Tampereen ammattikorkeakoulu

Ketterän ohjelmistokehityksen suunnittelu ja käyttöönotto Haltulla

Joonas Särkiniemi

OPINNÄYTETYÖ
Huhtikuu 2024

Tieto- ja viestintätekniikka
Ohjelmistotekniikka

TIIVISTELMÄ

Tampereen ammattikorkeakoulu
Tieto- ja viestintätekniikan tutkinto-ohjelma
Ohjelmistotekniikka

SÄRKINIEMI, JOONAS:
Ketterän ohjelmistokehityksen suunnittelu ja käyttöönotto Haltulla

Opinnäytetyö 59 sivua, joista liitteitä 1 sivua
Huhtikuu 2024

Opinnäytetyön tehtävänä oli valmistaa helposti ymmärrettävä ketterän ohjelmistokehittämisen toimintamalli ja ohjekirja Haltun työntekijöille käyttöönotettavaksi, joka rakentaa toimintaa nykyisten käytäntöjen päälle. Opinnäytetyön avulla etenkin uudet työntekijät voivat ottaa pienellä vaivalla käyttöön ketterän ohjelmistokehittämisen perusteet ilman aikaisempaa tietoa tai kokemusta aiheesta.

Työntekijän perehdyttäminen erilaisiin ketterän ohjelmistokehittämisen malleihin on kallista ja aikaa vievää. Opinnäytetyön tavoitteena oli antaa pohjatiedot ja luoda yksinkertainen toimintamalli, jota hyödyntämällä työntekijän on helppo ottaa mallin menetelmät käyttöön mahdollisimman pienellä kynnyksellä ja ohjata omaa toimintaa yksin tai tiimissä työskennellessä.

Tietoperusta projektinhallinnasta ja erilaisista menetelmistä perustuu ennestään olemassa oleviin ohjelmistokehityksen ja projektinhallinnan toimintamalleihin, jotka ovat laajalti käytössä ohjelmistotuotannon alalla. Tietoperustan läpikäynnin yhteydessä vertaillaan toimintamallien kehitystä ja otetaan retrospektiiviä opinnäytetyön kehittämiseen.

Opinnäytetyö toteutettiin vahvasti yhteistyössä Haltun työntekijöiden kanssa, jotta tuloksena saadaan toimiva malli Haltun toimintaan sopivaksi. Työn määrittely ja osa työn lähteistä perustuu ennestään luotuun kirjallisuuteen, joka on luotu Haltun toimesta.

Tuloksena on Haltun toimintaan sopiva ohjekirja sisäiseen käyttöön, jossa keskeinen tavoitteen täyttyminen on toimintamallin toteuttamisen käyttöönoton helppous ja oman työn rakenteellisuuden parantaminen, joka näkyy projektinhallinnan tasolla projektityöskentelyn etuna. Lopulliset tulokset opinnäytteestä saadaan useamman erityyppisten projektien loputtua, mutta johtopäätöksenä kehitetty malli soveltuu paremmin Haltun toimintaan ennalta olevien mallien sijasta.

Asiasanat: ohjelmistokehitys, ketterät menetelmät, scrum, projektinhallinta

ABSTRACT

Tampereen ammattikorkeakoulu
Tampere University of Applied Sciences
Degree Programme in Software Engineering

SÄRKINIEMI, JOONAS:
Agile Software Development Planning and Deployment at Haltu

Bachelor's 59 pages, appendices 1 pages
April 2024

The purpose of this thesis was to create an easily understandable agile software development framework and manual for Haltu's employees that builds procedures on top of the existing ones. With the help of the framework, new employees can learn the basics of agile software development without any previous knowledge or experience in the subject.

It is expensive and time consuming to introduce new employees to different agile software development models. The goal of this thesis was to give basic knowledge and create a simple framework to help employees put the different methods into practice with as low a threshold as possible and guide their own work when working alone or in a team environment.

The scientific knowledge is based the already existing literature that is already in use in software engineering. While going through the history and development of current practices, they are used to compare and develop the practices proposed in the thesis.

The thesis was done in cooperation with Haltu's employees to create a working framework to fit Haltu's existing practices. The specification and some of the sources used were based on the existing literature created within Haltu.

The result of this thesis is a fitting guide to be used within Haltu wherein the main goal is to easily take up the written practices and apply them to current project work. The thesis improves work structure and procedures which can prove beneficial to project management. In conclusion, the created framework suits Haltu's needs better than existing frameworks.

Key words: software development, agile, scrum, project management

SISÄLLYS

LYHENTEET JA TERMIT

UX	User Experience
PERT	Program Evaluation and Review Technique
PMI	Project Management Institute

1 JOHDANTO

Tämä kehitysprojekti on Haltun kanssa yhteistoimin toteutettu sisäinen suunnitteluprojekti. Tarkoituksena oli kehittää Haltun tiimityöskentelyä, sisäistä kommunikaatiota ja yleistä projektin hallintaa työtekijöiden kesken.

Työssä ei lähdetty keksimään uutta projektihallinnan viitekehystä alusta alkaen, vaan ottaa käyttöön jo ennestään olemassa oleva ketterä kehitysmenetelmä käyttöön muokkaamalla sitä sopimaan Haltun toimintatapoihin.

Ketteriä kehitysmenetelmiä on kehitelty ja tutkittu vuosikymmeniä, joten ei ole olennaista käyttää saman verran aikaa uuden viitekehyksen luomiseen. Käytännön tasolla aikaisemmat yritykset ottaa käyttöön ketterä ohjelmistokehittämisen malli on jäänyt puutteelliseksi mallin käytäntöjen puutteeseen.

Opinnäytetyöllä lähdetään ratkomaan ongelmia ja esteitä, joiden takia aikaisemmat yritykset eivät ole jääneet käytäntöön. Ennestään toteutettuja menetelmiä on ainoastaan tarkoitus soveltaa Haltun omiin käytäntöihin. Sovelletuksi menetelmäksi sovittiin hyödyntää ketteriä kehitysmenetelmiä, joista Scrum-projektinhallintamenetelmä soveltuu Haltun käyttötarkoitukseen parhaiten.

Haltun sisäisen vapaaehtoisuuteen perustuvan kokouksen myötä Scrum--projektinhallintamenetelmää päätettiin alkaa kehittämään työryhmän avulla, johon työntekijät pystyivät liittymään vapaaehtoisina ja siten vaikuttamaan projektin lopputulokseen. Projektin johtoon valittiin Haltun teknologiajohtaja valvomaan työn laatua ja edistystä. Teknologiajohtaja ei osallistunut kehitykseen, koska projektin tarkoituksena oli luoda työntekijöiden suunnittelema projektinhallinta menetelmä, joka soveltuu heille parhaiten. Kehitetty malli otettiin käyttöön kokeilun avulla, jossa työntekijät pääsivät toimimaa luotujen käytäntöjen pohjalta ja antamaan palautetta kirjoitetuille ohjeille.

