kehittäminen” ketterään kehittämiseen yleisesti ja tämän jälkeen syvennytään Scrum-menetelmään. Scrumiin liittyvät haasteet on luokiteltu seitsemään ryhmään, jotka esitellään luvussa 3 ”Scrumin haasteet järjestelmäkehitysprojekteissa”. Lopuksi luvussa 4 ”Yhteenveto ja pohdinta” kerrataan tutkielmassa esitellyt haasteet ja esitellään jatkotutkimusaiheita.

Tutkielman aiheen tarkastelu on hyödyllistä ketterän kehittämisen suosion kasvusta sekä Scrum-menetelmän yleisyydestä johtuen. Projektinhallinnallisesta näkökulmasta Scrumin haasteisiin perehtymällä voidaan pyrkiä saavuttamaan kokonaisuudessaan tehokkaamman ja joustavammin etenevän järjestelmäkehitysprojektin, sillä kehitystiimien toiminnassa on Scrumin lähes 20-

vuotisesta olemassaolosta huolimatta haasteita, jotka vaikuttavat koko kehitysprojektin tehokkuuteen. Scrumin avulla voidaan saavuttaa tehokkaampi projektinhallinta, sillä Scrum on projektiviitekehyksenä mukautuva ja muokattavissa (Hu, Yuan & Zhang, 2009). Tästä syystä pyrin todentamaan tutkielmassa, että haasteiden tuntemus sujuvoittaa järjestelmäkehitystä, sillä haasteet tuntemalla ja tunnistamalla niitä voidaan myös ehkäistä ja niihin voidaan reagoida ajoissa. Haasteet tiedostamalla tiimit voivat siis pyrkiä luomaan projektille toimintatavat, jotka pyrkivät tietoisesti ottamaan huomioon mahdolliset ongelmakohdat projektin suorittamisessa ja järjestelmäkehityksessä, ja tätä kautta toimimaan tehokkaammin koko projektin näkökulmasta. Valitsin tutkielman tarkemmaksi näkökulmaksi juuri Scrum-menetelmän, sillä se ei varsinaisesti suoraan anna tehtäviä tiimille, vaan ajatuksen toimintatavoista. Tutkielman tuloksista käykin ilmi, että keskeisimpänä haasteena kehitystiimeissä on juuri menetelmän puutteellinen tuntemus ja ymmärtäminen, mikä vaikuttaa useiden pienempien haasteiden taustalla.

Tutkielma on toteutettu kirjallisuuskatsauksena, käyttäen hakusanoja ”scrum challenges”, ”scrum problems” ”scrum implementation”, ”scrum complexity”, sekä ”agile development”. Lähdemateriaali on poimittu Google Scholar- sekä IEEE-tietokannoista. Scrumin haasteisiin liittyen on vertailtu kymmenestä eri tutkimuksesta nousseita yleisiä haasteita ja luokiteltu ne erikseen omiin ryhmiinsä. Ryhmät ovat kehitystiimi, asiakas, dokumentaatio, työskentely-ympäristö, Scrumin suorittaminen ja tehokkuus, vaatimukset ja järjestelmä, sekä mittaaminen ja arviointi. Jaottelulla pyritään selkeyttämään Scrumiin liittyviä ongelmia, sekä luomaan kattavan kuvan yleisimmin Scrum-projektien aikana kohdattavista haasteista. Ryhmät ja jaottelu on toteutettu keräämällä ensin haasteet yksitellen kustakin tutkimuksesta ja vertaamalla haasteiden osuvuutta joko kehitysprojektiin tai järjestelmään liittyviksi ongelmiksi. Suurin osa tutkimusten haasteista liittyi kuitenkin enemmän kehitysprojektiin kokonaisuudessaan, joten haasteista on pyritty tunnistamaan yhteiset teemat tietojärjestelmäkehitysprojektien olennaisimpiin osa-alueisiin liittyen. Tätä kautta tutkielmaan on muodostettu seitsemän ryhmää, joissa esitellään tutkimuksissa eniten ilmi nousseet haasteet. Osa tutkimuksissa osoitetuista haasteista on karsittu pois, jotta tutkielmassa säilyisi johdonmukainen ja selkeä rakenne. Tällaiset karsitut haasteet esiintyivät esimerkiksi vain yhdessä kymmenestä tutkimuksesta ja eivät tätä kautta antaneet tarpeeksi vahvaa näyttöä Scrum-menetelmän haasteista.

Tutkielmassa suurimmaksi haasteeksi nousee Scrum-menetelmän heikko tuntemus, josta syystä useimmat haasteet seitsemästä haasteryhmästä johtuvat. Jatkotutkimuksen kannalta olisikin hyödyllistä tarkastella, kuinka menetelmä saadaan tarpeeksi hyvin ymmärretyksi tai koulutetuksi, jotta projektinhallinta olisi tehokasta ja Scrumista saataisiin todella sellaista lisäarvoa, jonka saavuttamiseen menetelmä on alun perin suunniteltu.

2 KETTERÄ KEHITTÄMINEN

Luvussa käydään läpi ketterän kehittämisen periaatteita. Alaluvussa 2.1 tarkastellaan ketteriä menetelmiä, niiden historiaa, nykypäivää ja eroavaisuuksia perinteisiin järjestelmäkehitysmenetelmiin. Alaluvussa 2.2 kerrotaan Agile Manifeston vaikutuksesta ketterään kehittämiseen. Alaluvussa 2.3 syvennytään tutkielmassa tarkasteltavaan Scrum-menetelmään ketterän kehittämisen viitekehyksenä.

2.1 Ketterät menetelmät

Ketterän kehittämisen ja ketterien menetelmien historia järjestelmäkehityksessä yltää 1950-luvulle, jolloin iteratiivinen ja inkrementaalinen kehitystapa, eli toistuva ja vähittäin kasvava kehittäminen, saivat käsitteenä alkunsa. 1950-luvun loppupuolelle sijoittuvaa NASA:n (National Aeronautics and Space Administration) ”Mercury”-projektia on kuvailtu noudattaneen ketteriä menetelmiä, vaikka kyseessä ei ollutkaan järjestelmäkehitysprojekti vaan avaruusohjelma. Projektissa noudatettiin lyhyitä, noin puolen päivän mittaisia kehitysjaksoja ja jokainen muutos suunniteltiin, toteutettiin ja testattiin kunkin kehitysjakson aikana. (Larman & Basili, 2003).

Ketterillä menetelmillä tarkoitetaankin ennen kaikkea järjestelmäkehitysmenetelmiä, joissa pyritään kiinnittämään huomiota etenkin kehitysprojektin visiointiin, sitoutumiseen, iteratiivisuuteen, palautteen antamiseen, asiakkaan sitouttamiseen ja tekniseen onnistumiseen (Highsmith & Cockburn, 2001). Nämä kuusi ketterien menetelmien ominaisuutta tiivistävät ketterän kehittämisen perusajatuksen, jossa erona perinteiseen kehittämiseen on tehdä järjestelmäkehityksestä joustavampaa ja innovatiivisempaa. Lisäksi pyritään tuottamaan ja kehittämään korkealaatuisia järjestelmiä tehokkaasti, samalla pyrkien vastaamaan ja reagoimaan asiakkaan ja sidosryhmien vaatimuksiin (Lapham, Williams, Hammons, Burton & Schenker, 2010, s. 5).

Highsmithin ja Cockburnin (2001) esittelemien ketterien menetelmien perusominaisuuksien mukaan visioinnilla pyritään varmistamaan, että kehitysprojektissa keskitytään oikeisiin asioihin kehitettävän järjestelmän kannalta, esimerkiksi tuotteen laadullisiin ominaisuuksiin. Projektiin sitoutumisella tarkoitetaan projektin kokonaisuuden hallintaa, eli projektin laajuuden, tavoitteiden ja riskien tulee olla selvitetty ja dokumentoitu. Iteratiivisuus tarkoittaa kehitysprojektin toteuttamista toistuvien vaiheiden kautta (Fowler & Highsmith, 2001) ja iteratiivisuuden tarkoituksena on kehittää järjestelmää pienissä osissa kerrallaan. Jatkuvalla palautteen antamisella tarkoitetaan eri kehitysmenetelmissä päivittäin tapahtuvaa lyhyttä palaveria, jossa projektiryhmän jäsenet kertovat mitä ovat viimeisimmäksi tehneet ja ovatko he kohdanneet esimerkiksi jotakin haasteita työssään. Jatkuvan palautteen ja tiedonjakamisen tarkoituksena on pitää kehitysryhmän jäsenet tietoisina muiden jäsenten vastuista ja tehtävien edistymisestä, ja tätä kautta lisätä kehitystehtävien tehokkuutta. Asiakkaan sitouttamisella tarkoitetaan puolestaan asiakkaan ottamista vuorovaikutuksellisesti mukaan kehitysprojektiin. Asiakkaan ja kehittäjän välisellä vuorovaikutuksella pyritään luomaan asiakkaalle paras mahdollinen tuote tai palvelu ja tätä kautta lisätä asiakkaan liiketoiminnan arvoa. Kuudentena ja viimeisenä ketterien menetelmien ominaisuutena Highsmith ja Cockburn esittelevät teknisen onnistumisen, jolla tarkoitetaan asiakkaan liiketoiminnan arvon lisäämistä kehittämällä teknisesti erinomainen tuote tai palvelu.

Esimerkki tunnetusta perinteisestä järjestelmäkehitysmenetelmästä on niin sanottu Vesiputous-malli, jonka mukaan projekti on jaettu kuuteen osaan, jotka suoritetaan yksi kerrallaan. Kehitysprojekti Vesiputous-mallissa aloitetaan analyysillä, jossa selvitetään kehitettävän järjestelmän vaatimukset. Suunnitteluvaiheessa suunnitellaan itse kehitettävä järjestelmä ja sen tekninen toteuttaminen sekä projektin eteneminen. Toteutus- ja testausvaiheessa toteutetaan järjestelmä ja testataan sen toiminta. Käyttöönotto- ja ylläpitovaiheessa järjestelmä implementoidaan asiakkaan käyttöön, ja mahdollisesti jatketaan sen ylläpitämistä ja jatkokehittämistä tarvittaessa. (Cohen, Lindvall & Costa, 2003, s. 3, Beck, 1999 mukaan). Jokainen vaihe toteutetaan vain kerran, eikä edelliseen vaiheeseen palata takaisin sen valmistuttua.

Vesiputous- ja muihin perinteisiin järjestelmäkehitysmenetelmiin verrattuna ketterät menetelmät poikkeavat projektin toteuttamistavoissa, eli yksi kerrallaan etenevät työvaiheet on korvattu toistuvilla sykleillä. Cohen, Lindvall ja Costa (2003, s. 12) nostavat ketterien menetelmien tärkeimmiksi piirteiksi mahdollisuuden reagoida järjestelmän muuttuviin vaatimuksiin projektin aikana, sekä jatkuvan yhteistyön asiakkaan kanssa palautteen saamiseksi. Ketterien menetelmien iteratiiviset eli toistuvat työvaiheet mahdollistavat nämä ominaisuudet, sillä yhden työvaiheen toteuttaminen voi sisällyttää perinteisiin menetelmiin verrattuna esimerkiksi suunnittelua, toteutusta ja testaamista (Lapham ym., 2010, s. 6–7). Merkittävimpänä erona perinteisten ja ketterien menetelmien välillä on juuri kehitysprojektin työvaiheiden suorittamisjärjestys: perinteisissä menetelmissä työvaiheet suoritetaan kerran ja ketterissä menetelmissä työvai-

heita toistetaan, kunnes järjestelmä on valmis käyttöönotettavaksi (Palmquist, Lapham, Miller, Chick & Ozkaya 2013, s. 3–14).

