Tampereen yliopisto

Lotte Karlsson

OHJELMISTOPROJEKTIEN RISKIENHALLINTA JA KETTERÄT PROJEKTINHALLINTAMENETELMÄT

Informaatioteknologian ja viestinnän tiedekunta

Kandidaatintutkielma

Maaliskuu 2024

TIIVISTELMÄ

Lotte Karlsson: Ohjelmistoprojektien riskienhallinta ja ketterät projektinhallintamenetelmät
Kandidaatintutkielma
Tampereen yliopisto
Tietojenkäsittelytieteiden tutkinto-ohjelma
Maaliskuu 2024

Riskien toteutuminen ohjelmistoprojekteissa voi johtaa aikataulujen ja budjettien ylityksiin. Ohjelmistoprojektit pystyvät valmistautumaan riskeihin toteuttamalla riskienhallintaa. Nykyaikaisten ohjelmistoprojektitiimien suosimat ketterät työskentelytavat ovat kuitenkin vaikuttaneet riskienhallinnan toteutukseen. Tässä kandidaatintutkielmassa tutkitaan riskienhallintaa, ketteriä menetelmiä ja niiden suhdetta toisiinsa ohjelmistoprojekteissa. Painotus työssä on riskienhallinnan roolin selvittäminen ketterässä työskentelyssä ja aihepiiriin liittyvien haasteiden tarkastelu ohjelmistoprojektien kontekstissa. Tutkielma on toteutettu kirjallisuuskatsauksen muodossa.

Työ on toteutettu kolmessa käsittelyluvussa. Ensimmäisenä esitellään riskienhallinnan konseptia ja siihen liittyviä yleisiä toimintatapoja. Riskienhallintaprosessi kuvataan alusta loppuun seitsemässä eri vaiheessa. Toisessa kappaleessa käsitellään ketterää ajattelua ja työskentelyä. Kappaleessa tutustutaan ketteriin periaatteisiin ja yleisiin toimintatapoihin, jotka yhdistävät ketteriä kehyksiä. Lisäksi tutustutaan Scrumiin ja Kanbaniin, jotka ovat kaksi ohjelmistoprojekteissa suosittua ketterää kehystä. Nämä kaksi kappaletta käsittelevät omia aiheitaan erikseen. Kolmannessa kappaleessa näkökulma keskittyy riskienhallintamallien ja ketterän työskentelyn väliseen suhteeseen.

Tutkielmassa havaittiin riskienhallinnan ja ketterien menetelmien välisen suhteen olevan monimutkainen. Tämä johtuu siitä, että resursseja vievät riskienhallinnan prosessit voivat joissain tapauksissa toimia ketterien menetelmien visiota vastaan. Osa ohjelmistoprojektitiimeistä näkee ketterien menetelmien sisäänrakennettujen toimintojen saavuttavan riskienhallinnan edut ilman erityistä riskienhallinnan implementaatiota. Edeltävien syiden vuoksi riskienhallinta voi joissain tapauksissa jäädä riittämättömälle tasolle. Varsinkin laajemmat ketterät ohjelmistoprojektit hyötyvät eksplisiittisten riskienhallintamallien käyttöönotosta. Ketteriin kehyksiin sopivia riskienhallintamalleja on kehitetty, mutta ne eivät ole saavuttaneet laajaa käyttöä. Tutkielma toteaa jatkokehityksen tarpeen olemassa oleville malleille varsinkin pitkäaikaisten käytännön kokeilujen muodossa.

Avainsanat: riskienhallinta, ketterät menetelmät, ohjelmistokehitys, projektityöskentely, projektinhallinta

Tämän julkaisun alkuperäisyys on tarkastettu Turnitin OriginalityCheck –ohjelmalla.

Sisällysluettelo

1 Johdanto **1**

2 Riskit ja riskienhallinta projektityöskentelyssä **2**

2.1 Riskienhallinnan suunnittelu 3

2.2 Riskien tunnistaminen 4

2.3 Riskien laadullinen ja lukumäärällinen analyysi 5

2.4 Riskivastatoimet ja monitorointi 7

3 Ketterät projektinhallintamenetelmät **8**

3.1 Ketteriä menetelmiä yhdistäviä tekijöitä 9

3.2 Scrum 10

3.3 Kanban 11

4 Ketterien menetelmien suhde riskienhallintaan ohjelmistoprojektissa **12**

5 Keskustelu **14**

6 Yhteenveto **15**

Lähdeluettelo **16**

1 Johdanto

Ketterät ohjelmistokehitysmenetelmät ovat olleet jo pitkään suosiossa ohjelmistoprojekteissa. Ketteryyden tuomat edut kuten mukautuvuus muutoksiin, jatkuva vuorovaikutus asiakkaiden kanssa ja inkrementaalinen kehitystapa ovat keskiössä ketterässä työskentelyssä. Ohjelmistokehitysprojektit ovat kuitenkin kehitystavasta riippumatta alttiita erilaisille riskeille, ja riskienhallinnan huomioiminen on yhä relevanttia. Epäonnistuneet ohjelmistoprojektit viivästyvät, sisältävät teknisiä ongelmia ja voivat aiheuttaa mittavia taloudellisia menetyksiä (Krasner, 2018).

Projektit voivat välttää epäonnistumisia onnistumalla riskienhallinnan implementoimisessa. Riskienhallinnan prosesseja ovat muun muassa projektiin kuuluvien riskien selvitys sekä analyysi riskien vaikutuksista ja todennäköisyyksistä. Lisäksi riskeihin varaudutaan riskivastausten muodossa, ja riskejä monitoroidaan koko projektityöskentelyn ajan. Näiden prosessien implementoinnissa voidaan kuitenkin kohdata erilaisia haasteita, varsinkin suhteessa ketteriin kehyksiin.

Tämän tutkielman pyrkimys on kirjallisuuskatsauksen muodossa selvittää riskienhallinnan yleisiä prosesseja, esitellä ketterää ajattelua ja ketterien kehyksien toimintaa sekä tarkastella näiden kahden aihepiirin suhdetta toisiinsa. Tutkielma on jaoteltu seuraavalla tavalla. Luku 2 sisältää käytetyt tutkimusmenetelmät liittyen tiedonhakuun. Luku 3 esittelee riskienhallinnan yleisiä prosesseja. Ydinlähteenä tämän kappaleen esiteltyihin metodeihin on käytetty ohjelmistoalalla arvostettua riskienhallintateosta - PMBOK-opasta (*Project Management Body of Knowledge*) (Project Management Institute, 2017; Vieira et al., 2020). Luku 4 sisältää esittelyä ketteristä menetelmistä, ja kahdessa luvun 4 alikappaleessa tutustutaan kahteen ketterään kehykseen - Scrumiin ja Kanbaniin. Luku 5 käsittelee ketterien kehysten käytön suhdetta riskienhallintaan. Luku 6 on keskustelukappale, jossa pohdin mahdollisia johtopäätöksiä tutkielman pohjalta. Luku 7 kattaa yhteenvedon koko kirjallisuuskatsauksesta.

Tutkimusmenetelmiin kuului useiden verkkopalvelujen ja tietokantojen käyttöä. Kirjallisuutta haettiin Tampereen yliopiston tarjoamasta hakupalvelu Andorista, jonka kautta löytyi relevanttia opaskirjallisuutta ja tieteellisiä tekstejä. Lisäksi tietoa haettiin Google Scholar -palvelusta ja tietojenkäsittelyn keskeisestä tietokannasta nimeltä *ACM* (*Association for Computing Machinery*), josta on löydetty relevantteja tutkimusartikkeleja.