Keskeisin työryhmän tavoite oli saada olemassa oleviin projekteihin projektijohtaja, joka poistaa ja ratkoo esteitä kehittäjiltä, jotta he voivat keskittyä olennai-

seen. Projektijohtaja tietää olennaisen projektista, johtaa projektiryhmän työskentelyä ja huolehtii projektin tapahtumien pitämisestä, aikatauluttamisesta sekä kommunikaatiosta asiakkaan kanssa. Kuka tahansa projektin jäsenistä tai sen ulkopuolelta voi toimia projektijohtajana luodun menetelmän pohjalta.

2 HALTU OY

Haltu on keskisuuri yritys, jossa on noin 25 työntekijää laajalta ohjelmistokehityksen osaamisalueelta. Haltun perustaja on Mikko Savilahti yhdessä teknologiajohtaja Ilkka Hakkarin kanssa, jotka perustivat Haltun yhdessä vuonna 2010. Toimitusjohtajana toimii Mikki Aalto-Ylevä, joka toimi ennen myyntipäällikkönä. Haltu Oy:n liikevaihto vuonna 2021 oli 1,4 miljoonaa, josta tilikauden tulos 30 000 euroa.

Haltu Oy on ohjelmistotalo Tampereella, joka erikoistuu ohjelmistoratkaisuihin eri kohderyhmille monenlaisiin tarpeisiin. Haltu tarjoaa ketterää ohjelmistokehitystä enimmäkseen yksityisille toimihenkilöille ja julkiselle sektorille. Tarjontaan kuuluu ohjelmistokehitystä, verkkopalveluiden ylläpitoa, erilaisia strategiapalveluita ja UX-palveluita.

Haltu sopeutuu tarpeen mukaan asiakkaan tai jo olemassa olevan projektin tarpeisiin tehdä projektin toteutus erilaisilla teknologioilla. Projektit toimivat Haltun suunnitteleman palvelinympäristön päällä, jonka ansiosta projekteja on helppo pitää ajan tasalla ja käynnistää kehitysympäristö kustannustehokkaasti. Tavallisen ohjelmoinnin lisäksi Haltu tarjoaa tukea yritysrahoituksen suunnittelussa, graafisessa suunnittelusta, saavutettavuuden arvioinnissa, käännösten tekemisessä, palvelumuotoilussa ja ylläpidossa.

Oletuksena Haltu tarjoaa ohjelmistokehityksen toimituksen kokonaisuudessaan ja auttaa asiakasta tekemään ohjelmistokehitykseen liittyviä päätöksiä, koska yksityisasiakkailta ei voi odottaa täyttä tietämistä ohjelmistokehityksestä. Yksityisasiakkaiden lisäksi Haltun asiakaskunta koostuu paljon kuntapuolen toteutuksista ja muista kuntapuolen sektorin toimijoista, pienistä startup-yrityksistä, koulutusalan toimijoista ja yrityspuolen asiakkaista. Lisäksi Haltulla on omia tuotteita, kuten huoltamoiden hallintasovellus Huoltotasku ja koulujen Velmu-työpöytä.

3 PROJEKTIN KÄYNNISTÄMINEN

Päätös kehittää Haltulle ketterää ohjelmistokehitystä lähti alun perin Haltun sisäisestä kokouksesta. Haltulla kokoontuu muutamaan otteeseen kuukaudessa ryhmä, jota kutsutaan Viisikoksi. Ryhmän tarkoituksena on edustaa Haltun työntekijöitä ja päättää työntekijöiden toivomista muutoksista sekä kehittää Haltun toimintaa. Viisikkoon osallistuminen on vapaaehtoista ja nimensä mukaisesti siihen kuuluu viisi henkilöä. Useamman vapaaehtoisen kuin viisi tapauksessa järjestetään perinteiset vaalit, jossa eniten ääniä saaneet valitaan osallisiksi.

Ketterän ohjelmistokehityksen projektiryhmä perustui myös vapaaehtoisuuteen. Kaikki halukkaat pääsivät mukaan osallistumaan projektiin, jos oli kiinnostusta lähteä kehittämään Haltulle toimintatapoja. Projektiryhmään osallistui viisi henkilöä kehittämään ohjelmaa. Laadun ja Haltun asiakkuuden puolesta projektiryhmään osallistui Haltun teknologiajohtaja Ilkka Hakkari valvomaan toteutusta koskematta itse sisältöön. Osallistuneesta viidestä henkilöstä neljällä on ohjelmiston kehittämisen taustaa, mutta ryhmään haluttiin myös kehitystiimin ulkopuolelta osallinen, joka voisi tuoda omaa näkökulmaa projektiin. Tarkoituksena oli kuitenkin, että kehitetyn menetelmän pohjalta kuka tahansa voisi toimia projektijohtajana.

3.1 Projektin toimintatavat

Projektiryhmää oli tarkoitus kohdella, kuin mitä tahansa normaalia projektiryhmää ohjelmistoprojekteissa, mutta poikkeuksena sitä pyöritettiin ketterällä ohjelmistokehityksen menetelmällä Scrumilla. Tämä tarkoitti sitä, että projektilla oli aikataulu, budjetti, asiakas ja projektijohtaja, joka fasilitoi tapahtumat.

Budjettia oli kuudelta henkilöltä käytettävissä kahden tunnin verran viikoittain. Tämä huomioon ottaen päätettiin työn kestävän kuusi kuukautta ja tapaamisia olevan kerran viikossa. Jokaiselle viikolle annettiin agenda ja tapaamisista jäi tehtäväksi työajalle toinen tunti projektin edistämiseen. Tapaamiset toistuivat

samana ajankohtana kerran viikossa ja Haltun toimitiloissa paikallaolijat kokoontuivat kokoushuoneeseen ajan salliessa. Toimiston ulkopuolella olevat työntekijät osallistuivat kalenterikutsun kautta palaveriin.

Projektia sovittiin pyöritettävän ketterän ohjelmistokehityksen Scrumin tavalla, joten projektille tarvittiin projektijohtaja, joka tapahtumista ja kokoontumisista. Projektiryhmän jäsenistä lähdin projektijohtajan rooliin viemään projektiryhmää eteenpäin kohti tavoitetta.

Ensimmäisen kuuden kuukauden aikana projektiryhmä sai vastuuksi kerätä substanssia tulevaan koulutusohjelmaan eli suunnitellaan, mitä sisältöä halutaan ja millä tavalla se kirjoitetaan, jotta siitä tulee mahdollisimman ymmärrettävä projektin ulkopuolisille henkilöille. Lisäksi kartoitettiin hyötyjä ohjelmasta ja käsiteltiin mahdollisia esteitä ottaa Haltun oma ketterä ohjelmistokehityksen menetelmä käyttöön.