2.2 Agile Manifesto

Ketterään järjestelmäkehitykseen on voimakkaasti vaikuttanut helmikuussa vuonna 2001 julkaistu Agile Manifesto, jonka tarkoituksena oli herättää keskustelua järjestelmäkehityksestä, kehitysmetodeista ja organisaatioista ketteryyden kannalta. Käytännössä tarkoituksena oli siis esittää ajatuksia sujuvamman järjestelmäkehityksen näkökulmasta.

Agile Manifesto, tai suomennettuna Ketterä Manifesti sai alkunsa, kun 17 eri ketterien kehitysmenetelmien edustajaa kokoontuivat yhteen pohtimaan ketteryyden merkitystä järjestelmäkehityksessä. Osallistujien tarkoituksena oli edustamiensa menetelmien kautta nostaa esille järjestelmäkehityksen ongelmia ja löytää mahdollisesti ratkaisumahdollisuuksia ja ajatusmalleja ketteryyden näkökulmasta. Tapaamisessa olivat esillä muun muassa Extreme Programming (XP), Scrum, Taipuisa Järjestelmän Kehitysmenetelmä (DSDM), Adaptive Software Development (ASD), Crystal, Feature-Driven Development (FDD) ja Pragmatic Programming, jotka kaikki edustavat ketteriä kehitysmenetelmiä. (Beck ym., 2001).

Ketterä Manifesti, koostuu neljästä ydinajatuksesta, jotka pyrkivät kuvaamaan ketterää kehittämistä kokonaisuutena. Manifesti ja sen neljä ydinajatusta tiivistävät yhteisen pohjan eri ketterille kehitysmenetelmille, kuten myös Scrumille. Ketterän Manifestin mukaan järjestelmäkehitys on ketterää, mikäli siinä on:

- Huomioitu yksilöitä ja vuorovaikutuksellisuutta ennemmin kuin erilaisia kehitysprosesseja ja työvaiheita.
- Pyritty toteuttamaan toimiva järjestelmä sen sijaan, että keskityttäisiin dokumentoimaan kaikki kokonaisvaltaisesti.
- Tehty asiakasyhteistyötä ja huomioitu asiakkaan vaatimuksia kehitykseen läpi projektin, kankeiden sopimusneuvotteluiden sijaan.
- Reagoitu tarvittaessa asiakkaan tai sidosryhmien muuttuviin vaatimuksiin, vaikka muutokset eivät olisi olleet alkuperäisessä suunnitelmassa.

(Beck ym., 2001).

2.3 Scrum

Tutkielma keskittyy tarkastelemaan ketteristä menetelmistä Scrum-menetelmää, jonka tekivät tunnetuksi Ken Schwaber ja Jeff Sutherland 1990-luvulla. Schwaber ja Sutherland saivat inspiraation 30 päivän iteraatio- eli kehitysjaksoihin japanilaisesta kehittämiskulttuurista, muun muassa Hondan, Canonin ja Fujit-

sun käyttämistä ”Shashimi”-menetelmästä. (Larman & Basili, 2003). Scrum on siis yksi ketterän järjestelmäkehityksen viitekehyksistä, joka koostuu kolmesta roolista ja tuotoksesta sekä viidestä eri työvaiheesta (kuvio 1). Viitekehys on luotu erityisesti monimutkaisten projektien toteuttamiseen ja hallintaan (Schwaber & Sutherland, 2012, s. 136). Menetelmä esitellään kaaviona sivulla 13 (kuvio 2).

Scrum-menetelmän tarkoituksena on luoda jokaiselle kehitysprojektille projektin ja kehitysryhmän vaatimuksia vastaavat työskentelytavat. Scrumin, kuten muidenkin ketterien kehitysmenetelmien, tärkeimpänä arvona on myös pyrkiä luomaan kehitysprojektille joustava ja vaatimuksiin mukautuva kehitysympäristö, jota edesauttavat lyhyet kehitysjaksot eli sprintit, joita toistetaan, kunnes kehitettävä järjetelmä on valmis käyttöönotettavaksi. (Srivastava, Bhardwaj & Saraswat, 2017).

Scrum-kehitysryhmään kuuluu kolme eri roolia, jotka jaetaan ryhmän kesken. Nämä roolit ovat Scrum-mestari (Scrum Master), tuotteen omistaja (Product Owner) sekä Scrum-tiimi tai kehitystiimi, johon varsinaiset järjestelmän kehittäjät lasketaan kuuluvaksi. Jokaisella tiimin jäsenellä on oma rooli kehitysprojektissa, mitkä on pyritty valitsemaan jäsenten osaamisen ja henkilökohtaisten ominaisuuksien perusteella, jotta kehitettävä järjestelmä pystyttäisiin toteuttamaan parhaalla mahdollisella osaamisella, täyttäen asiakkaan vaatimuksen lopputuotteen suhteen. (Schwaber & Sutherland, 2012, s. 138). Scrum-mestarin rooli muodostuu kehitystiimin tapaamisten organisoinnista, sprintin tehtävälistan (Sprint Backlog) sekä kehitettävän järjestelmän vaatimusten toteuttamisen seuraamisesta. Scrum-mestarin pääasiallinen rooli kehitystiimissä on siis toimia johtoasemassa projektin organisoinnin ja seuraamisen kannalta, vaikka perusajatuksena Scrum-menetelmässä on pyrkiä kehitysryhmän itseohjautuvuuteen (Rising & Janoff, 2000, s. 31). Tuotteen omistajan tehtävänä on puolestaan päättää kuhunkin sprinttiin tehtävät tehtävälistalta ja arvioida sprintin lopuksi tiimin saavuttamat tulokset. Kehitystiimi toteuttaa järjestelmän, eli suorittaa toteutuksen ja testauksen kunkin sprintin tehtävien mukaisesti. (Schwaber & Sutherland, 2012, s. 138–139).

Scrumin kolme tuotosta ovat puolestaan jokaisen sprintin aikana tapahtunut kehitys eli tuoteversio, tuotteen tehtävälista sekä sprintin tehtävälista. Kehitettävästä järjestelmästä muodostetaan yhdessä asiakkaan tai loppukäyttäjän kanssa tuotteen tehtävälista, eli vaatimukset järjestelmän suhteen. Tuotteen tehtävälista sisältää siis kaikki järjestelmän vaatimukset niin käytettävyyden kuin toiminnallisuuksien kannalta, ja tuotteen tehtävälista jaetaan sprinttikohtaisesti sprintin tehtävälistaksi, eli jokaiselle kehitysjaksolle valitaan osa järjestelmän vaatimuksista kehitettäväksi. Lisäksi sprintin tehtävälista jaetaan kehittäjien kesken, eli jokainen tiimin jäsen, joka osallistuu järjestelmän kehittämiseen, suorittaa jotakin vaatimuksista. Järjestelmän vaatimusten jakamisella niin sprinttien kuin kehittäjienkin kesken pyritään luomaan joustavuutta ja läpinäkyvyyttä kehittämiseen, sillä kun vaatimuksia on paljon myös vaatimusten seuraaminen ja toteuttaminen saavat kriittisen merkityksen. (Schwaber & Sutherland, 2012, s. 147–151).

Scrumin viisi työvaihetta ovat sprintti, sprintin suunnittelupalaveri, päivittäinen Scrum-palaveri, sprintin katselmus ja retrospektiivi. Sprintti eli kehitysjakso on korkeintaan kuukauden mittainen ajanjakso, jonka aikana toteutetaan kaikki Scrumin työvaiheet. Jokaisen sprintin jälkeen on valmiina tuoteversio, jonka kehittämistä jatketaan uudessa sprintissä. Kunkin sprintin aluksi pidetään korkeintaan yhden päivän mittainen sprintin suunnittelupalaveri, jossa tiimi päättää yhdessä sprintin tavoitteet ja nostaa tuotteen tehtävälistalta tavoitteita vastaavat tehtävät sprintin omalle tehtävälistalle. Lisäksi jaetaan tehtäviä kehittäjien kesken, useimmiten osaamisen mukaisesti. Sprintin tehtävälistan avulla saadaan varmistettua, että jokaisen sprintin jälkeen tuoteversio on kehittynyt edellisestä, ja että koko tuotteen tehtävälistalta on toteutettu asiakkaan vaatimuksia vastaavia ominaisuuksia, eli tuoteversio saadaan vastaamaan asiakkaan odotuksia. Kun sprintin suunnittelupalaveri on pidetty ja sprintin tehtävälista määritetty, kehitystiimi aloittaa sprintin varsinaisen toteuttamisen, eli tuoteversion kehittämisen. (Schwaber & Sutherland, 2012, s. 141–147).

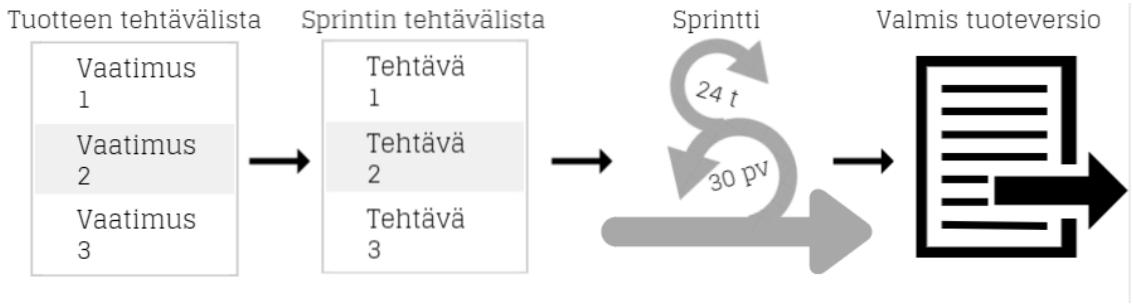
Päivittäinen, noin 15-minuuttia kestävä Scrum-palaveri pidetään kehittäjien kesken, ja palaverin tarkoituksena on käydä läpi edellisen päivän lopputulema ja peilata niitä asetettuihin tavoitteisiin. Tiimin jäsenten tulee vastata seuraaviin kysymyksiin: Mitä tein edellisenä päivänä, mitä aion tehdä seuraavana päivänä, sekä onko edistymiselleni esteitä. Näillä pyritään selvittämään, eteneekö tiimi sprintin tehtävälistan mukaisesti ja reagoimaan mahdollisiin suunnittelemattomiin muutoksiin tai esteisiin. (Schwaber & Sutherland, 2012, s. 145).