Hakupalveluissa ja tietokannoissa käytettiin hakusanoja kuten “Agile”, “Ketterät menetelmät”, “Riskienhallinta”, “Risk management”, “Project Management”,

“Projektienhallinta”, “Project”, “Software”, “Ohjelmistokehitys”. Termejä yhdistettiin “AND” -operaattorilla hakujen tarkennuksena. Hakusanoilla löytyi suuri määrä lähteitä, joista valikoitiin sellaisia, joiden näkökulma ei ollut liian spesifi. Riskienhallinnan ja ketterien menetelmien suhteen lähteet sisältävät tutkimusartikkeleja ja muita hyödyllisiä julkaisuja. Valintakriteereihin sisältyi vaatimus molempien aiheiden samanaikaisesta käsittelystä, ja sovellettavuus ohjelmistotuotannon kannalta. Riskienhallinnan ja ketterien menetelmien käsittelykappaleiden osalta nähtiin aiheelliseksi ottaa pohjaksi tärkeitä perusteoksia, joilla on suuri määrä viittauksia tieteellisissä teksteissä. Ketterien menetelmien osalta valikoitui ketterän ohjelmistokehityksen julistus, Scrumin osalta virallinen Scrum-opas, ja riskienhallinnan osalta keskeinen PMBOK-opas. (Beck et al., 2001; Project Management Institute, 2017; Schwaber & Sutherland, 2020).

2 Riskit ja riskienhallinta projektityöskentelyssä

Projekti on tilapäinen pyrkimys, jonka tavoite on luoda ainutlaatuinen tuote, palvelu tai lopputulos. Projektit ovat luonteeltaan monimutkaisia ja alttiita riskeille johtuen useista syistä. Esimerkkeinä aikataululliset, taloudelliset, tekniset ja ihmisiin liittyvät tekijät vaikuttavat projektin onnistumiseen, ja aiheuttavat mahdollisia riskejä projektin etenemiselle. Ohjelmistokehitysprojekteissa on omat spesifit riskinsä, sillä ne voivat olla monimutkaisia ja laajoja kokonaisuuksia. Ei ole epätavallista että projektit saattavat epäonnistua täyttämään aikatauluvaatimukset tai budjetin, tai niiden lopputulokset eivät vastaa loppukäyttäjien odotuksia (Bannerman, 2008). Näihin haastavien projektien epävarmuustekijöihin varautuminen vaatii järjestelmällistä lähestymistapaa eli riskienhallintaa. Projektin riskienhallinta tarkoittaa prosesseja, joilla voidaan vähentää projektin kulkua vaikeuttavien *negatiivisten* riskien toteutumisen todennäköisyyttä tai vaikutusta. Projektin tavoitteita edistävien *positiivisten* riskien osalta varaudutaan hyötymään niistä. Positiivisia riskejä ovat esimerkiksi ennakoitua nopeampi projektin valmistuminen, uudet hyödylliset teknologiat ja kilpailijoiden epäonnistuminen. (Project Management Institute, 2017)

Jokaisella projektilla on oma lähestymistapana riskienhallintaan. Ohjelmistoprojekteille on tyypillistä pohjata riskienhallinnan kehys johonkin standardiin, oppaaseen tai alakohtaisiin asiantuntijatahoihin (Vieira et al., 2020). Tietyt riskienhallintastandardit, kuten *ISO 14971* ovat suunniteltu jonkin spesifien tuoteryhmien kehittämistä varten. ISO 14971 on kehitetty projekteille joissa kehitetään lääketieteellisiä

laitteita (Teferra, 2012). Yleisempiä, projektin luonteesta riippumattomia standardeja ja teoksia on kuitenkin olemassa, kuten *ISO 31000* (*2018*) ja *PMBOK* (2017). Standardeja on useita, mutta johdonmukaisuuden vuoksi seuraavissa osissa keskitytään pääasiassa yhden pohjateoksen näkökulmaan. PMBOK-opas on projektinhallinnan standardi, joka käsittelee muiden aiheiden lisäksi myös riskienhallintaa. Teos esittää riskienhallinnan prosessit seitsemään osaan:

1. Riskienhallinnan suunnittelu
2. Riskien tunnistaminen
3. Riskien kvalitatiivinen analyysi
4. Riskien kvantitatiivinen analyysi
5. Riskivastatoimien suunnittelu
6. Riskivastausten implementointi
7. Riskien monitorointi. (Project Management Institute, 2017)

2.1 Riskienhallinnan suunnittelu

Riskienhallinnan suunnittelu on riskienhallintaprosessin ensimmäinen askel. Sen huolellinen laatiminen edesauttaa projektin onnistumista. Zwikael ja Sadeh (2006) toteavat että riskienhallinnan suunnitteluun laatuun panostaminen edesauttaa projektin menestymistä varsinkin projekteissa, joissa riskin määrä tai projektin koko on suuri. Suunnitelmassa tehdään linjauksia koskien riskienhallinnan budjettia, riskien tunnistamisen metodeita, ajoitusta ja valittuja työkaluja. Riskienhallinnan suunnitelmaan sisältyy myös mahdolliset vastuuhenkilöt, alustavat todennäköisyys- ja vaikutustaulukot sekä alustavia riskikategoriakuvauksia.

Yksi tapa kuvata riskikategorioita on *riskien luokittelurakenne RBS* (*Risk Breakdown Structure*), jossa luodaan kategorioita riskeille. Esimerkkejä ovat tekniset riskit, hallinnolliset riskit ja ulkopuoliset riskit. Näihin abstrakteihin kategorioihin voidaan sijoittaa tarkentavat alakategoriat kuten virhearviot projektin aikamääreistä, ongelmat kommunikaatiossa ja lainsäädännölliset ongelmat. (Project Management Institute, 2017)

Todennäköisyys- ja vaikutustaulukot ovat taulukkoja, joihin voidaan myöhemmin sijoittaa analyysivaiheissa konkreettisesti riskejä. Todennäköisyystaulukossa voidaan esittää riskien tapahtumisen todennäköisyyksiä joko sanallisesti tai numeerisesti. Sanalliset kuvaukset todennäköisyystaulukossa ovat esimerkiksi “korkea todennäköisyys” tai “matala todennäköisyys”. Vaikutustaulukossa määritellään sekä negatiivisten, että positiivisten riskikategorioiden vaikutus projektiin. Projektin koko ja resurssit vaikuttavat siihen,

minkälaiset vaikutukset koetaan suurina riskeinä. Mahdollisia vaikutuksia ovat esimerkiksi ajallisten tai rahallisten resurssien menetys. (Project Management Institute, 2017)

Kun suunnitelma on valmis, riskienhallinnan ja projektin käyttötarkoituksiin varataan projektidokumentit. *Projektidokumentit* ovat sijainti kaikelle projektin riskienhallintaan liittyvälle tiedolle. Suunnitelman kaikki osat lisätään projektidokumentteihin, kun ne valmistuvat. Dokumentit toimivat keskeisenä ja helposti saatavilla olevana tiedonlähteenä kaikille projektissa mukana oleville sidosryhmille. *Riskirekisteri* on tärkeä osa projektidokumentteja, joihin sijoitetaan kaikki riskienhallinnan työskentelyn havainnot, analyysit, taulukot ja työnjaot. Projektidokumentteja ja riskirekisteriä on tarkoitus päivittää iteratiivisesti koko projektityöskentelyn ajan. Ajantasainen rekisteri mahdollistaa riskien seurannan ja kyvyn vastata muuttuviin olosuhteisiin (Project Management Institute, 2017)

2.2 Riskien tunnistaminen

Riskienhallinnan toinen osa on tunnistaminen, joka viittaa sekä yksittäisten riskien että laajempien projektiriskien tunnistamiseen ja dokumentointiin. Riskien tunnistamista on kuvattu iteratiiviseksi prosessiksi, johon voivat osallistua kaikkia projektiin liittyvät osapuolet, kuten kehittäjätiimi, projektin johto, omistajat ja loppukäyttäjät. Tarkoituksena on tunnistaa ja listata kaikki mahdolliset projektissa tapahtuvat riskit. Projektit voivat olla laajoja kokonaisuuksia, ja siksi varsinkin suuria projekteja on helpompaa käsitellä osissa riskien tunnistamisvaiheessa (Pasha et al., 2018).