4 PROJEKTINHALLINNAN TEORIAA

Projektinhallinta on tapa johtaa ryhmän työskentelyä ja ryhmän sisällä olevia tahoja kohti yhteisesti asetettua tavoitetta onnistuneesti ryhmän asettamien ehtojen mukaisesti. (Phillips 2003, 354) Projektin käynnistettyä nämä yhteiset tavoitteet, rajoitteet ja ryhmän koostumus tulee olla listattuna projektin dokumentaatioon, mikä antaa pohjan projektin onnistumiselle. (Westland 2022.)

Projektinhallinnan päämäärä on luoda onnistunut tuote, joka noudattaa valmiin määritelmää asiakkaan tai tuoteomistajan näkökulmasta. Varsinkin abstrakteissa toteutuksissa projektinhallinnalla on tavoitteena muodostaa yhteys asiakkaaseen, jotta on mahdollista luoda yhteisesti ymmärretyt tavoitteet projektille, jotka ovat mahdollista toteuttaa annettujen rajoitteiden sisällä. (PMI n.d.)

Jokainen ryhmän sisällä tehty päätös tulisi tehdä projektin tavoitteet mielessä, jotta päätökset vievät projektia onnistumista kohti. Jokainen asetettu rajoite ja tavoite projektille vaikuttaa päätöksentekoon ja projektissa työskentelevään ryhmään. (AdaptiveWork 2021.) Rajoitteita projektille on budjetti, aikataulu ja laajuus. Liian tiukat ja vääränlaiset rajoitteet haittaavat päätöksentekoprosessia ja ryhmän työskentelyä projektin sisällä. (Angelo 2006, 1–2)

Projekti on väliaikainen tehtävä tai prosessi, jolla on määritelty alku sekä loppu, yleensä tietyllä määritetyllä aikavälillä, budjetilla ja työntekijöillä. Projektin tehtävä on tuottaa jonkinlainen tuote tai prosessi, joka on valmis sille asetetun valmiin määritelmän toteutettua. Tämänlainen tuote tarvitsee hallintaa, strategiaa ja mahdollisesti projektin luonteen mukaan teknisiä taitoja omaavaa johtoa. (Cattani, Ferriani, Frederiksen & Florian 2011.)

4.1 Historia

Ennen projektinhallinnan kehystä, projekteja johdettiin usein sosiaalisen taitojen omaavien tai vaadittujen teknisten taitojen omaavien henkilöiden toimesta, ku-

ten insinöörien ja arkkitehtien kautta. Projektinhallinnan kehittäminen onkin lähtenyt monien eri alojen tarpeesta 1900-luvulla. (Kwak 2021.) Projektinhallinnan esi-isinä voidaan pitää Henry Ganttia ja Henri Fayolia, jotka toimivat alan insinööreinä. He molemmat tutkivat Frederick Winslow Taylorin teorioita projektinhallinnan tieteellisestä näkökulmasta. Merkittävimpänä pohjana tieteelliselle näkökulmalle pidetään työn allokoinnin kehystä, jossa tehtäviä rikotaan pienempiin osa-alueisiin, jotta projektiryhmän työskentelyä voidaan jakaa kokonaisuuksiin, joita on helpompi hallita. Hankkeen osituksen lisäksi tieteellinen näkökulma on luonut pohjan resurssien allokoimiselle projektin sisällä. (Firsthand n.d.)

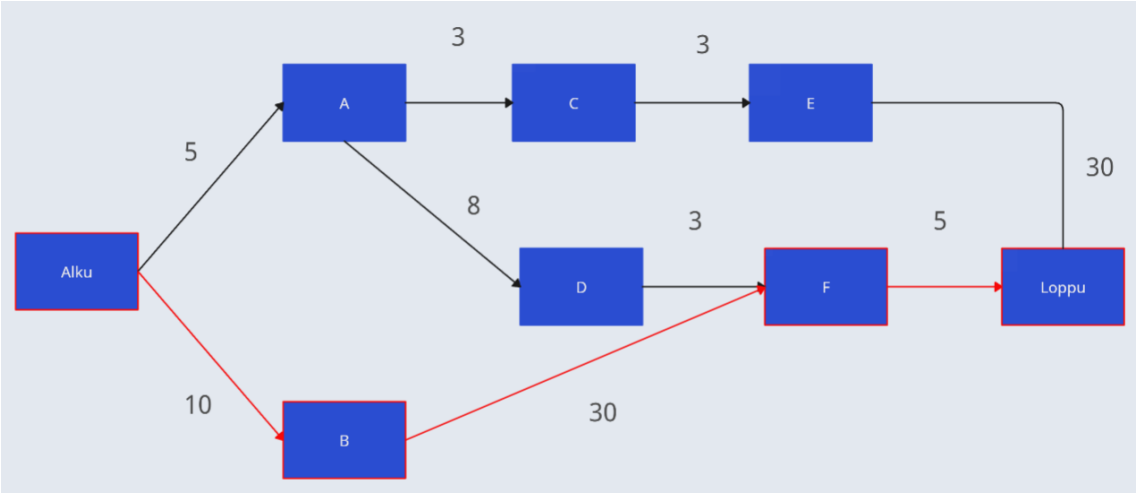
Modernia projektinhallinnan kautta voidaan pitää 1950-vuodesta alkaen, jolloin eri alojen insinöörit alkoivat liittoutua ja projektinhallinnan kehys tunnistettiin yhtenä kokonaisuutena osaa sääntöjä noudattavaa käyttäytymistä ja insinöörien prototyyppimallia. (Cleland & Roland 2005, 25) Ennen modernia aikakautta projektinhallintaa käytettiin vain tarpeeseen eikä suunnitellusti läpi projektien. Käytössä olevat mallit kriittinen polku ja PERT-malli pohjautuivat matematiikkaan ja algoritmeihin. (Azzopardi n.d.)