Sprintin katselmus pidetään aina sprintin päätyttyä kehitystiimin, Scrummestarin, tuotteen omistajan ja muiden sidosryhmien, esimerkiksi asiakkaan kesken. Sprintin katselmuksessa tarkastellaan sprintin aikana kehitettyä tuoteversiota niin tähänastisen kehityksen kuin lopputuotteenkin kannalta. Katselmuksessa pyritään selvittämään mitä sprintin aikana todellisuudessa on saatu aikaiseksi ja millä tavoin, sekä miten saavutettu tuoteversio peilautuu toivottuun lopputuotteeseen nähden. Lisäksi tarkastellaan tuoteversion ja sprintin tehtävälistan osuvuutta, eli onko tehtävälistan toiminnot ja ominaisuudet saatu toteutettua tuoteversioon halutulla tavalla. Sprintin katselmuksesta saadulla informaatiolla pyritään linjaamaan jo seuraavan sprintin tehtävälistaa ja tuoteversion kehittämistä, sekä luoda edelleen läpinäkyvyyttä ja avoimuutta koko kehitysprojektista kaikille projektin osallisille. (Schwaber & Sutherland, 2012, s. 146).

Viimeinen, eli viides Scrumin työvaiheista on retrospektiivi, eli tilaisuus, jossa kehitystiimi tarkastelee edellistä sprinttiä ja työn suorittamista. Retrospektiivin ensisijainen tarkoitus on kehittää tiimin tekemistä ja työtapoja tulevaa sprinttiä varten, jonka vuoksi retrospektiivissä ei käsitellä tuoteversiota. Käytännössä tiimi tarkastelee esimerkiksi tiimin jäsenten yhteistyötä ja sen mahdollisia kehityskohteita, sprintin tehtävälistan vaatimusten ymmärtämistä ja suorittamista, sekä mitä kehityskohteita työn laadussa voisi olla ja mitä tiimi voisi tehdä paremmin tulevassa sprintissä. Scrum-menetelmään kuuluu olennaisesti järjestelmäkehityksen lisäksi tiimin työtapojen avoin tarkastelu, joten retrospektiivi pyrkii antamaan tiimille näkökulmia tuottavuuden, luovuuden ja tehok-

kuuden parantamiseen tulevia sprinttejä varten. (Schwaber & Sutherland, 2012, s. 147).

KUVIO 1 Scrumin roolit, tuotokset ja työvaiheet (Schwaber & Sutherland, 2012, s. 57)



KUVIO 2 Scrum-menetelmä kaaviona (Schwaber & Sutherland, 2012, s. 158)

3 SCRUMIN HAASTEET JÄRJESTELMÄKEHITYS-PROJEKTEISSA

Tässä tutkielmassa esiteltävät Scrumin haasteet on kerätty kymmenestä tutkimuksesta, joissa on nostettu esille yleisimpiä Scrum-menetelmän haasteita projektinhallinnan, tiimityöskentelyn, tehokkuuden ja vaatimusten kannalta. Tutkimuksissa esiintyneet haasteet on ryhmitelty seitsemäksi osa-alueeksi, jotka ovat kehitystiimi, asiakas, dokumentaatio, työskentely-ympäristö, Scrumin suorittaminen ja tehokkuus, vaatimukset ja järjestelmä, sekä mittaaminen ja arviointi. Tutkimukset, joista haasteet on kerätty ja ryhmitelty ovat Akif ja Majeed (2012), Cho (2008), Edwards (2008), Eloranta, Koskimies, Mikkonen ja Vuorinen (2013), Gold ja Vassell (2015), Hajjdiab ja Taleb (2011), López-Martínez, Juarez-Ramirez, Huertas, Jiménez ja Guerra-Garcia (2016), Santos, Goldman, Shinoda ja Fischer (2011), Therrien ja LeBel (2009), sekä Vijay Anand ja Dinakaran (2015). Haasteet on jaoteltu ryhmiin keräämällä ensin yksitellen kustakin tutkimuksesta keskeisimmät Scrumin ongelmakohdat, ja tämän jälkeen vertaamalla haasteiden osuvuutta joko kehitysprojektiin tai kehitettävään järjestelmään. Suurin osa haasteista peilautui ennemmin kehitysprojektiin kuin pelkkään järjestelmään, joten tutkimuksista nousseet haasteet ryhmiteltiin tämän jälkeen keskeisiin järjestelmäkehitysprojekteihin liittyviin osa-alueisiin. Kuten tutkielman johdannossa on mainittu, osa tutkimuksissa osoitetuista haasteista on karsittu pois, jotta tutkielmassa säilyisi johdonmukainen ja selkeä rakenne. Tällaiset karsitut haasteet esiintyivät esimerkiksi vain yhdessä kymmenestä tutkimuksesta ja eivät tätä kautta antaneet tarpeeksi vahvaa näyttöä Scrum-menetelmän haasteista. Lähdetutkimukset on tarkemmin luokiteltu ja taulukoitu sivulla 33 (liite 1).

3.1 Kehitystiimi

Scrum-menetelmä on ennen kaikkea kehitystiimin viitekehys, jonka kuuluisi selkeyttää ja lisätä joustavuutta tiimin toimintaan. Tutkimusten mukaisesti ke-

hitystiimiin liittyviä haasteita kohdataan viestinnässä tiimin sisällä, jäsenten osaamisessa ja sitoutumisessa projektiin, jäsenten välisessä työnjaossa ja roolituksessa, tiimin itseohjautuvuudessa sekä työn häiriöissä sprintin aikana.

3.1.1 Sisäinen viestintä

Tiimin sisäinen viestintä koetaan haasteeksi, sillä Scrum-menetelmässä korostetaan jatkuvaa kommunikaatiota ja päivittäisen Scrum-palaverin tärkeyden merkitystä, mutta tiimin jäsenten aikataulut voi olla vaikeaa saada täsmäämään päivittäisen palaverin mahdollistamiseksi. Lisäksi osa tiimeistä työskentelee hajautetusti, vaikka Scrumissa pyritäänkin suosimaan yhteistä työskentelyympäristöä, joten etäpalaverien toteuttaminen päivittäin voi myös olla haaste (Cho, 2008). Akif ja Majeed (2012) nostavat lisäksi esille ongelman jatkuvan yhteydenpidon ja viestinnän kuormittavuudesta. Osa kehittäjistä ei koe päivittäisten palaverien merkitystä tärkeänä, jonka vuoksi palaverit saattavat menettää merkitystään projektin edistämisen kannalta, ja niistä saattaa tulla tapahtuma, jossa ei keskustella projektista lainkaan. Tämä voi haitata sprintin tehokkuutta, sillä tiimin mielestä tarpeeton tapaaminen keskeyttää työnteon päivittäin.

Viestinnän merkitys tiimin sisällä korostuu Scrum-menetelmässä, mutta sen tulisi kuitenkin olla vapaaehtoista ja viestinnän arvon muodostua tiimin toimintatapojen mukaisesti. Tärkeintä tiimin sisäisessä viestinnässä onkin oikean ja toimivan viestintätyylin löytäminen kullekin tiimille, sen sijaan, että painotettaisiin ainoastaan jatkuvan viestinnän merkitystä. Kun tiimi löytää itselleen sopivan viestintäkanavan ja tasapainon siihen, millaisista asioista viestitään ja kuinka usein, myös viestinnän väärinymmärrykset vähenevät (López-Martínez, Juarez-Ramirez, Huertas, Jiménez & Guerra-Garcia, 2016). Viestinnän arvoa tiimissä voi myös lisätä tiimin yhteisöllisyys ja henkilökemioiden kohtaaminen, eli tiimin jäsenten kannattaa pyrkiä tutustumaan myös työnteon ulkopuolella (Akif & Majeed, 2012).

3.1.2 Jäsenten osaaminen ja sitoutuminen

Jäsenten osaaminen ja sitoutuminen kehitysprojektiin koetaan haasteeksi Scrumissa, sillä projektit ovat nopeatempoisia ja tiimiltä vaaditaan kykyä itseohjautuvuuteen eli kykyä toimia ilman varsinaista projektipäällikköä. Projektin onnistumisen kannalta on tärkeää, että tiimin jäsenet ovat teknisesti osaavia ja kykeneviä itsenäiseen kehittämiseen, mutta haasteeksi koetaan etenkin motivoituneisuus, sillä kehittäjät voivat toimia yhtä aikaa useissa projekteissa ja tästä syystä menettää motivaatiotaan osaan projekteista (Santos, Goldman, Shinoda & Fischer, 2011). Lisäksi motivaatiota projektia kohtaan voi laskea ketteryyden ymmärtämättömyys, eli tiimin jäsen ei koe Scrumia hyväksi menetelmäksi tai ei ymmärrä mitä lisäarvoa siitä saadaan (Hajjdiab & Taleb, 2011). Tästä syystä olisi erittäin tärkeää, että etenkin tiimin uudet jäsenet tai jäsenet, joiden Scrum-tuntemus ei ole laaja, koulutetaan ja pyritään sulauttamaan tiimiin mahdollisimman aikaisessa vaiheessa, sillä uudet tekijät hidastavat projektin etenemistä

tahtomattaan ja heidän ymmärryksensä Scrum-menetelmästä on suuressa roolissa tehtävistä suoriutumiseen nähden (Therrien & LeBel, 2009).

Tiimin jäsenten motivaation ja sitoutumisen haasteeksi koetaan myös läpinäkyvyyden korostaminen etenkin jäsenten osaamisessa. Kehittäjät voivat tuntea epävarmuutta omasta osaamisestaan muun muassa oman työn haasteiden esilletuomisen vuoksi päivittäisissä Scrum-palavereissa. Tästä voi seurata motivaation laskua Scrumiin menetelmään, esimerkiksi juuri omien taitojen epävarmuuden vuoksi (López-Martínez ym., 2016). Läpinäkyvyyden pelon tuomia haasteita voi pyrkiä ratkaisemaan kattavalla koulutuksella, omalla mentorilla tai ohjaajalla, tai esimerkiksi parityöskentelyllä kokeneemman kehittäjän kanssa (Hajjdiab & Taleb, 2011).

3.1.3 Jäsenten itseohjautuvuus, työnjako ja roolitus

Jäsenten itseohjautuvuus, työnjako ja roolitus on nostettu esille Scrumin haasteeksi, sillä vaikka menetelmä pyrkiikin osallistamaan kaikkia jäseniä, niin sprinttien aluksi tapahtuvaan suunnitteluun voi olla vaikea osallistaa kehittäjiä tai mahdollisia testaajia. Kehittäjät ja testaajat voivat kokea, että heidän työnsä alkaa varsinaisesti vasta myöhemmässä vaiheessa, joten heillä voi olla haasteita osallistua päätöksentekoon heti sprintin alusta (Vijay Anand & Dinakaran, 2015). Myös Hajjdiab ja Taleb (2011) nostavat esille osallistumisen jaettuun päätöksentekoon itseohjautuvuuden näkökulmasta. Tiimin jäsenten roolituksen ei tulisi vaikuttaa päätöksentekoon osallistumiseen, mutta etenkin kehittäjät ja testaajat voivat kokea, että heidän osallistumisensa ei ole tärkeää, joka aiheuttaa haasteita niin päätöksenteon kuin tiimin itseohjautuvuudenkin kannalta, mikäli osallistuminen ei ole tasa-arvoista. Jaetun päätöksenteon ja itseohjautuvuuden lisäämiseksi tiimeissä suositellaan kiinnitettävän huomiota etenkin tasa-arvoisen ja avoimen ilmapiirin ylläpitämiseen päätöksentekotilanteissa. Akif ja Majeed (2012) mainitsevat myös itseohjautuvuuden kannalta tärkeäksi, että tiimi ja jäsenet tietävät oikeutensa osallistua päätöksentekoon tasapuolisesti, mikä on yksi Scrumin ominaisuuksista.