Riskien tunnistamiseen on useita eri tapoja. Yksi esimerkki on ideariihi (*brainstorming*), jossa osallistujat yrittävät tunnistaa riskejä joko vapaamuotoisen tai järjestelmällisen tapahtuman aikana. Ideariihessä voidaan käytännössä käyttää mitä vain riskien tunnistamismetodeja. Vaihtoehtona PMOK-opas ehdottaa tehtävälistaa (*checklist*), jossa on listattuna mahdollisissa aikaisemmissa projekteissa tapahtuneita riskejä. Lista voidaan koota esimerkiksi ideariihessä. Tehtävälistan käsittelyn aikana on tarkoitus merkitä sekä vanhat riskit aiemmasta listasta, ja myös löytää uusia potentiaalisia riskejä. Mikäli tehtävälistaa käytetään, on tarkoituksena myös päivittää sitä iteratiivisesti. Listan päivittämiseen on tärkeä sitoutua, sillä muuten listan kirjoittamiseen kulutetut resurssit menevät hukkaan, tai listaa jatkossa hyödyntävä tiimi ei voi luottaa siihen (Zhou et al., 2008). Riskejä voidaan löytää myös esimerkiksi haastatteluilla, joihin voi osallistua projektiin liittyviä osapuolia ja asiantuntijoita. (Project Management Institute, 2017)

Eräs keino riskien tunnistamiseen on juurisyyanalyysi, jossa koetetaan löytää syyt, jotka johtavat johonkin tunnistettuun ongelmaan, kuten projektin viivästymiseen tai

budjetin ylittymiseen. Ohjelmistokehityksen kontekstissa Lehtinen ja Mäntylä (2011) jakavat juurisyyanalyysin kategoriat neljään osaan: ihmiset, tehtävät, metodit ja ympäristö. Juurisyyanalyysi voidaan toteuttaa eri tavoilla, kuten "viisi miksi-kysymystä" -metodilla tai käyttämällä *FTA*-metodia (*Fault Tree Analysis*). Viisi miksi-kysymystä viittaa työskentelytapaan, jossa esitettyyn ongelmaan pureudutaan esittämällä ongelma, ja pohtimalla syytä, josta ongelma johtuu. Tätä kysymystä on tarkoitus toistaa useita kertoja – ehdotetusti 5. Ajatuksena on, että selvittämällä syyn syitä, voidaan selvittää ongelman juurisyy. FTA-metodissa kuvataan riskit puumallissa, jossa alkuperäisestä teoreettisesta riskistä haarautuu useita haaroja, jotka ovat mahdollisia syitä riskin kehittymiselle. Solmuna kuvattu riski voi koostua useista aliriskeistä. Solmun aliriskien suhde muihin saman tason aliriskeihin kuvataan "OR" ja "AND" -logiikkaoperaattoreilla. Solmu toteutuu silloin kun aliriskit toteutuvat logiikkaoperaattorin mukaan. (Project Management Institute, 2017)

Toinen riskien tunnistamistapa on SWOT-analyysi (*Strengths, Weaknesses, Opportunities, Threats*), jossa tunnistetaan yrityksen tai projektin sisäisiä ja ulkoisia tekijöitä. Vahvuuksia voivat olla kilpailuetuja edistävät piirteet kuten esimerkiksi projektin sisäinen osaaminen tai vahvat asiakassuhteet. Heikkoudet rajoittavat projektin suorituskykyä tai potentiaalia, ja niihin kuuluu muun muassa tekniset ongelmat tai resurssien puute. Mahdollisuudet viittaavat sellaisiin ulkopuolisiin tekijöihin, jotka tuovat etuja tai kasvumahdollisuuksia projektille. Esimerkkejä mahdollisuuksista ovat uudet teknologiat, markkinoiden kasvu ja muuttuvat trendit kuluttajien keskuudessa. Uhkiin kuuluvat negatiiviset ulkoiset tekijät kuten talouden epävarmuus, lainsäädännön muutokset ja teknologiset haasteet. (Puyt et al., 2023)

Sekä juurisyyanalyysi että SWOT-analyysi ovat tapoja tunnistaa projektiin liittyviä riskitekijöitä ja selkeyttää riskiympäristöä. Tunnistetut riskit listataan riskiraporttiin, joka sisällytetään projektidokumentteihin. Riskiraportti sisältää sekä yksittäiset riskit, projektiriskit, niiden alakategorisoinnit ja muut tunnistustyöskentelyn lopputulokset. Riskiraporttia on tarkoitus käyttää iteratiivisella tavalla, joka näkyy riskiraportissa ja riskienhallintatyössä siten, että tunnistettuja riskejä päivitetään projektidokumentteihin myös projektin kulun myötä. (Project Management Institute, 2017)

2.3 Riskien laadullinen ja lukumäärällinen analyysi

Riskien laadullisella analyysillä viitataan tunnistamisvaiheessa dokumentoitujen riskien käsittelyyn, jonka tarkoituksena on asettaa riskeille prioriteetit. Tämä voidaan saavuttaa

analysoimalla yksittäisien riskien todennäköisyyksiä ja vaikutuksia. Riskienhallinnan suunnitelmavaiheessa luotiin aikaisemmin kehykset sille, minkälaiset vaikutukset koetaan isoksi riskeiksi projektityölle, ja minkälaisia todennäköisyyksiä riskeillä voi olla. Potentiaalisia riskejä on myös tässä vaiheessa voitu jo alustavasti kategorisoida, jos projekti on käyttänyt RBS-luokittelurakennetta. (Project Management Institute, 2017) Riskien laadullisessa analyysissä määritelty asiantuntija tai asiantuntijaryhmä käsittelee tunnistetut riskit ja asettaa ne näihin ennalta määriteltyihin riskikehyksiin. Usein riskien asettaminen tehdään työpajan tai ideariihen muodossa. Ohjelmistoprojekteissa on kuitenkin huomattu, että usein vain yksi tai muutama henkilö osallistuu projektin laadulliseen analyysiin, ja lopputuloksena on suppeat kuvaukset riskeistä. Onnistuneessa analyysissä tulee prosessiin käyttää resursseja kuten tarvittavaa ammattitaitoa, kokemusta ja näkökulmia projektin eri osista. Analyysin tukena voidaan käyttää alan tutkimuksia vastaavista projekteista tai muuta julkaistua materiaalia. Laadullisen analyysin lopputulos on päivitetty riskirekisteri riskidokumenteissa. (Beinschróth, 2022)