Kriittisessä polussa ajastetaan projektin tehtäviä ja vaiheita tunnistamalla pitkäaikaiset tehtävät, niiden riippuvuus toisistaan ja mittaamalla ajallisesti tehtävien kesto alusta loppuun. Luettelemalla kaikki tehtävät ja niiden ajalliset riippuvaisuudet voidaan laskea pisin polku projektin alusta loppuun sekä tunnistaa ajallisesti, milloin yksittäiset muista riippumattomat tehtävät voidaan ottaa työstettäväksi, jolloin niistä ei koidu ajallisesti haittaa muille tehtäväkokonaisuuksille, jotka voivat olla riippuvaisia toisistaan. (Baker 2010)

Kriittisen polun mallissa mallinnetaan projektin tehtävien väliset riippuvuudet eli mikä tehtävä on suoritettava ennen toisen aloittamista. Riippuvuuksien jälkeen mitataan tehtävien kesto, jotta voidaan laskea jokaisen polun kesto lisäämällä tehtävien summa. Isoin summa on projektin kriittinen polku. (Levy & Wiest 1963, 1)

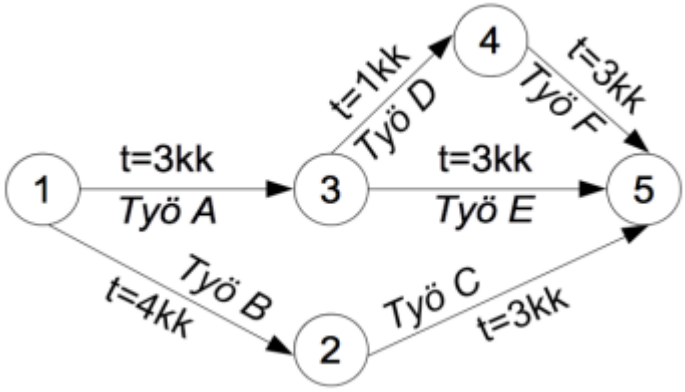
Kriittistä polkua käytettiin usein PERT-mallin kanssa, joka on statistiikkatyökalu projektinhallinnassa, joka analysoi ja esittää projektiin sisältyvät tehtävät. Mallin pohjana käytetään tehtävien arviointia pienimmän mahdollisimman aikavälin

mukaisesti, joka mahdollistaa projektin aikatauluttamisen jäntevästi ilman tarkkaa aikataulua. PERT-malli on tehtäväpohjainen näkökulma projektinhallintaan ajallisen alun ja lopun sijasta, joka soveltuu paremmin abstrakteihin isoihin projekteihin, jossa aikataulu on suurempi rajoite kuin hinta. (Miller 1962, 93)



KUVIO 1. Kriittisen polun toimintakaava.

PERT-malli keskittyy ajan suunnitteluun ja ajan hallinnoimiseen, jonka tarkoituksena on saavuttaa projekti loppuun mahdollisen nopeasti. (Miller 1962, 94)



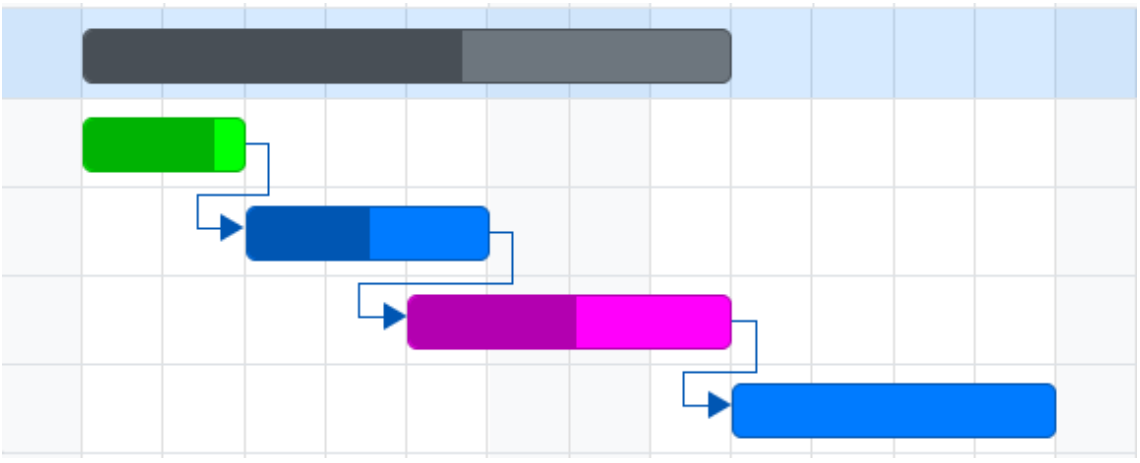
KUVIO 2. PERT-mallin toimintakaava (Eriksson 2023)

1950-luvun jälkeen merkittävin projektihallinnan teoria kustannusten hallinnointi on yhdysvaltalainen malli, jossa hallitaan ja ennustetaan projektin kustannuksia. Tavoitteena on optimoida projektin sijoituksia ja saavuttaa tasapainoinen suhde kustannusten, laadun ja ajan kanssa. Periaatteen tärkein osa on arvioida tarkasti tehtävien kustannuksia ja aikataulusta, jotta poikkeamia ei tapahdu. Kustannusten arviointi on suunniteltu helpottamaan päätöksentekoa ja auttamaan

projektia hallitsemaan resursseja. Sitä voidaan pitää vanhanaikaisena insinööritaitona keskittyä työn hinnan ja tehtävän vaatimaan käsityötaidon suhteeseen. Vuonna 1969 Yhdysvalloissa perustettiin projektihallinnan instituutti PMI, joka on julkaissut projektihallinnan ohjekirjan vuonna 1996, mikä käsittelee yleisempiä käsitteitä projektinhallinnassa ja käy läpi yhteisiä asioita, jotka toistuvat projektista toiseen sen luonteesta tai tyypistä huolimatta. (Haughey 2010.)

4.2 Projektinhallinnan esi-isät

Henry Gantt on tunnettu projektinhallinnan tekniikoista ja projektin suunnittelun kehittämisestä, josta tunnetuimpana työkaluna on janakaavio, joka nimettiin hänen mukaansa Gantt-kaavioksi. Gantt tutki projektinhallintaa tieteellisestä näkökulmasta, jonka pohjalta toteutettiin projektinhallintaa hallitsemalla olemassa olevia resursseja sekä jäljellä olevaa aikaa, jonka kautta arvioidaan projektin tarvitsema allokaatio. Gantt julkaisi paljon kirjallisuutta elämänsä aikana ja siirtyi myöhemmin elämässänsä tieteellisestä näkökulmasta projektinhallintaan sosiaalisesta näkökulmasta, jossa keskityttiin työntekijöiden tyytyväisyyteen. Tieteellisestä näkökulmasta Ganttin suurimpana perintönä projektinhallintaan on Gantt-kaavio. (Firsthand n.d.)



KUVIO 3. Gantt-kaavion esitysmalli.