Myös tiimin koko voi vaikuttaa suunnitteluun ja päätöksentekoon. Suurissa tiimeissä voi olla haasteena saada jokaisen kehittäjän ääni kuuluviin, sen lisäksi, että kehittäjillä on erilaisia osaamisalueita. Suuren tiimin tuomia haasteita itseohjautuvuuden kannalta voi olla haastavaa ratkaista, sillä esimerkiksi jäsenten osallistuminen tapahtuviin voi olla vaihtelevaa, mikäli tiimiä ei koeta tiiviiksi ryhmäksi (López-Martínez ym., 2016). Tehokkaan päätöksenteon näkökulmasta kannattaakin suosia 4-6 henkilön tiimejä, sillä jokainen jäsen lisää monimutkaisuutta kommunikaatioon, päätöksentekoon, tehtävien jakoon ja aikataulujen sovittamiseen (Therrien & LeBel, 2009).

Roolituksen haasteeksi koetaan liian suuren roolin ottaminen tiimissä, eli esimerkiksi Scrum-mestarin tai tuotteen omistajan leimautuminen projektipäälliköksi, jonka tehtävänä on seurata ja johtaa projektin edistymistä. Vaarana on myös väärien tehtävien suorittaminen, esimerkiksi tiimin kehittäjät saattavat hallinnoida tuotteen tehtävälistaa (Vijay Anand & Dinakaran, 2015). Yksi mer-

kittävistä rooleihin liittyvistä haasteista tutkimuksissa on etenkin tuotteen omistajan rooliin liittyvät haasteet. Tuotteen omistajan roolin puutteellinen osaaminen tai tuntemus luo haasteita koko projektin edistymiselle, sillä tuotteen omistaja vastaa tuotteen tehtävälistasta ja tätä kautta koko kehitettävän järjestelmän toteutumisesta vaatimustasolla (López-Martínez ym., 2016). Tuotteen omistajan tehtävänä on siis vastata myös järjestelmän laadun maksimoinnista ja tätä kautta tuottojen hankkimisesta yritykselle. Eloranta ym. (2013) mukaan osalla tiimeistä ei ole ollenkaan tuotteen omistajaa tai tuotteen omistajan roolissa toimii asiakkaan edustaja, jolloin kyseinen rooli on täysin väärinymmärretty, sillä tuotteen omistajan tulisi myös kyetä tuottamaan voittoa projektitiimille ja yritykselle, jonka sisällä tiimi mahdollisesti toimii. Roolien ymmärtämiseksi on välttämätöntä, että koko tiimi ja myös asiakas tuntee Scrumin viitekehyksenä, jotta menetelmästä saataisiin hyötyä ja lisäarvoa kehittämiselle (Wan, Zhu, & Zeng, 2013).

Työnjaon ja roolituksen haasteet Scrumissa voi tiivistää López-Martínez ym. (2016) mukaisesti tiimin jäsenten oikeiden osaamisalueiden tunnistamiseen, sillä Scrumin haasteena on usein ajatus siitä, että jäsenillä ei tulisi olla erikoistumisalueita, vaan heidän tulisi olla kaikkien osa-alueiden ammattilaisia. Tämä voi johtaa epärealistisiin odotuksiin jäsenten suoriutumisesta ja lisäksi aiheuttaa haasteita tehtävänjaossa, mikäli jäsenten osaamisalueita ei ole tunnistettu ja tehtäviä ei pystytä osoittamaan parhaalle mahdolliselle kehittäjälle. Lisäksi tarkoituksena olisi, että jokaisen jäsenen vahvuudet voitaisiin sovittaa yhteen tehtävien kanssa, ja mahdollisuuksien mukaan jokainen voisi valita itselleen tehtävät, joista tietää suoriutuvansa (Eloranta, Koskimies, Mikkonen & Vuorinen, 2013). Itseohjautuvassa tiimissä tehtävänjako suositellaan aloitettavaksi tehtävien jaottelulla vaikeustasojen mukaisesti, ja sitten pyrkiä osoittamaan tehtävät osaamisalueiden mukaisesti kullekin jäsenelle (Therrien & LeBel, 2009).

3.1.4 Työn häiriöt sprintin aikana

Viimeisenä kehitystiimiin liittyvänä Scrumin haasteena on esitetty työn häiriöt sprintin aikana, jotka johtuvat yleisemmin joko asiakkaan muuttuvista vaatimuksista tai Scrum-mestarin ja tuotteen omistajan sekaantumisesta kehittämiseen kesken sprintin (Akif & Majeed, 2012). Tiimi voi kokea haasteeksi asiakkaan muuttuvat vaatimukset järjestelmän suhteen, sillä jos kehittämistä on suunniteltu tarkasti eteenpäin ja asiakas ilmoittaa muutoksesta, jonka integroiminen jo kehitettyyn tuoteversioon tulee teknisesti haastavaksi, tiimi joutuu jo valmiiksi nopeatempoisessa kehittämisympäristössä keskeyttämään aloitetun version ja muuttamaan sitä toivotuksi. Tämä aiheuttaa suunnittelemattomia häiriöitä työhön, mikä kuormittaa tiimin jäseniä (Akif & Majeed, 2012). Scrum painottaakin etenkin ulkopuolisilta häiriöiltä suojautumista sprinttien aikana, jonka vuoksi etenkin asiakkaalta tulevat vaatimukset olisivat tärkeää saada selkeästi tuotteen ja sprintin tehtävälistalle ennen kehittämisen aloittamista. Toisaalta asiakkaan ja tiimin välinen jatkuva viestintä kuuluu Scrum-menetelmän perusajatuksiin, jolloin asiakkaalle pyritään antamaan myös mahdollisuus ke-

hittämisen seuraamiseen ja vaatimusten lisäämiseen tai muuttamiseen (Eloranta ym., 2013).

Lisäksi tiimi kokee kehittämisen ja työnteon sprintin aikana häirityksi, mikäli Scrum-mestari tai tuotteen omistaja kyselevät kesken sprintin kehitteillä olevasta tuoteversiosta, sillä aikaa kehittämiselle on niukasti. Yleisimmin Scrum-mestarin tai tuotteen omistajan puuttuminen kehittämiseen kesken sprintin johtuu tiimin kypsyydestä, eli voidaan kokea, että tiimi ei ole tarpeeksi kypsä toteuttamaan sprinttiä tai Scrumia kokonaisuudessaan ilman, että joudutaan välissä tarkastelemaan mitä tiimi on saanut aikaiseksi. Haastetta suositellaan ratkaistavaksi tiimin kouluttamisella Scrumiin, jotta voidaan varmistua siitä, että tiimi ymmärtää viitekehyksen periaatteet ja kykenee itseohjautuvaan kehittämiseen (Akif & Majeed, 2012).

3.2 Asiakas

Asiakkaaseen liittyviä haasteita Scrum-menetelmässä on esitelty Chon (2008) tutkimuksessa, jossa merkittävimmiksi haasteiksi on nostettu asiakkaan osallistuminen kehitysprojektiin sekä asiakkaan vaatimusmäärittely kehitettävän järjestelmän kannalta. Asiakkaan tulisi olla osallisena kehitysprojektiin alusta loppuun saakka, jotta jokaisen sprintin jälkeen julkaistava tuoteversio saisi palautetta ja versiota olisi mahdollista kehittää toivottuun suuntaan, mikäli siinä ilmenee asiakkaan puolelta tarvetta vaatimusten muuttamiselle. Asiakkaan osallistuminen koetaan kuitenkin Chon (2008) mukaan haasteeksi, sillä asiakkaat ovat usein kiireisiä, eivätkä ehdi osallistua esimerkiksi sprintin katselmukseen, joka on tarkoitettu ensisijaisesti tapahtumaksi, jossa asiakas pääsee osallistumaan tuoteversion tarkasteluun ja palautteenantoon.

Lisäksi asiakkaan vaatimusmäärittely kehitettävän järjestelmän kannalta voidaan kokea haasteeksi kehitystiimeissä, sillä asiakas ymmärtää usein ensisijaisesti vain oman liiketoiminnan arvon, eikä järjestelmäkehityksen prosesseja ja tekniikoita. Tämä voi aiheuttaa vaatimusmäärittelyssä muun muassa epäselviä vaatimuksia ja puuttuvaa informaatiota siitä näkökulmasta, mihin mitäkin asiakkaan toivomaa toiminnallisuutta käytetään. Kehitystiimit voivat kokea haasteellisena asiakkaan vaatimusten tulkitsemisen järjestelmäkehityksen kannalta, sillä asiakkaan vaatimuksia voidaan joutua muokkaamaan järjestelmäkehityksen sekä järjestelmän teknisen toteuttamisen vuoksi. (Cho, 2008). Haasteita tuottaa myös kehittäjien liiketoiminnallinen ymmärrys, eli kehittäjät tai tiimin jäsenet eivät tunne asiakkaan liiketoimintaa tai liiketoiminnan arvoa. Asiakas painottaa vaatimusmäärittelyssä asioita, joilla liiketoiminnan arvo pyritään maksimoimaan valmiin järjestelmän osalta, mutta kehittäjät eivät välttämättä kykene peilaamaan asiakkaan toivomia vaatimuksia ja toiminnallisuuksia yhteen teknisten ratkaisujen kanssa. (López-Martínez ym., 2016). Myös vaatimusten muokkaamiseksi kehitystiimi joutuu usein tarkentamaan asiakkaalta vaatimuksia, joka koetaan haasteeksi, sillä epäselvät tai puutteelliset vaatimukset hidastavat sprinttien ja kehittämisen etenemistä.

Kattavien ja konkreettisten vaatimusten saamiseksi sekä niiden muokkaaminen käyttäjätarinoiksi vaatii sujuvaa asiakasyhteistyötä, joka Eloranta ym. (2013) mukaan taataan parhaiten tapaamalla asiakkaan kanssa kasvotusten. Vaatimusten toimittaminen ainoastaan kirjallisessa muodossa voi aiheuttaa väärinymmärryksiä niiden tulkitsemisessa, joten varsinkin ennen kehittämisen aloittamista tulisi tavata asiakas kasvokkain. Lisäksi on suositeltu, että myös asiakas esittelisi omaa liiketoimintaansa kehitystiimille, jotta myös tiimi ymmärtäisi mitä käytännön etuja järjestelmällä pyritään saavuttamaan (Hajjdiab & Taleb, 2011).