Riskien lukumäärällisellä analyysillä viitataan riskejä koskevan datan numeeriseen analyysiin matemaattisilla malleilla. Tavoitteena on analysoida yksittäisten riskien yhteisvaikutusta. Lukumäärällinen analyysi on kuitenkin usein työlästä ja kallista. Sitä voidaan käyttää, mikäli riskidataa on tarpeeksi, ja sitä on muokattu laadullisessa analyysissä numeeriseen muotoon. Näiden syiden vuoksi lukumäärällistä analyysiä sovelletaan usein vain suuriin projekteihin, joihin on mahdollista hyödyntää jotain aiheeseen erikoistunutta kansainvälistä konsultointipalvelua. (Ghane, 2017)

Analyysiprosessissa valitaan jokin paras mahdollinen mallinnustapa ja siihen liittyvät työkalut, usein jokin valmis sovellus, ja käsitellään dataa sen kautta. Konsultointipalvelujen metodit tähän prosessiin eivät yleensä ole julkisia (Beinschróth, 2022). Valitun työkalun avulla luodaan simulaatio riskien yhteisvaikutuksesta, ja muodostetaan mallinnus. Tätä mallinnusta voidaan käyttää esimerkiksi projektia koskevien päätösten teossa ja viestinnässä sidosryhmien kanssa. (Project Management Institute, 2017)

Yksi tällainen mallinnus voidaan tehdä projektin aikataulutuksesta. *Monte Carlo -metodi* on usein käytetty tapa ennustaa epävarman tapahtuman lopputuloksia, ja sitä voidaan hyödyntää esimerkiksi projektin aikataulun lukumäärällisessä analyysissä. Tässä tapauksessa Monte Carlo -metodissa projekti jaetaan osiin, joilla on määritelty optimistiset, pessimistiset ja todennäköisimmät kestot. Ohjelmistoa hyödyntäen, näille kestoille luodaan tuhansia satunnaisia arvoja, jolloin saadaan projektin eri osien kestojen

todennäköisyysjakaumat. Arvoja voidaan käyttää luomaan malli, jota simuloimalla voidaan ennustaa todennäköisyyttä suoriutua projektista tietyssä aikarajassa. (Zhang & Jin, 2020)

Projektin aikataulutusta on myös mahdollista analysoida kriittisyysanalyysillä tässä vaiheessa riskienhallintaprosessia. Sen päämääränä on selvittää yksittäisten tapahtuman kriittisyys numeerisesti, ja määrittää sen vaikutus *kriittiseen polkuun*. Kriittinen polku viittaa riskienhallinnassa tapahtumajärjestykseen, jossa tietyssä vaiheessa toteutuva riski vaikuttaa negatiivisesti projektin loppuosaan esimerkiksi viivästymisen muodossa. Lukumäärällisen analyysin kokonaislopputuloksena on yksityiskohtaista todennäköisyysanalyysiä, kuvaajia, ja tekstimuotoista tulkintaa tuloksista. Tiedot tallennetaan projektidokumentteihin. Uusia luotuja dokumentteja voidaan hyödyntää riskivastatoimien suunnittelussa. (Project Management Institute, 2017)

2.4 Riskivastatoimet ja monitorointi

Riskivastatoimet ovat riskeihin valmistautuvia toimia joiden tarkoitus on vähentää projektin alttiutta riskeille. Vastatoimet suunnitellaan riskirekisterin ja muiden projektidokumenttien pohjalta, joihin on ideaalisessa tilanteessa kerätty aiemmissa vaiheissa kattavat selvitykset projektin yksittäisistä riskeistä, ja laajemman tason projektiriskeistä. Laadukkaat dokumentit mahdollistavat vastatoimien suunnittelun, jossa pystytään priorisoimaan vastatoimet keskenään ja mitoittamaan niiden kustannustehokkuus sopivaksi. (Project Management Institute, 2017)

On olemassa *proaktiivisia* ja *reaktiivisia* vastatoimia. Proaktiivisilla vastatoimilla viitataan toimiin, joilla on tarkoitus välttää tuleva riski, kun taas reaktiivisella vastatoimella viitataan sellaiseen vastatoimeen, joka on suunniteltu aktivoitavan vasta kyseisen yksittäisen riskin toteutuessa. PMOK-opas määrittelee viisi strategiaa uhkien vastatoimien suunnitteluun:

1. *Eskaloiminen*, joka viittaa riskin jättäminen huomiotta projektiryhmän tasolla. Tätä käytetään sellaisessa tilanteessa, jossa riski ei ole projektiryhmän hallittavissa, tai kuulu projektiryhmän valtuuksiin. Jokin muu organisaation elin hoitaa eskaloidun riskin.
2. *Välttäminen*, joka viittaa riskin poistamiseen jollain keinolla kuten projektin rakenteen tai projektityöskentelyn muuttamisella siten että riski vältetään. Tätä voidaan käyttää, jos riski on todennäköinen ja sillä on suuri vaikutus.
3. *Siirtäminen*, joka viittaa riskin vastuun välittämistä toiselle osapuolelle. Tämä voi olla esimerkiksi jokin vakuutus tai muu sopimus.

4. *Lieventäminen,* joka viittaa riskin todennäköisyyden pienentämiseen tai vaikutuksen pienentämiseen. Ohjelmistoprojektissa tämä voisi ilmetä esimerkiksi testaamisen lisäämisellä, tai vähentämällä riippuvuuksia muista palveluista.
5. *Hyväksyminen*, joka viittaa siihen, että kyseinen riski jätetään huomiotta, ja sen tilaa ainoastaan seurataan. Tämä eroaa eskaloinnista siten että kukaan organisaatiossa ei hoida riskiä. Se on sopiva tapa alhaisen prioriteetin riskeihin, joissa on pieni vaikutustekijä. (Project Management Institute, 2017)

Riskivastatoimien suunnittelun lopputulokseksi saadaan päivityksiä projektidokumentteihin, ja muutoksia riskienhallintasuunnitelmaan. Tämän jälkeen suunniteltuja proaktiivisia vastatoimia implementoidaan projektityöskentelyyn ja valmistaudutaan implementoimaan reaktiivisten vastatoimien osalta. Riskejä ja projektityöskentelyä monitoroidaan koko projektityöskentelyn ajan osana riskienhallintaa. Jatkuvan monitoroinnin etu on valmius muutoksiin. Arvioinnin kohteena ovat esimerkiksi riskivastatoimien teho, yksittäisten, laajempien riskien arviointi, riskienhallinnan lähestymistavan sopivuus, projektiin liittyvien oletusten oikeellisuus ja uudet riskit. Riskejä voidaan monitoroida keräämällä dataa projektista ja työskentelyn tahdista, ja kokouksista projektitiimin kanssa. Dokumentaation osalta pidetään kirjaa ajankohtaisiin oletuksiin liittyvistä tiedoista, kohdatuista konkreettisista ongelmista ja muista riskiympäristöön liittyvistä tiedoista. Nämä tiedot lisätään projektidokumentteihin ja niitä päivitetään muutoksien mukaan. (Project Management Institute, 2017)

Yhteenvetona voidaan todeta riskienhallinnan olevan monivaiheinen ja työläs prosessi, joka kuluttaa resursseja projektitiimiltä. Riskienhallintaan kuuluu pitkäjänteinen ja iteratiivinen työskentely koko projektin ajan. Työmäärästä huolimatta onnistunut riskienhallinta on kuitenkin etu ohjelmistoprojektille, sillä se edistää projektien suotuisia lopputulemia (Bannerman, 2008).