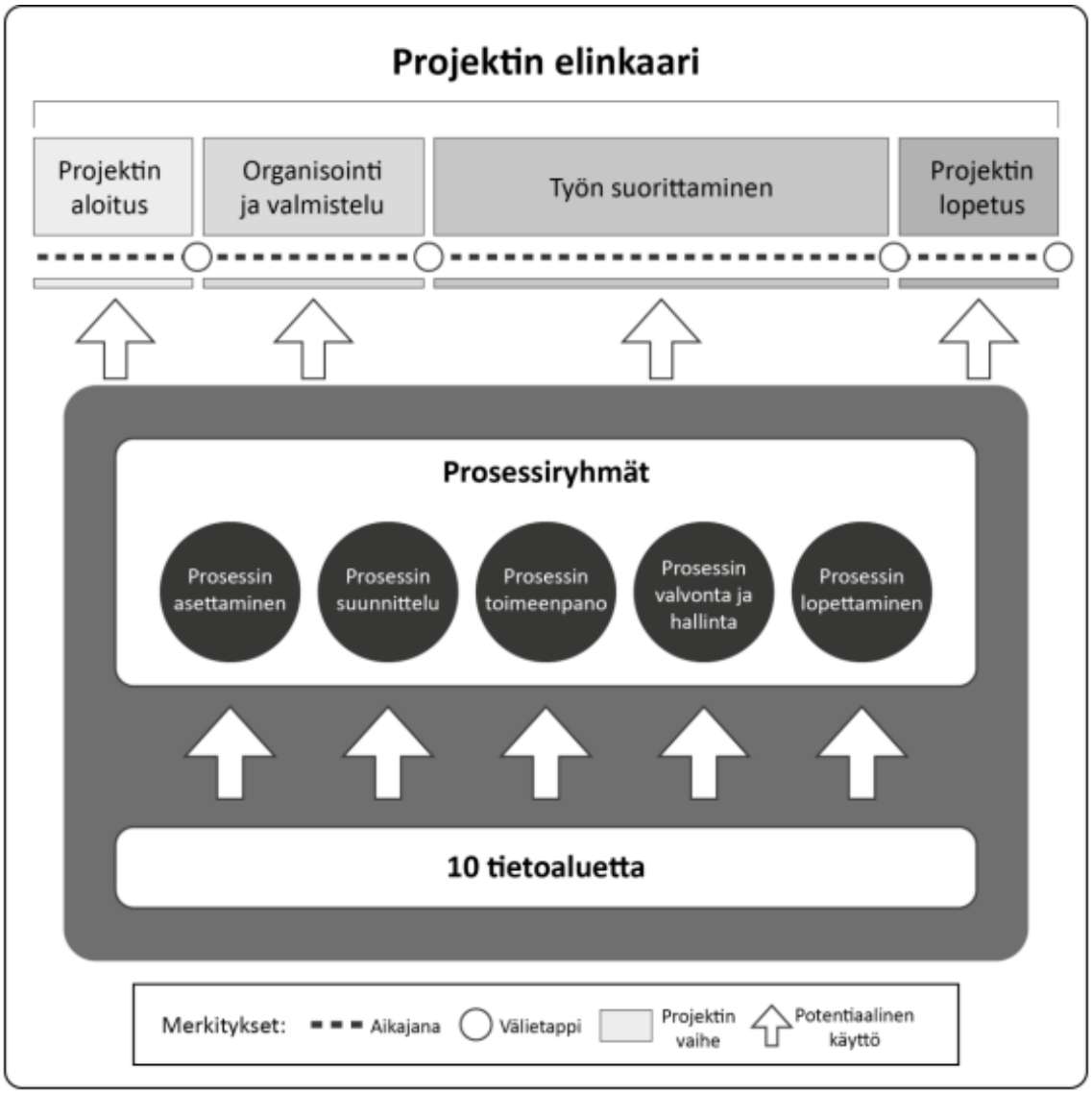
Kaavion lisäksi Ganttin kirjoitti tuotannon tehokkuuden teoreettinen analysoinnista, jossa tieteellistä analyysiä käytetään kaikkiin projektityön kokonaisuuksiin,

jonka tarkoituksena on eliminoida vahingot ja sattumat. Myöhäisemmässä elämässä Gantt keskittyi sosiaaliseen näkökulmaan projektinhallinnassa ja loi bonusjärjestelmän, jossa palkittiin projektinjohtajat heidän työntekijöidensä osaamisen ja tietotaidon parantumisen mukaisesti. Bonusjärjestelmä johti juurensa osittain hänen näkökulman muutoksensa takia, jossa hän uskoi yritysten sosiaalisesta vastuusta ympäristöön ja työyhteisöön. (Sheldrake 2003, 35–43.)

Henri Fayol loi yhden ensimmäisistä syvemmän teorian projektinhallinnan johtamisesta, fayolismin. Henri Fayolin mukaan on kuudenlaista järjestötoimintaa, viidenlaista johtamisen periaatetta ja 14 johtamiseen liittyvää periaatetta. Fayolismi on johtamisen teoria, jolla tutkitaan johtamisen roolia organisaatiossa. Teorian perusidea on kehittää johtamisen käytäntöjä, jolla voidaan vähentään väärinkäsityksiä ja lisätä tehokkuutta. Lyhykäisyydessään Fayolismi on antanut suunnan, miten projekteja ja yrityksiä hallitaan nykyään, miten työntekijöitä johdetaan johdonmukaisesti, reilusti ja miten yritystä voidaan jakaa eri hallinnollisiin osiin. (Witzel 2003, 96)

5 PROJEKTIN ELINKAARI

Projektin elinkaari voidaan jakaa viiteen eri vaiheeseen prosessien mukaisesti. Jokainen prosessi liittyy toisiinsa ja odottaa aikaisemman prosessin valmistumista, jotta voidaan edetä seuraavaan prosessiin. Prosessit voidaan jakaa projektin käynnistymiseen, projektin suunnitteluun, toteuttamiseen, laadun valvonta ja projektin päättymiseen. Prosessien lisäksi projektia voidaan jakaa myös tuotannon tai tuotteen näkökulmasta. (Wysocki 2013, 42)



KUVIO 4. Projektin elinkaari (A Guide to the Project Management Body of Knowledge Sixth Edition 2017, 18, muokattu)

Projektin käynnistys on tärkein prosessi, koska siinä määritellään projektin luonne ja sen laajuus. Huonosti käynnistetty projekti todennäköisesti ei tule onnistumaan sille asetettujen rajojen sisässä. Käynnistämisen yhteydessä tulisi luoda projektin dokumentaatio, josta käy ilmi tärkeimmät aiheet, kuten projektin laajuus, rahoitus, suunnitelma, tuotteen kannattavuus ja rahoitus. Mahdolliset epäkohdat ja virheet on mahdollista korjata ensimmäisessä vaiheessa suhteellisen kivuttomasti.