3.3 Dokumentaatio

Dokumentaatioon liittyviä haasteita Scrumissa ovat dokumentaation merkityksen vähentyminen verrattuna perinteisiin järjestelmäkehitysmenetelmiin, sekä vaatimusmäärittelyn ja dokumentaation yhteistoimivuus. Scrum-menetelmässä painotetaan tarkan ja kokonaisvaltaisen dokumentaation vähentämistä, sillä kattavan dokumentaation tuottamisen uskotaan vievän aikaa järjestelmän kehittämiseltä ja testaamiselta (Beck ym., 2001).

Cho (2008) esittelee kehitystiimien haasteeksi dokumentaation merkityksen vähentymisen, sillä tiimin jäsenet voivat kokea, että muiden kehittäjien tekemän koodin etenemistä ja muutoksia on vaikea seurata, jos muutoksia ei ole dokumentoitu kaikille jäsenille. Akif ja Majeed (2012) esittävät, että kehitystiimit pitävät haasteena erityisesti koodiin tehtyjä suunnittelemattomia muutoksia, jotka aiheutuvat viestinnän hajautumisesta ja yhteisen dokumentaation puuttumisesta. Tiimien sisäinen viestintä voi olla hajautettu useaan eri viestintäpalveluun, kuten sähköpostiin ja pikaviestimiin, joiden välillä tiimin eri henkilöt viestivät joko tiimin tai muutaman henkilön kesken. Hajautettu viestintä aiheuttaa haasteita tiimeissä Akif ja Majeed (2012) mukaan etenkin niissä tapauksissa, kun kehitettävän järjestelmän vaatimuksia muokataan muualla kuin sprintin suunnittelupalaverissa, sillä tarkan dokumentaation puuttuessa tämä voi aiheuttaa sovitun muutoksen päivittämisen unohtamista tuotteen tehtävälistalle.

Lisäksi kehitystiimit voivat kokea haasteena dokumentaation ja vaatimusmäärittelyn kohtaamisen, sillä jos vaatimusmäärittely ei ole kattava tai sitä voi tulkita eri tavoin, se voi aiheuttaa ongelmia koodiin tehtäviin muutoksiin, mikäli kehittäjillä on eri näkemyksiä vaatimuksista. Jos muutosta ei ole dokumentoitu tai perusteltu miksi muutos on toteutettu näin, kehitystiimin sisällä voi herätä epätietoisuutta ja ristiriitoja vaatimusten näkökulmasta. (Cho, 2008).

Dokumentaatioon liittyviä haasteita voi pyrkiä ehkäisemään pitämällä tuotteen tehtävälistan aina ajan tasalla sekä näkyvillä kaikille projektin osallisille. Jos esimerkiksi vaatimuksiin tehdään muutoksia niin, että kaikki eivät ole paikalla päättämässä muutoksesta, muutos tulisi kirjata välittömästi koko tiimin näkyville tehtävälistaan. (Hajjdiab & Taleb, 2011).

3.4 Työskentely-ympäristö

Kehitystiimin työskentely-ympäristöön liittyviä haasteita Scrum-menetelmässä on projektin paikallisuudesta riippuen joko avoimen tilan tai hajautetun tiimin tuomat haasteet. Scrum-menetelmä sisältää ajatuksen kehitystiimin paikallisuudesta, eli ajatuksena on, että tiimi työskentelisi samassa tilassa parhaan mahdollisen läpinäkyvyyden ja kommunikaation saavuttamiseksi (Faniran, Badru & Ajayi, 2017). Avoin tila, kuten avotoimisto voi kuitenkin olla haaste, sillä jatkuvasti tapahtuva kommunikaatio, kysymykset ja melu aiheuttavat rauhattomuutta ja keskittymisvaikeuksia. Jatkuvasti keskeytyvä työnteko voi myös pitkällä aikavälillä aiheuttaa tuottavuuden heikentymistä sprinttikohtaisesti. (Cho, 2008).

Hajautetulla tiimillä tarkoitetaan esimerkiksi kehitystiimiä, joka toimii osastoittain tai kokonaan maantieteellisesti eri sijainneissa (Lanubile, 2006). Hajautettujen kehitystiimien haasteiksi Scrumissa on koettu kommunikaatioongelmat (Akif & Majeed, 2012), sekä sprintin suunnittelu ja analyysi jaetun päätöksenteon avulla (Vijay Anand & Dinakaran, 2015). Kommunikaatioongelmat hajautetuissa työskentely-ympäristöissä voivat aiheutua muun muassa eriävistä työskentelyajoista, kulttuurisista eroista tai sopivan viestintävälineen puuttumisesta (Ghosh, 2012).

Sprintin suunnittelun ja analyysin ongelmat voivat olla seurausta tiimin kommunikaation ongelmista, sillä esimerkiksi etäpalaverien ajankohdan sopiminen on koettu haasteeksi kehitystiimeissä, jotka työskentelevät hajautetusti. Mikäli tiimin jäsenet eivät pysty osallistumaan yhteisiin Scrum-tapahtumiin etänä, kuten videoyhteyksin, on myös vaarana, että jaettu päätöksenteko epäonnistuu ja päätöksentekovalta keskittyy vain osalle jäsenistä (Vijay Anand & Dinakaran, 2015). Hajautetun tiimin viestintää voi pyrkiä tukemaan sopimalla projektin alussa yhteisen viestintäkanavan (esimerkiksi jokin pikaviestin) (Akif & Majeed, 2012) tai järjestämällä esimerkiksi kerran kahdessa viikossa kasvokkaistapaamisen tiimin jäsenten kesken, jossa käydään läpi projektin edistymistä (Hajjdiab & Taleb, 2011).

3.5 Scrumin suorittaminen ja tehokkuus

Ketterissä menetelmissä ja etenkin Scrumissa on tarkoitus pitää kehitysjaksot lyhyinä, jotta tehokkuus ja tuottavuus säilyisivät. Scrumin tehokkuuteen liittyvistä haasteista nousi lähdetutkimuksissa etenkin ilmi Scrum-tapahtumien ajoittaminen ja niissä toimiminen, ja kuinka kokonaisvaltaisesti tiimi on ymmärtänyt Scrum-menetelmän käytännössä.

3.5.1 Tiimin kypsyys ja Scrumin ymmärtäminen käytännössä

Scrumin tehokkuuden haasteiksi kehitystiimit ovat kokeneet tiimin kypsyyden Scrumin toteuttamiseen, sillä jos tiimin jäsenten taitotaso Scrum-projekteissa toimimiseen vaihtelee ensikertalaisesta ammattilaiseksi, on vaarana, että tehokkuus kärsii (Akif & Majeed, 2012). Osaavien henkilöiden rekrytoiminen projekteihin on kuitenkin haasteellista, sillä Scrum ei anna valmiiksi rekrytointimallia esimerkiksi sen suhteen, millainen osaaminen ja koulutustausta jäsenillä kuuluisi olla. Tärkeintä on kuitenkin pyrkiä hankkimaan teknisesti osaavien henkilöiden lisäksi henkilöitä, jotka ymmärtävät ketteryyden ja sen tuoman lisäarvon kokonaisuudessaan eivätkä pelkästään suorita menetelmää (López-Martínez ym., 2016).

Myös Santos ym. (2011) nostavat esille haasteen Scrumin ymmärtämisestä, eli useilla tiimeillä voi esiintyä ongelmia menetelmän toteuttamisessa niissä tilanteissa, kun tiimi luulee toimivansa Scrumin mukaisesti, mutta todellisuudessa kehitysprojekti noudattaakin jotakin muuta kehittämismenetelmää. Näissä tilanteissa haasteena on usein Scrumin todellisen lisäarvon tuominen kehitysprojektiin sekä yritykselle, eli tiimit saattavat olettaa Scrumin olevan kyseiseen projektiin sopiva viitekehys, mutta sen hyötyjä ei ole kunnolla arvioitu ja kokematon Scrum-tiimi voi päätyä toteuttamaan menetelmää vain, koska se on trendikästä tai sen oletetaan tekevän projektista tuottavan ja tehokkaan. (Santos ym., 2011).

3.5.2 Scrum-tapahtumien ajoittaminen ja tapahtumissa toimiminen

Scrumin ymmärtäminen projektiviitekehyksenä antaa pohjan myös Scrum-tapahtumien tehokkuudelle. Kuten alaluvussa 3.1 Kehitystiimi on mainittu, Scrum-tapahtumien ajoittaminen voi olla haaste tiimin jäsenten omien aikataulujen vuoksi, mutta myös sprinttien aikataulutus koetaan haasteeksi tehokkuuden kannalta (Eloranta ym., 2013). Scrum ei tarjoa valmiiksi määriteltynä optimaalisinta sprintin pituutta, mutta tavallisimmin sprintin kesto on noin neljä viikkoa. Neljän viikon mittaisten sprinttien haasteena on kuitenkin niin sanotusti aikataulullisen paineen puuttuminen kehittämisestä, eli sprintin ollessa aikataulujen osalta kevyempi, eli ei niin nopeatempoinen, kehittämisessä ei päästä parhaaseen mahdolliseen tulokseen verrattuna lyhyempiin ja nopeatempoisempiin sprintteihin. Kuitenkin myös erittäin lyhyissä sprinteissä (1–1,5 viikkoa) tehokkuus kärsii, sillä tällöin esimerkiksi päivittäiset Scrum-palaverit vievät suhteessa enemmän aikaa koko sprintin aikataulusta ja lyhentävät kehittämiselle jäävää aikaa. Optimaalisinta olisi valita sprintin kestoksi jotakin tältä väliltä, jotta aikaa itse kehittämiselle olisi riittävästi, mutta ei niin, että sitä jäisi jokaisessa sprintissä käyttämättä. Sprintin kesto suositellaan arvioitavan sen mukaan, paljonko tuotteen tehtävälistalta on minimissään saatava suoritettua sprintin aikana. Tehtävät ja vaatimukset kannattaa lisäksi pyrkiä arvioimaan haastavuuden lisäksi myös ajallisesti, eli kuinka kauan kunkin tehtävän suorit-

tamiseen menee. Tätä kautta pystytään valitsemaan sopiva määrä tehtäviä kuhunkin sprinttiin, sekä ajoittaa sprintti oikean mittaiseksi. (Eloranta ym., 2013).

Aikataulutuksen lisäksi tapahtumiin liittyvä haaste on Chon (2008) mukaan oikeiden asioiden tekeminen oikeissa tapahtumissa, sillä Scrumiin kuuluu olennaisena osana ajatus eri työvaiheista ja tapahtumista, joihin jokaiseen on priorisoitu tietyt käsiteltävät asiat. Yksi suurimmista haasteista tähän liittyen Vijay Anand & Dinakaran (2015) mukaan on Scrum-palaverit, joissa nopeaksi tarkoitetun välikatsauksen sijaan tiimi aloittaakin ratkaisemaan kohtaamiaan ongelmia. Tämä pitkittää 15-minuuttiseksi tarkoitettua palaveria ja saattaa heikentää sprintin tehokkuutta. Scrumin kokonaisvaltainen ymmärtäminen ja tehokkuus voivat heijastua toisiinsa myös tilanteissa, joissa kehittäjät kokevat tekevänsä itsenäistä työtä saadessaan omat tehtävät sprintin tehtävälistalta, ja käyttävät sprintin aluksi paljon aikaa tehtäviensä suunnitteluun. Tästä voi aiheutua tehokkuuden eriarvoistumista jäsenten kesken, sillä osa jäsenistä saattaa olla pidemmällä omien tehtävien suorittamisessa sprintin edetessä loppua kohti kuin ne, jotka ovat sprintin aluksi käyttäneet aikaa työnsä tarkkaan suunnitteluun. Pääpaino sprintin aikana tulisi olla kehittämisessä ja suunnittelu tehtävien osalta toteuttaa sprintin suunnittelupalaverissa, jotta sprintit ja kehittäminen sujuisivat mahdollisimman suoraviivaisesti (López-Martínez ym., 2016).