3 Ketterät projektinhallintamenetelmät

Ketterät menetelmät (*Agile*) viittaavat projektinhallinnassa käytettäviin kehyksiin, metodeihin ja ajatusmalleihin, jotka painottavat käytännöllistä ja joustavaa näkökulmaa projektin toteuttamiseen. Käytännöllisyys ja joustavuus näkyy ketterissä menetelmissä siten, että ne painottavat vähemmän projektityöskentelyn muodollisia osia. Ketterän ajattelun perusajatukset on listattu *Ketterän ohjelmistokehityksen julistuksessa* (*Agile Manifesto*) seuraavalla tavalla:

1. Yksilöitä ja kanssakäymistä enemmän kuin menetelmiä ja työkaluja
2. Toimivaa ohjelmistoa enemmän kuin kattavaa dokumentaatiota
3. Asiakasyhteistyötä enemmän kuin sopimusneuvotteluja
4. Vastaamista muutokseen enemmän kuin pitäytymistä suunnitelmassa.
   (Beck et al., 2001)

Nämä ajatukset nousivat vastauksena 90-luvun lopun projektityöskentelyn yleisille käytännöille ohjelmistoprojekteissa. Siihen aikaan oli tyypillistä tehdä laajoja ja tarkkoja vaatimusmäärittelyjä ja dokumentaatioita projekteille ennen kuin varsinainen työprosessi aloitettiin. Tarkan suunnittelun ongelmiksi koettiin siihen kuluva aika, ja heikkous tilanteissa, joissa työprosessin aikana ilmeni projektiin vaikuttavia muutoksia. (Girvan & Girvan, 2022)

3.1 Ketteriä menetelmiä yhdistäviä tekijöitä

Ketterä ajattelu toimii pohjana usealle ketterälle menetelmälle ja kehykselle, jotka toimivat ketterien perusajatusten mukaan. Esimerkkejä näistä menetelmistä ja kehyksistä ovat *Scrum*, *Kanban, FDD* (*Feature Driven Design*) ja *XP* (*Extreme Programming*). Vaikka osa näistä on kehitetty ennen ketterän menetelmän konseptia, kuten XP, kategorisoidaan ne yleisesti kuitenkin ketteriksi menetelmiksi. (Girvan & Girvan, 2022)

Ketterillä kehyksillä on yhdistäviä tekijöitä, jotka määrittävät ne konkreettisesti ketteriksi. Yksi ketteriin perusajatuksiin nojaava periaate on työn jakaminen osiin, eli iteratiivisuus. Esimerkiksi Scrumissa työn on jaettu jaksoihin, joissa tiimi sitoutuu suorittamaan tietyn määrän työtä rajoitetussa ajassa. Kanbanissa työn jakaminen tapahtuu visuaalisen hallintajärjestelmän avulla, jossa työ tehdään pieninä, erikseen tunnistettavina tehtävinä. Kanbanissa ei ole siis työjaksoja, vaan iteratiivisuuden ajatellaan näkyvän jatkuvana “virtana”, jossa uusia tehtäviä lisätään tauluun. Työ siirtyy joustavasti eri vaiheiden läpi, ja tiimi voi keskittyä viemään yksittäiset tehtävät loppuun ennen uuden aloittamista. Ketterien kehysten iteratiivisuudessa on siis eroavaisuuksia, mutta ne pohjautuvat samaan ketterään periaatteeseen. (Girvan & Girvan, 2022)

Toinen ketteriä kehyksiä yhdistävä tekijä on inkrementaalisuus. Inkrementaalisuudella viitataan siihen, että projektin iteraatiot rakentuvat aikaisempien iteraatioiden päälle. Konkreettisesti tavoitteena on siis luoda mahdollisimman aikaisin toimiva malli, jota voidaan jatkokehittää. Tällä tavalla projektin toimitusaika on lyhyt – ja tuote käytettävissä ja testattavissa aikaisemmassa vaiheessa. Käytettävästä tuotteesta voidaan kerätä asiakkaalta palautetta, jota voidaan hyödyntää seuraavassa iteraatiossa, jossa tavoitteena on luoda

projektille uusi inkrementti. Tämänkaltainen palautepohjainen asiakasyhteistyö on keskeisessä osassa ketterää työskentelyä.

Kolmas yhteinen tekijä on mukautuvuus. Mukautuvuudella viitataan siihen, että ketteriä kehyksiä on tarkoituksena mukauttaa jokaisen kehitystiimin spesifiin toimintaympäristöön ja vahvuuksiin. Ketterissä perusajatuksissa painotetaankin, että ketterässä työskentelyssä on tärkeää priorisoida sujuvuutta ennalta määritettyjen asioiden ylitse. (Girvan & Girvan, 2022) Jokaisessa ketterässä menetelmässä ja kehyksessä on kuitenkin selkeästi omat tapansa implementoida ketteriä periaatteita. Tarkastellaan seuraavaksi esimerkkinä *Scrumia* ja *Kanbania*, jotka ovat ohjelmistoalalla tunnetuimpien ketterien kehysten joukossa (Pham & Pham, 2017).

3.2 Scrum

Scrum on suosittu kehys ketterään projektityöskentelyyn, jossa työskennellään syklisissä ja iteratiivisissa kehitysjaksoissa, joita kutsutaan Sprinteiksi (*Sprint*). Scrum-ohjeistuksessa Scrum kuvataan tiivistettynä kolmivaiheiseksi seuraavalla toistuvalla rakenteella:

1. Tuoteomistaja ohjaa tuotteen visiota, priorisointia ja tarjoaa projektitiimille tarvittavat vaatimukset sprinttiin
2. Scrum-tiimi tuottaa sprinttiin valitsemastaan työstä arvoa tuottavan *inkrementin*.
3. Scrum-tiimi ja sen sidosryhmät tarkastelevat tuloksia ja mukauttavat tulevia sprinttejä sen pohjalta
4. Toistetaan kohtia 1–3

   (Schwaber & Sutherland, 2020)

Scrumissa projektityötiimiä kutsutaan *Scrum-tiimiksi*, joka koostuu Scrum-ohjeiden mukaan useimmiten alle 10 jäsenestä, jotka on jaettu kolmeen eri työrooliin. Roolit ovat seuraavanlaiset:

1. Scrum Master
2. Tuoteomistaja
3. Kehittäjä

Scrum Masterin tehtävänä on varmistaa, että Scrum-prosessia noudatetaan, ja auttaa Scrum-tiimiä toimimaan tehokkaasti. Roolissa toimiva henkilö myös konkreettisesti järjestää kokoukset. Tuoteomistajan rooli vastaa tuotteen visiosta, tavoitteista, ja tuotteen kehitysjonon (*Product backlog*) valmistelusta ja viestimisestä. Tuotteen kehitysjonolla viitataan Scrumissa listaan, jossa on määriteltynä tuotteeseen liittyviä toteutettavia ominaisuuksia. Scrum-tiimin jäsenet suunnittelevat seuraavan Sprintin siten, että tuotteen

kehitysjonosta valitaan toteutettavat osat inkrementin luomista varten. Valmistellussa Sprintissä on määriteltynä tavoitteet, valitut kohdat kehitysjonosta ja suunnitelma toimintatavoista näiden toteuttamiseksi. Tätä kutsutaan Sprintin kehitysjonoksi (*Sprint backlog*). Scrum-tiimin on tärkeää arvioida toteutettavien ominaisuuksien prioriteettia jatkuvana prosessina. (Schwaber & Sutherland, 2020)