Projektin käynnistämisen jälkeen edetään suunnitteluun, jossa arvioidaan projektille aikataulu, budjetti ja resurssit työn edistymisen kannalta. Lisäksi projektille tehdään riskianalyysi sen toteutumisen kannalta, jotta riskejä voidaan ennaltaehkäistä projektin edistyessä. Hyvin tehty suunnitelma nostaa projektin onnistumisen mahdollisuuksia. Jos projektille on mahdollista asettaa mittareita, joita voidaan mitata järjestelmällisesti, tulisi mittarit asettaa tässä kohtaa projektin elinkaarta.

Suunnitelman hyväksymisen jälkeen edetään toteuttamiseen. Projektinhallinnan kannalta on ehdottoman tärkeää, että projektissa noudatetaan sille asetettuja tavoitteita ja rajoja. Toteutuksen tärkeimmät toimet ovat työn allokointi, kommunikointi ja henkilöstöhallinto muiden resurssien ohella, kuten budjetti tai materiaalit. Toteuttamisen vaiheessa työn on tarkoitus konkretisoitua eli työn tuloksena tuotetaan jotain. Projektin dokumentointi tässä vaiheessa on iso osa projektin onnistumista, jotta projektin laajuus, budjetti ja aikataulu pysyvät onnistumisen rajoissa. Hyvän dokumentaation ansiosta voidaan huomata projektia uhkaavat ongelmat etukäteen ennen uhkakuvien oikeaa toteutumista. Dokumentaatio on ainut työkalu valvoa projektin historiaa ja antaa viitekehystä toteutuneelle työlle, josta näkyy mitä toteutuksen vaiheessa on tuotettu.

Laadun valvonta seuraa projektin toteutusta ja huomaa projektia uhkaavat ongelmat ennenaikaisesti, jotta niihin voi reagoida hyvissä ajoin. Ajallinen laadun valvonta huomaa muutokset työssä, jotka voivat vaikuttaa projektin aikatauluun ja budjettiin. Valvontaan kuuluu hallinta toteutuksessa olevasta työstä, työn aikataulusta ja budjetista verrattuna alkuperäiseen suunnitelmaan. Epäkohdan löydyttyä tehdään suunnitelma, miten alkuperäiseen suunnitelmaan voidaan palata, jotta projekti voi onnistua. Laadun valvonta on jatkuva prosessi, joka tukee

loppukäyttäjää, korjaa epäkohtia ja päivittää tuotetta haluttuun suuntaan ajan kuluessa.

Viimeisenä prosessina on projektin lopetus. Projektin päättyminen on virallinen sopimus osapuolten välillä toteutuksen lopettamisesta. Kaikki projektiin liittyvä työ on vietävä loppuun ja virallisesti sulkea projekti. Projektin lopetukseen kuuluu retrospektiivi projektin kulusta, jotta osapuolet voivat oppia kokemuksesta ja suoriutua tulevaisuuden projekteista paremmin. (Kissflow 2023.)

5.1 Projektijohtaja

Projektin elinkaaren kannalta on tärkeää, että projektilla on yksi projektinjohtaja. Projektinjohtajan tulee olla ammattilainen alallaan ja on vastuussa ihmisistä projektin sisällä. Onnistumisen kannalta on tärkeää, että projektin sisälle valitaan oikeat ihmiset. Projektinjohtaja valvoo projektin budjettia, työn laatua, aikataulua sekä elinkaaren prosesseja ja niiden toteutumista.

Projektinjohtajan tulee ymmärtää projektin vaatimukset, rajoitukset ja valmiin määritelmä, jotta aikataulussa voidaan pysyä ja tehdä oikeita valintoja projektin onnistumisen kannalta. Usein projektinjohtaja on asiakkaan edustaja, joka ymmärtää ja huolehtii asiakkaan tarpeista. Joustavuus ja ihmissuhteiden luonti ovat tärkeitä ominaisuuksia hyvälle projektinjohtajalle. (U.S. Office of Personnel Management 2003.)

6 KETTERÄT KEHITYSMENETELMÄT

Ketterät kehitysmenetelmät ennakoivat tarpeen joustavuudelle, ovat iteratiivisia ja rakentavat tuotteelle vaatimukset yhteistyössä asiakkaiden ja käyttäjien kanssa. Joustavat suunnitelmat tehdään usein yhteistyössä ja toteutunut työ on evolutiivista. Ketterä projekti vaatii ryhmän jäseniltä oman työn organisoimista ja yhteistyökykyisyyttä sidosryhmien kanssa. Ketterä projekti pyrkii aikaiseen julkaisuun, jatkuvaan kehitykseen, joustavaan muutokseen suunnitelmiin tai vaatimuksiin ja pyrkii parempaan ymmärrykseen projektista ongelmien ratkaisua varten. Ketterillä menetelmillä on neljä periaatetta, jotka kuvaavat mallien toimintaa. (Agilealliance n.d.). Neljä periaatetta on määritelty The Agile Manifesto-teoksessa.

Yksittäiset vuorovaikutukset ovat tärkeämpiä kuin prosessit ja työkalut. Ihmiset ajavat tuotantoa ja vastaavat yrityksen tarpeisiin. Projektin sisällä työskentelevät ihmiset ovat kaikista tärkein osa projektia ja johtavat tuotantoa. Jos prosessit tai työkalut johtavat tuotantoa, projekti ei todennäköisesti pysty vastaamaan muutoksiin tarpeeksi joustavasti.

Tavoitteena keskittyä tuottaa toimivaa tuotetta kuin läpikohtaista dokumentaatiota. Aikaisemmin iso osa käytetystä ajasta kului dokumentaation kirjoittamiseen tuotannon aikana. Dokumentaation vaatimukset hidastavat tuotantoa ja vaatii paljon aikaa. Ketterät menetelmät eivät poista dokumentaation tarpeellisuutta, vaan yksinkertaistavat dokumentaation keskeisiin toiminnallisuuksiin, joita vaaditaan työn suorittamiseen.

Projektia tehdään yhteistyössä erilaisten neuvottelujen sijasta. Tarkoituksena on keskittyä yhteistyöhön projektinjohtajan ja asiakkaan kesken. Yhteistyö asiakkaan kanssa vaatii jatkuvaa ylläpitoa läpi kehityksen, joka auttaa projektin suunnitelmien muutoksessa ja sovittujen asioiden ylläpidosta.