Scrumiin viitekehyksenä liittyy myös haasteita järjestelmäkehitysprojekteissa etenkin organisaation oppimisen (Hajjdiab & Taleb, 2011), sprinttien muokattavuuden (Akif & Majeed, 2011) sekä valmiin määritelmän (’Definition of Done’) (Vijay Anand & Dinakaran, 2015) kanssa. Organisaation oppiminen Scrum-menetelmän kautta voi olla haaste tiimeille, koska tiimit keskittyvät usein toteuttamaan Scrumia viitekehyksenä ja keskittyvät tarkasti Scrumin prosesseihin, mutta unohtavat organisaation oppimisnäkökulman, eli eivät esimerkiksi pyri tietoisesti kehittämään retrospektiivissä nousseita tiimin tai projektin kehitys- ja ongelmakohteita. Organisaation oppiminen ja sprinttien muokattavuus liittyvät haasteina toisiinsa, sillä Scrumin tarkoituksena on tarjota tiimeille malli ketterästä kehittämisestä, mikä on kuitenkin muokattavissa tiimin ja projektin omiin tarpeisiin. Tiimit saattavat kuitenkin kokea haasteeksi Scrumin muokattavuuden ja täyden tehokkuuden saavuttamisen, etenkin mikäli tiimin jäsenillä on eriäviä näkemyksiä Scrumin toteuttamisesta. Scrumin toteuttamisen eriävät näkökannat jäsenten keskuudessa voivat vääristää tiimin jäsenten suhtautumista itseohjautuvuuteen ja ketteryyteen, luomalla ilmapiirin, jossa ketteryys tarkoittaa kevyempää projektinhallintamenetelmää, jossa sovituilla aikatauluilla tai tapaamisilla ei ole suurta merkitystä. Tämä tuo haasteita projektin tehokkuuteen, mikäli jäsenet eivät onnistu muokkaamaan Scrumista sopivaa menetelmää oman projektin kannalta (López-Martínez ym., 2016). Näiden lisäksi valmiin määritelmä, eli tiimin yhteinen käsitys valmiista tuoteversiosta (Davis, 2013), aiheuttaa haasteita tiimeissä, mikäli tuoteversio saadaan aiempaa suunniteltua täyttämään sen sprintille asetetut vaatimukset. Tällöin tiimin vaarana on sortua testaamaan versiota tarpeettomasti useita kertoja tai lisäämään sprintille suunnittelemattomia vaatimuksia, vaikka sprintin voisi esimerkiksi päättää ja version esitellä asiakkaalle.

3.6 Vaatimukset ja järjestelmä

Vaatimuksiin ja kehitettävään järjestelmään liittyviä haasteita ovat lähdetutkimusten mukaisesti tuotteen tehtävälistan hallinta, sekä tuoteversion laatu ja testaaminen.

3.6.1 Tuotteen tehtävälistan hallinta

Tuotteen tehtävälistan hallintaan liittyvät haasteet jakautuvat kahteen osaan: käyttäjätarinoiden muuttamiseen tehtäviksi sekä abstraktien vaatimusten ymmärtämiseen (Hajjdiab & Taleb, 2011). Käyttäjätarinat ovat kuvauksia järjestelmän ominaisuuksista käyttäjän näkökulmasta, eli kuka käyttää järjestelmää ja mitä järjestelmän käytöllä pyritään saavuttamaan. Käyttäjätarinoiden avulla pyritään luomaan kokonaisvaltainen kuva ominaisuuksista ja toiminnallisuuksista, joita kehitettävältä järjestelmältä vaaditaan, nimenomaan järjestelmän käyttäjän näkökulmasta. (Zeaaraoui, Bougroun, Belkasmi & Bouchentouf, 2013).

Käyttäjätarinat toimivat pohjana tuotteen tehtävälistan laadinnassa, eli käyttäjätarinoista tulisi poimia oikeat asiat, jotta ne voidaan muuttaa tehtäviksi ja lisätä sprinttien tehtävälistalle. Virheellisesti tulkitut käyttäjätarinat ja niistä muodostetut tehtävät johtavat projektia väärään suuntaan ja aiheuttavat ongelmia aikatauluun. (Hajjdiab & Taleb, 2011). Haasteena tässä on erityisesti koko projektin laadullinen onnistuminen. Mikäli tuotteen tehtävälista muodostetaan väärien vaatimusten perusteella, tuoteversiot ja lopputuote eivät myöskään vastaa asiakkaan toivomuksia. Useimmiten tällaisiin tilanteisiin johtaa asiakkaan ja kehittäjien eroavat näkemykset kehitysprojektista: asiakas haluaa saavuttaa järjestelmäkehityksen avulla arvoa liiketoiminnalleen, mutta kehittäjät näkevät järjestelmän vain tuoteversioiden ja kehitysprojektin kautta (Vijay Anand & Dinakaran, 2015).

On myös mahdollista, että asiakkaan toimittamat vaatimukset, joiden pohjalta käyttäjätarinat on luotu, on lähtöisin taholta, joka ei tunne liiketoimintaa kattavasti. Tämä voi aiheuttaa käyttäjätarinoihin ja sitä kautta koko tuotteen tehtävälistalle puutteellisia vaatimuksia, jotka puolestaan muodostavat valmiin tuotteen kokonaisuudessaan. Tehtävälistan hallintaan kuuluukin olennaisesti vaatimusten tarkkailu ja niiden päivittäminen asiakkaan toiveiden mukaisesti (Edwards, 2008). Käyttäjätarinoiden merkitys tuotteen tehtävälistan luomisessa on Scrumissa kriittinen, sillä Scrum painottaa kattavan vaatimusmäärittelyn merkitystä koko projektin laadullisessa onnistumisessa. Asiakkaalta tulisi aina pyrkiä saamaan pelkän vaatimusluettelon sijaan monipuoliset käyttäjätarinat, joissa toimintojen todellista käyttötarkoitusta on pohjustettu ja perusteltu (Eloranta ym., 2013).

3.6.2 Tuoteversion laatu ja testaaminen

Tuoteversion laatuun ja testaukseen vaikuttavia haasteita Scrumissa ovat ajallisesti lyhyet ja nopeatempoiset sprintit sekä järjestelmän kokonaisuudenhallinta. Tiimit ovat kokeneet haasteena etenkin hyvälaatuisen tuoteversion tuottamisen sprinttien ajallisen rajallisuuden vuoksi, koska jokaisen sprintin lopputuloksena tulisi olla aina edelliseen tuoteversioon verrattuna kokonaisuudeltaan laajempi versio. Tämä aiheuttaa haasteita etenkin koodin laatuun, sillä paineen alla tuotettu koodi voi sisältää virheitä, jotka vaikuttavat jatkossa koko järjestelmän toimivuuteen. Virheellisen koodin lisäksi myös version testaaminen koetaan haastavaksi, johtuen ajatuksesta, että jokaisen sprintin päätteeksi pitäisi olla jotakin valmista, mitä esitellä asiakkaalle (Akif & Majeed, 2012). Nopeasta työtahdista huolimatta tiimissä tulisi aina painottaa puhtaan ja toimivan koodin merkitystä, sillä koodin korjaaminen tulevassa sprintissä vie aikaa varsinaiselta kehittämiseltä (Therrien & LeBel, 2009).

Mikäli testauksen yhteydessä esiintyy virheitä, joita ei saada ratkaistua myös järjestelmän versioiden kokonaisuudenhallinta muodostuu haasteelliseksi (Akif & Majeed, 2012). Myös Vijay Anand ja Dinakaran (2015) esittelevät testauksen ja kokonaisuudenhallinnan yhteyttä, ja heidän mukaansa tiimit kokevat testauksen sitä haastavammaksi mitä pidemmälle kehitysprojektissa on edetty. Tämä johtuu tuoteversion laajuudesta ja toiminnallisuuksien yhteensovittamisesta, eli haasteena on saada kuluvan sprintin tehtävät ja vaatimukset sopimaan jo aiemmin kehitettyyn versioon. Myös paineensietokyky kokonaisuudenhallintaan liittyen voi olla Scrum-tiimien haaste, etenkin jos jossakin sprintissä todetaan, että uudet toiminnallisuudet eivät sovi yhteen aiemman kehityksen kanssa. Testauksen ja järjestelmän ominaisuuksien yhteentoimivuuden kannalta testaus tulisi aina suorittaa ympäristössä, joka jäljittelee mahdollisimman tarkasti tulevaa järjestelmän käyttöympäristöä ja siihen kohdistuvia rasitteita. Tällöin saadaan paras kuva järjestelmän toimivuudesta kokonaisuudessaan. (Therrien & LeBel, 2009). Toisaalta tiimeillä voi olla haasteena myös testaamisen ajoittaminen, sillä useat tiimit saattavat jättää testauksen viimeiseksi tehtäväksi sprintin aikana, vaikka koodia kannattaisi testata aina sen kehittämisen yhteydessä. Testaus tulisi kuitenkin aina suorittaa jokaisessa sprintissä, jotta uutta koodia ei kirjoiteta edellisessä sprintissä luodun testaamattoman koodin päälle. (Eloranta ym., 2013).

Kaiken kaikkiaan uuden järjestelmän kehittämistä ei kannata aloittaa, ennen kuin tiimillä on tarpeeksi laaja käsitys kehitettävästä järjestelmästä vaatimusten osalta, sekä aikeena julkaista myös valmis tuote asiakkaalle. Tiimin tehtävänä on selvittää asiakkaan todellinen tarve järjestelmälle ja aloittaa kehitysprojekti vasta, kun asiakkaalta on saatu tarpeeksi kattavat vaatimukset käyttäjätarinoiden ja sitä kautta tuotteen tehtävälistan muodostamiseen. (Therrien & LeBel, 2009).

3.7 Mittaaminen ja arviointi

Viimeisenä lähdetutkimuksissa nousseista Scrumin haasteista on kehitysprojektin edistymisen ja kokonaisuuden mittaaminen ja arviointi. Akif ja Majeed (2012) esittelevät tutkimuksessaan kaksi haastetta liittyen Scrum-projektien seurantaan: kehitettävän järjestelmän laadullinen mittaaminen, sekä projektin riskinhallinta. Scrumin haasteena tuoteversioiden ja lopulta valmiin järjestelmän mittaamisessa ja arvioimisessa on ennen kaikkea laatua arvioivien työkalujen puuttuminen. Scrum ei tarjoa kehitystiimeille valmista mallia laadulliseen arviointiin, vaan tiimien tulee itse valita sopiva menetelmä laadun mittaamiseen ja analysoimiseen.