Sprint-jakson aikana Scrum-tiimin jäsenet työstävät Sprintin kehitysjonossa listattuja työtehtäviä. Etenemistä Sprintin aikana tukee päivittäispalaveri (*Daily Scrum*). Päivittäispalaverissa tiimin jäsen selittää edellisen päivän työtuloksista, seuraavan päivän suunnitelmista ja kertoo mahdollisista haasteista. Päivittäispalaverin tarkoitus on synkronoida tiimin jäsenten työskentelyä, luoda avoimuutta tiimin jäsenten kesken ja sitouttaa projektiin. Ketterään ajatteluun ja toimintaan kuitenkin kuuluu mahdollisuus toimia joustavalla tavalla. Scrum-ohjeet painottavat päivittäispalaverin suunnitelmien muokattavuutta yksittäisen päivänkin aikana. Tiivistettynä Sprint-jakson aikana kaikki Scrum-roolit suorittavat omia tehtäviänsä, ja osallistuvat yhteisiin päivittäispalavereihin. (Schwaber & Sutherland, 2020)

Sprint-jakson loppuvaiheilla pidetään *Sprintin katselmointi* (*Sprint review*), joka on työpajamainen kokous, jossa Scrum-tiimi esittelee työnsä tuloksia tärkeimmille sidosryhmille. Tarkoituksena on keskustella iteraation saavutuksista ja tarkastella edistymistä tuotteen tavoitteita kohti. Sprint-jakson lopuksi pidetään Sprintin retrospektiivi (*Sprint retrospective*), jossa Scrum-tiimi arvioi Sprintin sujuvuutta. Sprint retrospektiivissä pääpainopiste on ihmistenvälisissä toimissa ja prosesseissa, ja se eroaa siten Sprintin katselmoinnista, jonka painopiste on tuote ja tuotettu inkrementti. (Schwaber & Sutherland, 2020)

3.3 Kanban

Kanban on toinen usein käytetty kehys ketterään työskentelyyn. Se pohjaa menetelmänsä *Lean*-ajatteluun, joka kehitettiin osana Toyotan tuotantojärjestelmää (T*oyota Production System* tai *TPS*). Lean-ajattelussa keskiössä on analysoida työprosesseja siten, että voidaan tunnistaa tehottomia työtapoja ja vähentää sekä fyysisten, että henkisten resurssien tuhlaamista (Ohno, 1988). Lean-ajattelu on myös läsnä Scrum-kehyksessä, mutta Kanban-kehyksen prosesseissa se on selkeämmin havaittavissa. (Pham & Pham, 2017)

Kanbanista on useita interpretaatioita, ja se on ketterän ketteränä menetelmänä tarkoitettu mukautettavaksi. Pelkistetyssä Kanban-työskentelyssä ei ole kuitenkaan Scrumin tapaan iteraatioita, vaan työkulku on jatkuva ja sitä ei eritellä sprintteihin.

Eroavaisuutena on myös mainittava, että Kanban ei määrittele uusia rooleja kuten Scrumissa. Pääpiirteissään Kanban-työskentely sisältää seuraavia osia:

1. Työn kulun visualisointi
2. Työnkulun hallinta
3. Hidastavien tekijöiden tunnistaminen
4. Keskeneräisen työn volyymin rajoittaminen
5. Palautteen kerääminen työn kehittämiseen
6. Työn arviointi ja optimisointi. (Pham & Pham, 2017)

Työn kulun visualisointi on keskeistä Kanbanissa. Ei ole tarkasti määritelty tapaa, jolla visualisointi tulisi suorittaa, mutta kuuluisin ja käytetyin tapa on käyttää Kanban-taulua. Kanban-taulussa työn vaiheet on jaettu niitä kuvaaviin sarakkeisiin kuten “tekemättä”, “työn alla” tai “valmis”. Työtehtävät asetetaan näihin sarakkeisiin, ja siten voidaan nähdä kokonaiskuva kaikista työprosesseista ja niiden tilasta. Keskeneräisen työn volyymin rajoittaminen on keskeistä Kanbanissa. Uusia työtehtäviä on tarkoitus ottaa käsiteltäväksi vain, jos keskeneräisiä tehtäviä on yhteensä alle sovitun määrän. Ajatuksena on saavuttaa tällä nopeita toimitusaikoja, ja vähentää resurssihukkaa. (Pham & Pham, 2017)

Päivittäispalaveri on osa Kanbania, kuten Scrumissa, mutta eroavaisuutena painotus on työkulussa, toisin kuin Scrumissa, jossa painotus on Sprintin tavoitteissa. Palaverissa keskustellaan työtehtävien etenemisestä, ja niihin vaikuttavista mahdollisista esteistä tai hidasteista. Mahdollisiin muutoksiin voidaan reagoida, sillä tiimin on tarkoitus tavata usein. Uusia työtehtäviä voidaan luoda, mikäli kapasiteetti sen sallii, ja olemassa olevien tehtävien tilaa voidaan muuttaa. Yhteenvetona Kanban-päivittäispalaveri tarjoaa foorumin yhteistyöhön, tiedonvälitykseen ja muutoksiin reagoimiseen. (Pham & Pham, 2017)

4 Ketterien menetelmien suhde riskienhallintaan ohjelmistoprojektissa

Ketterät kehykset, kuten Scrum ja Kanban, tarjoavat joustavia lähestymistapoja ohjelmistoprojektien hallintaan ja toteuttamiseen. Kehysten suhde riskienhallintaan ohjelmistoprojekteissa on monimutkainen ja riippuu suuresti siitä, miten ketteriä menetelmiä sovelletaan ja miten organisaatio käsittelee riskejä (Albadarneh et al., 2015).

Ketterien menetelmien periaatteet työn jakamisesta iteraatioihin ja inkrementaalisuus mahdollistavat joustavuuden sekä kyvyn muutoksiin sopeutumiseen. Jotkut ketteriä kehyksiä käyttävät projektiryhmät eivät tähän ajatukseen viitaten aina käytä erikseen riskienhallintametodeja tai kehyksiä (Vieira et al., 2020). Tämänkaltaista lähestymistapaa

kutsutaan *implisiittiseksi* riskienhallinnaksi. Implisiittisessä riskienhallinnassa tarkoitus on saavuttaa riskienhallinnan edut ketterien kehysten tavanomaisten prosessien aikana – kuten päivittäispalaverien aikana. Schön, Radke ja Jordan (2020) toteavat implisiittisen riskienhallinnan toimivan autonomiselle tiimille, joka kehittää yhtä tuotetta itsenäisesti. Usean tiimin välinen yhteistyö voi lisätä tarvetta erilliselle riskienhallintaprosesseille tai kehyksille (Schön et al., 2020).