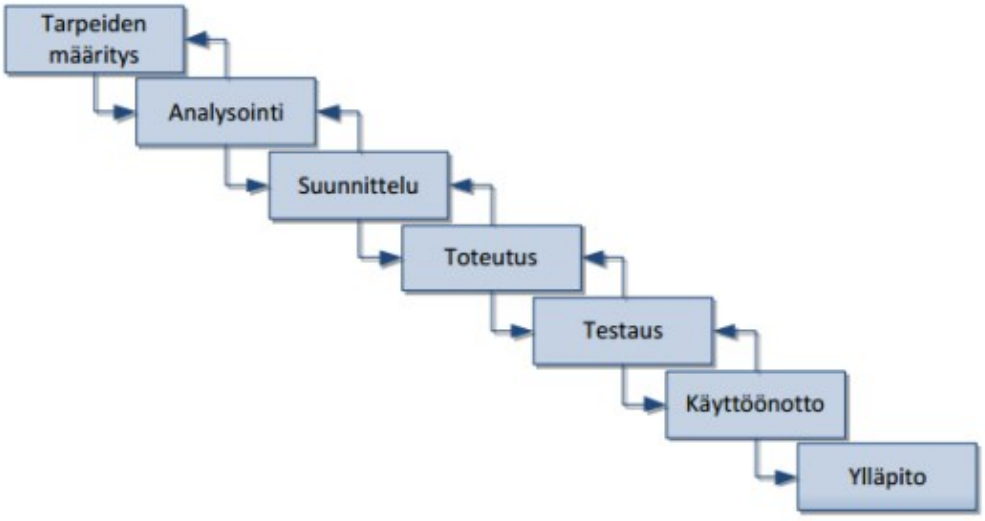
Reagoidaan muutoksiin. Aikaisemmin muutokset koettiin kulueränä, joten muutoksilta yritettiin välttyä kaikin mahdollisin keinoin. Ketterä ajattelu purkaa tätä

ideologiaa. Kehitystä tehdään pienissä iteraatioissa, mikä mahdollistaa muutoksien tekemisen aikaisessa vaiheessa, jotta projektiryhmä voi tehdä muutoksia prosesseihin tai suunnitelmiin. Muutos koetaan parannuksena projektin tuotteeseen ja antaa lisäarvoa tuotteelle.

Edellä mainittuja arvoja sekä käytäntöjä käytetään pohjana useissa ketterän kehityksen menetelmissä, jotka ovat saaneet suosiota 2000-luvun alusta. Ketterille kehitysmenetelmille on vaikea asettaa mittareita, joten empiiristä todistusaineistoa sen toimivuudesta on vaikea löytää, mutta menetelmät nauttivat suurta suosiota ja todistusaineistoa toimivuudesta henkilökohtaisen seurannan tasolla.

6.1 Historia

Iteratiivinen ohjelmistokehittäminen on saanut juurensa 1950-luvulta, mutta joustava ohjelmistokehittäminen nousi suosioon 1970-luvun alkupuolella. Joustava malli on iteratiivista kehittämistä, joka on hiljattain korvannut vesiputous syklin. Vesiputousmalli on syklistä kehittämistä lineaarisissa vaiheissa, jossa jokainen vaihe on riippuvainen toisestaan. Voidaan kuvitella, että tehtävät putoavat vesiputouksen tavalla ylhäältä alaspäin, jossa seuraava tehtävä alkaa ensimmäisen loputtua. Vesiputousmalli on alun perin rakennusinsinöörien käyttämä, mutta mukautui myöhemmin ohjelmistokehittämiseen muiden tietopohjaisten mallien puutteen takia. (Sacolick 2022.)



KUVIO 5. Vesiputousmalli kuvattuna (Haikala & Märijärvi 2006).

1990-luvulla ohjelmistokehittämisen mallit alkoivat kehittyä köykäisempiin pienempiin malleihin vesiputousmallista. Vesiputousmallia pidettiin liian säädeltynä, suunniteltuna ja liiallisena työn kontrollointina. Kritiikin pohjalta moni ketterä malli sai alkunsa, kuten nopean kehittämisen malli, yhteisen prosessin malli, dynaamiset järjestelmät ja Scrum. Lean ajattelua, josta ketterä projektinhallinta on saanut alkunsa ja Scrum-mallia voidaan pitää merkittävimpänä teorioina ohjelmistokehittämisen tulevaisuuden kannalta. (Lynn n.d.)

Uusien mallien suunnittelijat tapasivat vuonna 2001 yhteiseen kokoukseen keskustelemaan ketterästä kehittämisestä. Yhdessä he julkaisivat ketterän ohjelmistokehityksen julistuksen, josta ketterän ohjelmistokehittämisen neljä periaatetta saivat alkunsa. (Lynn n.d.)

6.2 Scrum

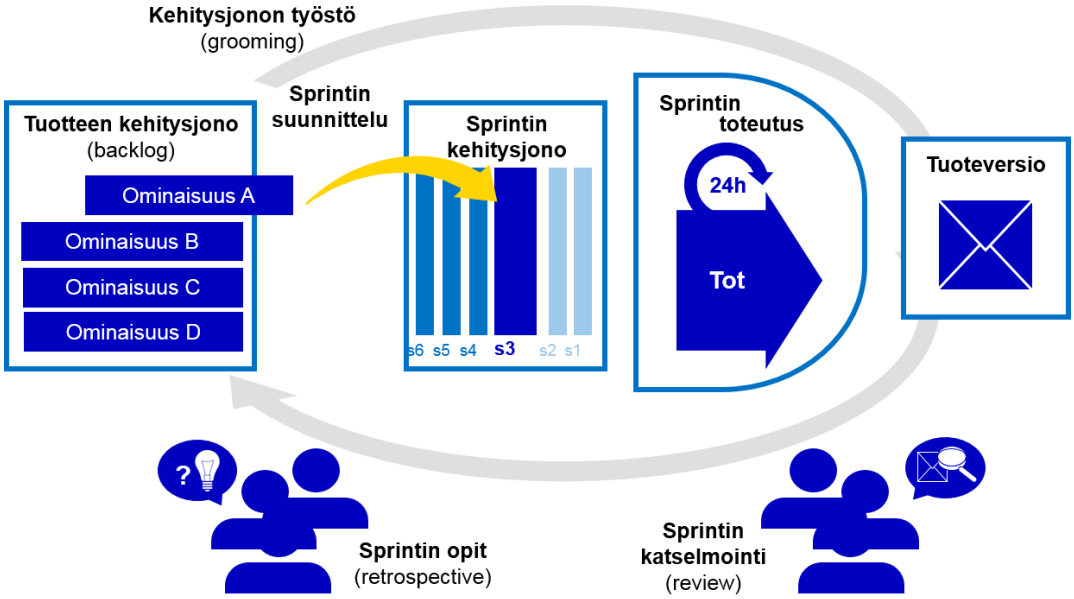
Scrum on ketterä kehitysmenetelmä, jolla voidaan ratkaista hankalia ongelmia, sopeutua tilanteeseen tarvittaessa ja samalla tuottaa tuloksekkaasti laadullisia tuotteita korkealla arvolla. Scrum on helppo työskentelyn kehys, joka luo sopeutuvia ratkaisuja. Scrum toimii parhaimmillaan pienissä, alle kymmenen hengen tiimeissä. Nimi tuleekin amerikkalaisesta jalkapallosta, jossa aloitukseen kerääntyy joukko pelaajia. Sen on keksinyt Hirotaka Takeuchi ja Ikujiro Nonaka, jotka käyttivät termiä ohjelmistokehityksessä ensimmäisen kerran Harvardin hallinnollisessa lehdessä vuonna 1986 (Knowledgehut 2023.)