Yleisesti Scrumin yhteydessä käytössä oleva, järjestelmän edistymistä mittaava menetelmä on niin sanottu tuotteen edistymiskäyrä. Tuotteen edistymiskäyrä on projektin etenemisestä visuaaliseen muotoon piirretty kuvio, jossa on oma käyrä projektin suunnitellulle sekä todelliselle edistymiselle. Edistymiskäyrän avulla pyritään tarkastelemaan projektin kokonaisvaltaista edistymistä suunniteltuun nähden, ja puuttumaan mahdollisiin poikkeuksiin käytännöntasolla. (Woodward, Cain, Pace, Jones & Kupper, 2013). Edistymiskäyrän haasteena mittaustyökaluna on kuitenkin sen suppeus, eli se tarjoaa vain aikataulullisen seurantamahdollisuuden projektin edistymisestä. Mikäli edistymiskäyrää kuitenkin käytetään projektin seurantatyökaluna, sen tulisi olla aina näkyvillä kaikille tiimin jäsenille, eikä pelkästään esimerkiksi Scrum-mestarille (Eloranta ym., 2013).

Laajempien projektin laadun mittaamis- ja arviointivälineiden löytäminen tai kehittäminen jää tiimien vastuulle (Eloranta ym., 2013). Vaikka laadun ja tiimin suorituskyvyn arviointiin ei tarjota Scrumin puitteissa valmista ratkaisua, projektin edistymisen ja laadullisen onnistumisen seuraamiseen vaaditaan tiimiltä avoimuutta, oman työn tarkastelua ja sopeutumiskykyä (López-Martínez, 2016). Scrum-tapahtumat, esimerkiksi retrospektiivi on tarkoitettu työn tarkasteluun, joten avoimella keskustelulla tiimin on kuitenkin mahdollista pyrkiä arvioimaan työmenetelmien ja kuluneen sprintin onnistumista kokonaisuudessaan (Schwaber & Sutherland, 2012).

Kuten mainittu, Scrum ei myöskään ohjaa tiimiä erityisen tarkasti projektin riskinhallinnan kanssa, voi pahimmassa tapauksessa johtaa koko projektin epäonnistumiseen. Yleisimpiä haasteita riskinhallinnan näkökulmasta ovat riskien tunnistaminen ja oikeiden ratkaisumallien sovittaminen riskeihin (Gold & Vassell, 2015). Projektin riskien tunnistaminen juuri oman projektin kannalta on usein haasteellista, sillä tunnetut riskit, kuten aikataulut tai viestintä asiakkaan kanssa koetaan itsestäänselvyyksiksi, mutta tiimi ei ole tarpeeksi kypsä tarkastelemaan tietoisesti omaan projektiin kohdistuvia riskejä. Riskit tulisi siis pyrkiä tunnistamaan ennen kehittämisen aloittamista, sillä projektit sisältävät aina riskejä. Riskien tunnistaminen ja analysointi auttaa myös riskien hallinnassa ratkaisumahdollisuuksien kannalta, sillä kattavassa riskien analysoinnissa tiimi

selvittää myös tavat, joilla riskien toteutuessa reagoidaan. (Larson & Gray, 2015, s. 310).

4 YHTEENVETO JA POHDINTA

Scrum on vakiinnuttanut suosionsa ketteränä kehitysmenetelmänä 2000-luvun aikana niin pienissä kuin laajemmissa ja hajautetuissakin järjestelmäkehitysprojekteissa. Scrumin hyödyiksi koetaan etenkin läpinäkyvyys, joustavuus, tehokkuus ja reagointialttius, jotka ovat perinteisiin järjestelmäkehitysmenetelmiin verrattuna tuottaneet nykyaikaiseen järjestelmäkehitykseen lisäarvoa lopputuotteiden laadullisesta näkökulmasta. Lisäksi Scrumin koetaan selkeyttävän projektinhallintaa lisäämällä kommunikaation määrää ja merkitystä sekä kehitystiimin sisällä että asiakkaan kanssa.

Scrumiin kuten muihinkin projektinhallinta- ja järjestelmäkehitysviitekehyksiin liittyy haasteita, jotka tässä tutkielmassa on jaoteltu seitsemään osa-alueeseen: kehitystiimi, asiakas, dokumentaatio, työskentely-ympäristö, Scrumin suorittaminen ja tehokkuus, vaatimukset ja järjestelmä, sekä mittaaminen ja arviointi. Kehitystiimiin liittyviksi haasteiksi on lähdetutkimusten mukaan koettu tiimin sisäinen viestintä etenkin jatkuvan kommunikaation näkökulmasta. Jatkuvasti tapahtuva viestintä voi projektin edistyessä menettää merkitystään ja tuntua jäsenistä liialliselta. Tämä nivoutuu myös yhteen projektiin sitoutumisessa, eli jäsenet voivat sitoutua eri intensiteetillä projektiin, mikä aiheuttaa haasteita etenkin motivaationäkökulmasta. Sprintin tehtävien työnjako sekä tiimin itseohjautuvuus Scrumin haasteina edustavat hyvin haastetta liittyen Scrumin ymmärtämiseen viitekehyksenä. Jos menetelmä ei ole tunnettu tiimin jäsenten keskuudessa, voidaan mahdolliset hyödyt ja lisäarvo jättää saavuttamatta projektinhallinnallisesta näkökulmasta. Yhteen vetäen voi sanoa, että kehitystiimiin liittyvät haasteet Scrumissa liittyvät pitkälti juuri menetelmän kokonaisvaltaiseen ymmärtämiseen. Tästä voi myös päätellä, että Scrum pitäisi menetelmänä olla tuttu tiimin jäsenille ennen projektin ja kehittämisen aloittamista, tai vaihtoehtoisesti järjestää jäsenille kattava Scrum-koulutus, jossa pyritään myös muokkaamaan menetelmästä sopiva tulevaa projektia ja juuri omaa tiimiä varten.

Asiakkaaseen liittyvät haasteet käsittelevät ennen kaikkea asiakkaan osallistumista projektiin viestinnän näkökulmasta, sekä asiakkaalta tulevan vaatimusmäärittelyn kattavuutta. Jatkuvan viestinnän tulisi kehitystiimin lisäksi

koskea myös viestintää asiakkaan kanssa, mutta haasteita aiheuttaa esimerkiksi asiakkaiden kiireisyys, jonka vuoksi sprintin katselmukseen osallistuminen epäonnistuu. Asiakkaan toimittamat vaatimukset kehitettävästä järjestelmästä ovat lisäksi tiimien ja kehittäjien kokema haaste, sillä asiakas ymmärtää usein vain oman liiketoimintansa arvon, eikä järjestelmäkehityksen periaatteita tai teknisiä ratkaisuja. Asiakkaaseen liittyvät haasteet johtuvat osittain kehitystiimin haasteiden mukaisesti Scrum-menetelmän heikkoon tuntemukseen. Tiimin lisäksi myös asiakkaalle tulisi painottaa menetelmän lisäarvoa tuottavia periaatteita, kuten juuri jatkuvan viestinnän merkitystä lopputuotteen laadullisen arvon maksimoimiseksi. Vaatimusten osalta asiakkaan kanssa tulisi aina järjestää tapaaminen, jossa vaatimuksia käydään läpi keskustellen, sen sijaan, että asiakas toimittaa esimerkiksi kirjallisen listan tiimille.

Dokumentaation haasteita Scrumissa ovat dokumentaation merkityksen väheneminen, sekä vaatimusmäärittelun ja dokumentaation yhteistoimivuus. Haasteet johtuvat lähdetutkimusten mukaan ennen kaikkea dokumentoimattomien muutosten seuraamisesta ja ratkaisuiden ymmärtämisestä ilman kattavaa dokumentaatiota. Dokumentaation merkityksen väheneminen pyritään ratkaisemaan jatkuvalla viestinnällä ja esimerkiksi Scrum-palaverilla, jossa on päivittäin mahdollisuus kertoa oman työn edistymisestä.

Kehitystiimin työskentely-ympäristön haasteiksi on koettu puolestaan samassa tilassa työskentelevillä työympäristön rauhattomuus, ja hajautetuilla tiimeillä viestinnän ja kommunikaation ongelmat. Tiimin yhteisillä viestinnän pelisäännöillä voidaan varmistaa sekä samassa tilassa työskentelevän että hajautetun tiimin kommunikaation sujuvuus ja kehittämisen tehokkuuden maksimointi tältä osin. Tiimin kannattaa pyrkiä sopimaan viestinnästä ensimmäisessä sprintin suunnittelupalaverissa tai mahdollisessa Scrum-koulutuksessa, jotta projektin edistyessä jokaiselle jäsenelle olisi selvää, kuinka usein viestitään, mistä asioista ja missä kanavissa.

Scrumin suorittamiseen ja tehokkuuteen liittyvät haasteet aiheutuvat pääasiassa menetelmän heikosta tuntemuksesta kehitystiimeissä. Jos tiimi ei esimerkiksi ymmärrä retrospektiivin merkitystä koko projektin laadullisen kehittämisen kannalta, on vaarana, että menetelmästä ei saada maksimaalista hyötyä. Lisäksi lähdetutkimuksissa nousee ilmi haaste tapahtumien ajoittamisesta ja menetelmän muokattavuudesta omaan projektiin sopivaksi.

Vaatimuksiin ja järjestelmään liittyviä haasteita ovat tuotteen tehtävälistan hallinta, sekä tuoteversion laatu ja testaaminen. Puutteelliset tai väärin tulkitut vaatimukset johtavat projektia väärään suuntaan, joten vaatimusten selvittämiseen tarvitaan riittävää yhteistyötä asiakkaan kanssa. Testaamiseen sekä laatuun liittyvät haasteet aiheutuvat etenkin projektien nopeasta työtahdista. Kattavalla järjestelmän vaatimusmäärittelyllä ja oikeanlaisella testausympäristöllä pyritään ratkaisemaan kyseisiä haasteita.

Viimeisenä haasteena tutkielmassa esitellään mittaamiseen ja arviointiin liittyvät haasteet, joita ovat etenkin kehitettävän järjestelmän laadullinen mittaaminen, sekä projektin riskinhallinta. Molemmat haasteet aiheutuvat pääasiassa siitä, että Scrum-menetelmä ei anna kehitystiimille valmiita työkaluja

kummankaan osa-alueen arviointiin. Ongelmaa voisi pyrkiä ratkaisemaan samoin kuten Scrumin muokattavuuden haastetta: arvioimalla ensin tiimin ominaisuuksia, työtapoja sekä osaamista kehitystiimin jäsenten kesken, ja tämän jälkeen etsimällä parhaat tavat mitata projektin laadullisia ominaisuuksia sekä riskejä.