Ketterässä työskentelyssä on myös joitain koettuja haasteita suhteessa erillisiin projektiin lisättyihin riskienhallinnan prosesseihin. Perinteiset riskienhallintaprosessit sisältävät paljon alkupainotteista suunnittelua, peilaten vanhanaikaisia ohjelmistokehitysmetodeja, joissa suunnittelu ja toteutus olivat nykyistä enemmän erillisiä osia työprosesseissa. Ongelmana on, että ketterät tiimit saattavat kokea raskaan riskienhallintakehyksen toimivan ketteryyttä vastaan. Useassa ketterässä kehyksessä ei ole myöskään spesifioitu riskienhallinnan integraation yksityiskohtia. (Nyfjord & Kajko-Mattsson, 2007)

Toisena haasteena voi toimia riskianalyysien toteuttaminen liian harvoin, koska niille ei ole järjestetty riittävää aikaa päivittäispalavereissa ja muissa toimissa. Jos projektitiimi toteuttaa riskianalyysiä ja monitorointia vain sprinttien suunnittelun aikana, se ei välttämättä riitä tunnistamaan muuttuvia riskejä. Kolmas ongelma on riskiosaamisen vähyys. Varsinkin pienissä projekteissa ei välttämättä ole sisäistä riskiosaamista, ja sen hankkimisen kustannukset saattavat olla projektin budjetin ulkopuolella. Nämä ovat lisäsyitä, joiden vuoksi jotkut projektit toteuttavat implisiittistä riskienhallintaa. (Muntés-Mulero et al., 2019)

Vaikka joissain ketterissä ohjelmistoprojekteissa turvaudutaan implisiittiseen riskienhallintaan, on myös mahdollista, että ketterien menetelmien epäonnistuminen muodostaa yhden riskitekijän. Mikäli jokin osa ketteriä prosesseja ei toimi, kuten ohjelmistoprojektitiimin kommunikaatio tai yhteistyö sidosryhmien kanssa – projektissa voidaan kohdata ongelmia. (Shrivastava & Rathod, 2019) Muita ketterässä työskentelyssä havaittuja ongelmia ovat muun muassa teknisen velan kasaantuminen, tasapainottelu tuotteen kehitysjonon hallinnan ja uusien ominaisuuksien kehityksen välillä, tehtävien priorisointiongelmat, epäselvä määritelmä valmiista tuotteesta, infrastruktuurin kehittäjien ja sovelluskehittäjien välinen eriytyminen sekä tiedon säilymisen ongelmat, erityisesti vähentyneen dokumentaation vuoksi. (Elbanna & Sarker, 2016) Globaaleissa ympäristöissä kuten monikansallisissa yrityksissä riskien painotukset ovat tiimien kommunikaation ja asiakasyhteistyön piirissä (Podari et al., 2020).

Riskienhallinnalle ei siis aina anneta tarpeellista painoarvoa ketterissä ohjelmistoprojekteissa, tai sen implementoinnissa voidaan kokea ongelmia. Näitä tekijöitä ajatellen on kehitetty jotain avustavia malleja ja kehyksiä. Esimerkkejä näistä ovat Shrivastavan ja Urvashin (2017, 2019), sekä Chaouch et al. (2019) mallit. Kyseiset mallit ehdottavat muunnoksia, parannuksia ja lisäyksiä ketteriin kehyksiin riskienhallinnan lisäämisen edistämiseen. Esimerkiksi Chaouch et al. (2019) ehdottavat PMBOK-tyylisen riskienhallinnan sovittamista Scrumiin, jossa oppaan määrittelemät riskienhallinnan osat suoritettaisiin tietyissä sprintin vaiheissa. Mallit ovat kuitenkin toistaiseksi jääneet teoreettisiksi, sillä niitä ei ole pystytty kokeilemaan tuotannossa joko ollenkaan tai riittävästi. Kirjoittajat painottavat töissään jatkokehityksen tarvetta sekä mallien parantelun että konkreettisten laajempien kokeilujen muodossa.

5 Keskustelu

Tämän kirjallisuuskatsauksena toteutetun tutkielman pyrkimys oli tutustua erikseen riskienhallintaan, ketteriin menetelmiin ja kehyksiin, sekä ketterän työskentelyn suhteesta riskienhallintaan ohjelmistoprojekteissa. Konsepteja eriteltiin käsittelykappaleisiin aiheittain. Tutkielma on esitellyt riskienhallintaprosessien yleisiä käytäntöjä, yleisesti käytettyjä ketteriä kehyksiä - Scrumia ja Kanbania sekä eritellyt riskienhallinnan ja ketterien menetelmien välistä suhdetta. Tutkielman osien yksityiskohtaisuutta rajoitti esiteltävien aiheiden määrä, jonka vuoksi kaikkea materiaalia ei voitu kuvata laajasti.

Tuloksena voidaan todeta riskienhallinnan ja ketterän työskentelyn välillä olevan haasteita. Ketterän työskentelyn luonne sisältää riskienhallintaa edesauttavia elementtejä kuten säännöllinen kommunikaatio päivittäispalavereissa ja valmius muutoksiin mukautumiseen. Ketterät tiimit eivät kuitenkaan aina panosta erilliseen riskienhallintaan – vaikka sillä voitaisiin saavuttaa etuja (Vieira et al., 2020).

Useita malleja on kehitetty parantamaan ketterien kehyksien riskienhallintaa, ja aiheesta löytyy useita tutkimusartikkeleita – mutta niissä esitellyt mallit ovat luonteeltaan kokeellisia tai teoreettisia, eikä niistä ole tarpeeksi käytännön näyttöä. Vieira et al. (2020) painottaa kirjallisuuskatsauksessaan kattavan oppaan puutetta. Tämän tutkielman laajuus ei käsitellyt spesifejä tutkimusmalleja kokonaisuuksissaan. Syynä tähän oli vakiintuneen ja yleisesti käytetyn standardin tai kehyksen puute. Jatkotutkimuksen aiheeksi jää olemassa olevien mallien jatkokehitys niin teoreettisen tutkimuksen tasolla, kuin myös konkreettisten

laajojen kokeilujen muodossa. Tasapainon löytäminen riskienhallintaprosessien raskauden ja ketteryyden välillä on koettu haastavaksi (Elbanna & Sarker, 2016).

6 Yhteenveto

Ketteriä kehyksiä käytetään laajasti ohjelmistokehitysprojektien projektienhallinnassa. Kehysten suhde riskienhallintaan on monimutkainen. Riskienhallintaa voi olla vaikea implementoida ketterään työskentelyyn sillä näiden kahden konseptin visioiden välillä voidaan kokea konflikteja. Tässä tutkielmassa eriteltiin riskienhallintaprosessien yleiset vaiheet seitsemään osaan. Riskienhallinta vaatii aikaa, resursseja, osaamista ja vaivaa. Iso osa riskienhallinnan työmäärästä korostuu projektin alkuvaiheessa, jolloin riskienhallintasuunnitelma tehdään, ja riskejä tunnistetaan sekä analysoidaan.

Ketterät kehykset puolestaan painottavat jatkuvaa ja iteratiivista kehitystä, jatkuvaa yhteistyötä sidosryhmien ja projektitiimin välillä, sekä käytäntöä teorian yli. Käytännöllisyys ilmenee siten että ketterä ajattelu priorisoi nopeaa toimittamista ja mukautuvuutta tarkkojen pitkäaikaisten suunnitelmien sijaan.

On yleistä, että kaikki ohjelmistoprojektitiimit eivät valitse sisältää työprosesseihin erillistä riskienhallintaa, vaikka tutkimukset näyttävät erillisten riskienhallintaprosessien lisäämisestä olevan hyötyjä kuten Vieira et al. (2020) toteavat kirjallisuuskatsauksessaan. Tällaiset riskienhallinnan pois jättävät tiimit luottavat ketterän työskentelyn luontaisten piirteiden vähentävän riskejä tarpeeksi. Ketterällä työskentelyllä ja riskienhallinnalla on kuitenkin myös yhdistäviä tekijöitä, kuten iteratiivisuus. Hankaluutena koetaan riskienhallintamenetelmien työmäärän painotus projektin alkuun ja projektiin lisätty dokumentaatio. Nämä tekijät voidaan nähdä vähentävät projektin ketteryyttä.

Ongelmista huolimatta riskienhallinnan erillinen lisääminen ketterään työskentelyyn on todettu vähentävän projektin mahdollisuutta epäonnistua. Jonkinlaista näyttöä on siitä, että varsinkin laajemmat projektit voivat hyötyä siitä. Implisiittinen riskienhallinta on sopivampi lähestymistapa sellaisissa tapauksissa, joissa projektin koko on pieni, tai riskit ovat pienet. Tutkielma toteaa tarvetta standardille, oppaalle tai mallille, jolla riskienhallintaa voitaisiin sisältää ketterään työskentelyyn. Tämänkaltaisen mallin tavoite olisi pitää lisätyt prosessit tarpeeksi kevyenä, jotta ketterät projektit olisivat halukkaita toimimaan sen kanssa. Tutkielma, ja myös aikaisempien tutkimuksien tulokset viittaavat tarpeelle jatkokehittää olemassa olevia malleja, ja testata niiden implementointia laajemmin.

Lähdeluettelo

Albadarneh, A., Albadarneh, I., & Qusef, A. (2015). Risk management in Agile software development: A comparative study. *2015 IEEE Jordan Conference on Applied Electrical Engineering and Computing Technologies (AEECT)*, 1–6. https://doi.org/10.1109/AEECT.2015.7360573

Bannerman, P. L. (2008). Risk and risk management in software projects: A reassessment. *Journal of Systems and Software*, *81*(12), 2118–2133. https://doi.org/10.1016/j.jss.2008.03.059

Beck, K., Beedle, M., Bennekum, A. van, Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R. C., Mellor, S., Schwaber, K., Sutherland, J., & Thomas, D. (2001). *Manifesto for Agile Software Development*. https://agilemanifesto.org/

Beinschróth, J. (2022). Implementing an effective qualitative risk analysis. *2022 IEEE 10th Jubilee International Conference on Computational Cybernetics and Cyber-Medical Systems (ICCC)*, 000143–000148. https://doi.org/10.1109/ICCC202255925.2022.9922899

Elbanna, A., & Sarker, S. (2016). The Risks of Agile Software Development: Learning from Adopters. *IEEE Software*, *33*(5), 72–79. https://doi.org/10.1109/MS.2015.150

Ghane, K. (2017). Quantitative planning and risk management of Agile Software Development. *2017 IEEE Technology & Engineering Management Conference (TEMSCON)*, 109–112. https://doi.org/10.1109/TEMSCON.2017.7998362

Girvan, L., & Girvan, S. (2022). *Agile From First Principles*. https://learning.oreilly.com/library/view/agile-from-first/9781780175812/

Krasner, H. (2018). *The Cost of Poor Quality Software in the US: A 2018 Report*. https://www.it-cisq.org/the-cost-of-poor-quality-software-in-the-us-a-2018-report/

Lehtinen, T. O. A., & Mäntyla, M. V. (2011). What are Problem Causes of Software Projects? Data of Root Cause Analysis at Four Software Companies. *2011 International Symposium on Empirical Software Engineering and Measurement*, 388–391. https://doi.org/10.1109/ESEM.2011.55

Muntés-Mulero, V., Ripolles, O., Gupta, S., Dominiak, J., Willeke, E., Matthews, P., & Somosköi, B. (2019). Agile risk management for multi-cloud software development. *IET Software*, *13*(3), 172–181. https://doi.org/10.1049/iet-sen.2018.5295

Nyfjord, J., & Kajko-Mattsson, M. (2007). Commonalities in Risk Management and Agile Process Models. *International Conference on Software Engineering Advances (ICSEA 2007)*, 18–18. https://doi.org/10.1109/ICSEA.2007.22

Ohno, T. (1988). *Toyota Production System*. https://learning.oreilly.com/library/view/toyota-production-system/9781000056488/

Pasha, M., Qaiser, G., & Pasha, U. (2018). A Critical Analysis of Software Risk Management Techniques in Large Scale Systems. *IEEE Access*, *6*, 12412–12424. https://doi.org/10.1109/ACCESS.2018.2805862

Pham, A. T., & Pham, D. K. (2017). *Business-Driven IT-Wide Agile (Scrum) and Kanban (Lean) Implementation*. https://learning.oreilly.com/library/view/business-driven-it-wide-agile/9781466557499/

Podari, Z., Arbain, A. F., Ibrahim, N., Abang Jawawi, D. N., Nasir Wan Kadir, W. M., & Fahmi, A. M. (2020). Systematic Literature Review on Global Software

Development Risks in Agile Methodology. *2020 8th International Conference on Information Technology and Multimedia (ICIMU)*, 231–236. https://doi.org/10.1109/ICIMU49871.2020.9243311

Project Management Institute. (2017). *A Guide to the Project Management Body of Knowledge (PMBOK® Guide)—Sixth Edition (ENGLISH)*. https://learning.oreilly.com/library/view/a-guide-to/9781628253900/

Puyt, R. W., Lie, F. B., & Wilderom, C. P. M. (2023). The origins of SWOT analysis. *Long Range Planning*, *56*(3), 102304. https://doi.org/10.1016/j.lrp.2023.102304

Schön, E.-M., Radtke, D., & Jordan, C. (2020). Improving Risk Management in a Scaled Agile Environment. In V. Stray, R. Hoda, M. Paasivaara, & P. Kruchten (Eds.), *Agile Processes in Software Engineering and Extreme Programming* (pp. 132–141). Springer International Publishing. https://doi.org/10.1007/978-3-030-49392-9_9

Schwaber, K., & Sutherland, J. (2020). *Scrum Guide | Scrum Guides*. https://scrumguides.org/scrum-guide.html#purpose-of-the-scrum-guide

Shrivastava, S. V., & Rathod, U. (2017). A risk management framework for distributed agile projects. *Information and Software Technology*, *85*, 1–15. https://doi.org/10.1016/j.infsof.2016.12.005

Shrivastava, S. V., & Rathod, U. (2019). A Goal-driven Risk Management approach for Distributed Agile Development Projects. *Australasian Journal of Information Systems*, *23*. https://doi.org/10.3127/ajis.v23i0.1843

Technical Committee ISO/TC 262. (2018). *ISO 31000:2018*. ISO. https://www.iso.org/standard/65694.html

Teferra, M. (2012). ISO 14971—Medical Device Risk Management Standard. *International Journal of Latest Research in Engineering and Technology (IJLRET)*, *03*(03), 83–87.

Vieira, M., C. R. Hauck, J., & Matalonga, S. (2020). How Explicit Risk Management is Being Integrated Into Agile Methods: Results From a Systematic Literature Mapping. *19th Brazilian Symposium on Software Quality*, 1–10. https://doi.org/10.1145/3439961.3439976

Zhang, S., & Jin, L. (2020). Research on Software Project Schedule Management Method based on Monte Carlo Simulation. *2020 IEEE 5th Information Technology and Mechatronics Engineering Conference (ITOEC)*, 1605–1608. https://doi.org/10.1109/ITOEC49072.2020.9141570

Zhou, L., Vasconcelos, A., & Nunes, M. (2008). Supporting decision making in risk management through an evidence-based information systems project risk checklist. *Information Management & Computer Security*, *16*(2), 166–186. https://doi.org/10.1108/09685220810879636

Zwikael, O., & Sadeh, A. (2007). Planning effort as an effective risk management tool. *Journal of Operations Management*, *25*(4), 755–767. https://doi.org/10.1016/j.jom.2006.12.001