Tutkielma pyrkii vastaamaan tutkimuskysymykseen: ”Mitkä ovat Scrum-menetelmän haasteet tietojärjestelmäkehitysprojekteissa?”. Tutkielmassa esitellyt haasteet voidaan peilata alaluvussa 2.2 esiteltyihin Agile Manifeston (Ketterän Manifestin) neljään ketteryyden perusajatukseen. Yksilöiden ja vuorovaikutuksellisuuden huomioiminen ennen kehitysprosesseja ja työvaiheita voi aiheuttaa viestinnän ja jatkuvan kommunikaation kuormittavuutta, mutta kehitystiimille mukautetussa viestinnän mallissa lisätä läpinäkyvyyttä ja joustavuutta kehitysprojektiin kokonaisuudessaan. Toimivan järjestelmän toteuttaminen ennemmin kuin kokonaisvaltainen dokumentointi tuo haasteita dokumentaation laatuun ja sen yhteistoimivuuteen vaatimusmäärittelyn kanssa, mutta luo lisää aikaa kehittämiselle ja tekee kehittämisestä suoraviivaisempaa. Jatkuvan asiakasyhteistyön ja kattavan vaatimusmäärittelyn painottaminen sopimusneuvotteluiden sijaan voi olla haasteellinen asiakkaan sitoutumisen näkökulmasta, mutta luo myös kehitysprojekteille avoimuutta ja mahdollisuuksia luoda asiakkaalle paras mahdollinen lopputuote. Viimeisenä neljästä ketteryyden perusajatuksesta asiakkaan muuttuviin vaatimuksiin reagoiminen voi olla haasteellista nopeatempoisesta työtahdista ja vaatimusten yhteensovittamisesta johtuen, mutta alkuperäisen suunnitelman muokkaamismahdollisuus johdattaa ennen kaikkea projektin luomaan asiakkaalle parhaan mahdollisen lopputuotteen.

Kaiken kaikkiaan useimmat tutkielmassa esitellyt haasteet liittyvät pohjimmiltaan Scrumin ymmärtämiseen viitekehyksenä sekä projektinhallintamenetelmänä. Lisäksi taustalla vaikuttava ketteryys ja sen arvojen käsittäminen laajemmassa mittakaavassa antavat pohjan myös Scrumin toteuttamiseen järjestelmäkehitysmenetelmänä. Haasteita ja ongelmia aiheutuu tilanteissa, kun tiimi tai asiakas ei ymmärrä Scrumin tapahtumien merkitystä tai miksi esimerkiksi kaiken tulisi olla läpinäkyvää koko projektin ajan, oli kyse sitten onnistumisista tai epäonnistumisista. Jatkotutkimuksen kannalta voisi olla tarpeellista selvittää, kuinka Scrum-menetelmä voidaan ottaa käyttöön mahdollisimman sujuvasti niin, että jokainen tiimin jäsen sekä asiakas ymmärtää mitä lisäarvoa menetelmällä voidaan tuoda kehitysprojektiin. Lisäksi Scrumin kokonaisvaltaisen kouluttamisen merkitys projektinhallinnan sujuvuuden näkökulmasta voisi toimia jatkotutkimusaiheena tutkielmassa esiteltyyn Scrumin haastenäkökulmaan.

LÄHTEET

Akif, R. & Majeed, H. (2012). Issues and challenges in scrum implementation. *International Journal of Scientific & Engineering Research, 3*(8), 1-4.

Beck, K. (1999). Embracing change with extreme programming. *Computer*, *32*(10), 70-77.

Beck, K., Beedle, M., Van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., . . . Jeffries, R. (2001). Manifesto for agile software development. Haettu 27.9.2018 osoitteesta https://moodle2016-17.ua.es/moodle/pluginfile.php/80324/mod_resource/content/2/agile-manifesto.pdf

Cho, J. (2008). Issues and challenges of agile software development with SCRUM. *Issues in Information Systems, 9*(2), 188-195.

Cockburn, A. & Highsmith, J. (2001). Agile software development, the people factor. *Computer, 34*(11), 131-133.

Cohen, D., Lindvall, M. & Costa, P. (2003). Agile software development. *DACS SOAR Report, 11*, 2003. Haettu 27.9.2018 osoitteesta http://users.jyu.fi/~mieijala/kandimateriaali/Agile%20software%20development.pdf

Davis, N. (2013). Driving quality improvement and reducing technical debt with the definition of done. (s. 164-168) IEEE.

Edwards, M. D. (2008). Overhauling a failed project using out of the box scrum. (s. 413-416) IEEE.

Eloranta, V., Koskimies, K., Mikkonen, T. & Vuorinen, J. (2013). Scrum anti-patterns--an empirical study. (s. 503-510) IEEE.

Faniran, V. T., Badru, A. & Ajayi, N. (2017). Adopting scrum as an agile approach in distributed software development: A review of literature. (s. 36-40) IEEE.

Fowler, M. & Highsmith, J. (2001). The agile manifesto. *Software Development, 9*(8), 28-35.

Ghosh, G. K. (2012). Challenges in distributed scrum. (s. 200) IEEE.

Gold, B. & Vassell, C. (2015). Using risk management to balance agile methods: A study of the scrum process. (s. 49-54) IEEE.

Hajjdiab, H. & Taleb, A. S. (2011). Adopting agile software development: Issues and challenges. *International Journal of Managing Value and Supply Chains (IJMVSC), 2*(3), 1-10.

Highsmith, J. & Cockburn, A. (2001). Agile software development: The business of innovation. *Computer, 34*(9), 120-127.

Hu, Z., Yuan, Q. & Zhang, X. (2009). Research on agile project management with scrum method. (s. 26-29) IEEE.

Jiang, L. & Eberlein, A. (2009). An analysis of the history of classical software development and agile development. (s. 3733-3738) IEEE.

Lanubile, F. (2006). Collaboration in distributed software development. *Software engineering* (s. 174-193) Springer.

Lapham, M. A., Williams, R., Hammons, C., Burton, D., & Schenker, A. (2010). *Considerations for using agile in DoD acquisition* (No. CMU/SEI-2010-TN-002). CARNEGIE-MELLON UNIV PITTSBURGH PA SOFTWARE ENGINEERING INST.

Larman, C. & Basili, V. R. (2003). Iterative and incremental developments. a brief history. *Computer, 36*(6), 47-56.

Larson, E. W. & Gray, C. F. (2015). A guide to the project management body of knowledge: PMBOK (®) guide. Project Management Institute.

López-Martínez, J., Juarez-Ramirez, R., Huertas, C., Jiménez, S. & Guerra-Garcia, C. (2016). Problems in the adoption of agile-scrum methodologies: A systematic literature review. (s. 141-148) IEEE.

Palmquist, M. S., Lapham, M. A., Miller, S., Chick, T., & Ozkaya, I. (2013). *Parallel worlds: Agile and waterfall differences and similarities* (No. CMU/SEI-2013-TN-021). CARNEGIE-MELLON UNIV PITTSBURGH PA SOFTWARE ENGINEERING INST.

Rising, L. & Janoff, N. S. (2000). The scrum software development process for small teams. *IEEE Software, 17*(4), 26-32.

Santos, V. A., Goldman, A., Shinoda, A. C. M. & Fischer, A. L. (2011). A view towards organizational learning: An empirical study on scrum implementation. (s. 583-589)

Schwaber, K. & Sutherland, J. (2012). *Software in 30 days: How agile managers beat the odds, delight their customers, and leave competitors in the dust* John Wiley & Sons.

Srivastava, A., Bhardwaj, S. & Saraswat, S. (2017). SCRUM model for agile methodology. (s. 864-869) IEEE.

Therrien, I. & LeBel, E. (2009). From anarchy to sustainable development: Scrum in less than ideal conditions. (s. 289-294) IEEE.

Vijay Anand, R. & Dinakaran M. (2015). Issues in scrum agile development principles and practices in software development. *Indian Journal of Science and Technology, 8*(35)

Wan, J., Zhu, Y. & Zeng, M. (2013). Case study on critical success factors of running scrum. *Journal of Software Engineering and Applications, 6*(02), 59.

Woodward, C. J., Cain, A., Pace, S., Jones, A. & Kupper, J. F. (2013). Helping students track learning progress using burn down charts. (s. 104-109)

Zeaaraoui, A., Bougroun, Z., Belkasmi, M. G. & Bouchentouf, T. (2013). User stories template for object-oriented applications. (s. 407-410) IEEE.

LIITE 1 SCRUMIN HAASTEIDEN LÄHDETUTKIMUKSET

Nimi	Julkaisupaikka	Tutkimuksen tyyppi	Ratkaisut	Hakusana ja -paikka	Viittausmäärä*
Akif, R. & Majeed, H. (2012). Issues and challenges in scrum implementation	International Journal of Scientific & Engineering Research Volume 3, Issue 8 August 2012	Laadullinen, kysely	x	"Scrum challenges" Google Scholar	35
Cho, J. (2008). Issues and challenges of agile software development with SCRUM	IACIS International Association for Computer Information Systems VOL IX No. 2, 2008	Laadullinen, haastattelu		"Scrum challenges" Google Scholar	78
Edwards, M. D. (2008). Overhauling a failed project using out of the box scrum	Agile 2008 Conference, IEEE Computer Society	Laadullinen, seurattu Scrum-tiimin toimintaa	x	"Scrum problem" IEEE Digital Library	7
Eloranta, V., Koskimies, K., Mikkonen, T. & Vuorinen, J. (2013). Scrum anti-patterns--an empirical study	20th Asia-Pacific Software Engineering Conference 2013	Laadullinen, kysely + haastattelu	x	"Scrum problem" IEEE Digital Library	12
Gold, B. & Vassell, C. (2015). Using risk management to balance agile methods: A study of the scrum process	2nd International Conference on Knowledge-Based Engineering and Innovation 2015 November 5-6	Laadullinen, haastattelu		"Scrum question" IEEE Digital Library	6
Hajjdiab, H. & Taleb, A. S. (2011). Adopting agile software devel-	International Journal of Managing Value and Supply Chains (IJMVSC) Vol 2 NO.3, Septem-	Laadullinen, case study	x	"Scrum challenges" Google Scholar	35

opment: Issues and challenges	ber 2011				
López-Martínez, J., Juarez-Ramirez, R., Huertas, C., Jiménez, S. & Guerra-Garcia, C. (2016). Problems in the adoption of agile-scrum methodologies: A systematic literature review	4th International Conference in Software Engineering Research and Innovation 2016	Laadullinen, kirjallisuuskatsaus	x	"Scrum problem" IEEE Digital Library	16
Santos, V. A., Goldman, A., Shinoda, A. C. M. & Fischer, A. L. (2011). A view towards organizational learning: An empirical study on scrum implementation	University of São Paulo	Laadullinen, haastattelu		"Scrum challenges" Google Scholar	8
Therrien, I. & LeBel, E. (2009). From anarchy to sustainable development: Scrum in less than ideal conditions	2009 Agile Conference IEEE Computer Society	Laadullinen, seurattu Scrum-tiimien toimintaa	x	"Scrum complexity" IEEE Digital Library	8
Vijay Anand & Dinakaran (2015). Issues in scrum agile development principles and prac-tices in software development	Indian Journal of Science and Technology Vol.8(35)	Laadullinen, systemaattinen kirjallisuuskatsaus		"Scrum challenges" Google Scholar	3

*viittausmäärä Google Scholarissa 6.11.2018