Menetelmän on tarkoitus olla helppo ottaa käyttöön. Sen voidaan kuvitella olevan vastakohta isoille tuotekehityksen ratkaisuille, joissa elementit ja käytännöt ovat pakollisia. Scrumin toteutus lähtee yksilöistä. Se antaa arvoa ja ymmärtää ihmisen omia tarpeita sekä itsensä kehittämistä. (Knowledgehut 2023.)

Scrum-mallia pyöritetään jatkuvasti toistuvissa pienissä uudelleen toistuvissa iteraatioissa, jotka kestävät usein viikosta kahteen viikkoon, mutta aikaväliä ei ole tarkoitus asettaa tietylle akselille, vaan pitää aikaväli sovitussa pienessä ajanjaksossa, jotta asioihin voi reagoida nopeammin. (Scrum n.d.)

Iteraatio on projektiryhmän ja asiakkaan sopima ajanjakson työjono, jonka loppu tuottaa jonkinlaista arvoa tuotteelle. Uusi työjono lukitaan iteraation alussa ja pyrkimyksenä on saada työjonon tehtävät suoritettua iteraation loppuun mennessä, ellei työjonoon tule muutoksia, jolloin aloitetaan uusi iteraatio. (Scrum n.d.)

Iteraatioita suoritetaan aina projektin alkamisesta projektin loppuun saakka. Isoa kokonaisuutta eli koko projektia pilkotaan tavallaan muutaman viikon mittaisiin tavoitteisiin. Iteraation aikana tiimi tapaa päivittäisessä kokouksessa tai raportoi valitussa paikassa tekemisistään päivittäin tiimille ja asiakkaalle. Tämä mahdollistaa nopeaa reagointia esteisiin ja mahdollisiin työjonon muutoksiin. Iteraation päätteeksi työjono käydään läpi ja kerrotaan iteraation aikana toteutetusta kehityksestä, jonka jälkeen prosessi aloitetaan uudelleen. (Schwaber 2004, 17-18)



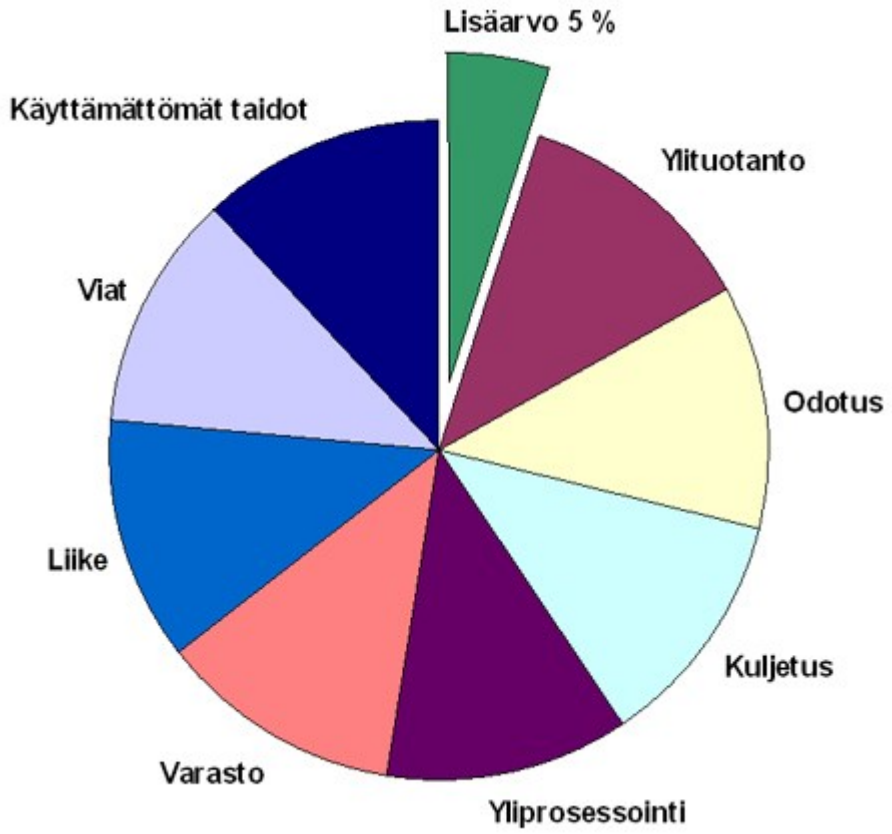
KUVIO 6. Scrum-ohjelmistokehittämisen malli (Digitaalinen Helsinki 2023).

6.3 Lean

Lean on alun perin adaptoitu Toyotan kehittämä organisaationaalinen johtamisfilosofia, joka tunnistaa ihmisten interaktiot ihmisten ja teknologian välillä. Filosofia lähti japanilaisen Taiichi Onon toimesta, joka lähti kehittämään tuotannon tehokkuutta Toyotan tehtaalla 1950-luvulla. (Lean n.d.)

Lean johtamisfilosofia tähtää vähentämään tuotantojärjestelmän sisäisten tapahtumien aikaa sekä vähentämään vastausviivettä asiakkaan ja toimittajan kesken. Periaatteena on vähentää hukkaa, jota voi koostua useista eri osa-alueista, kuten esimerkiksi odottamisesta. Jotta hukkaa voidaan poistaa, tulee ymmärtää mistä sitä koostuu.

Filosofian mukaan joitain tehtäviä tai kokouksia voitaisiin jättää pitämättä ja lopputulos olisi silti sama. Näistä turhista toimenpiteistä koostuu hukkaa ja ne tulisi olla tunnistettavissa. Hukalla tarkoitetaan kaikkea tekemistä tai tapahtumia, jotka eivät tuota asiakkaalle arvoa.



KUVIO 7. Toyotan alkuperäiset kahdeksan hukkaa (Piirainen 2008)

Yrityksen sisällä voidaan tehdä asioita, jotka ennaltaehkäisevät hukan muodostumista. Koko tietotekniikan ala on jatkuvaa oppimista ja yksi Lean-filosofian painokohdista on panostaa oppimiseen. Suunnittelemisen sijasta aikaa voidaan käyttää tuotteen kehittämiseen tai aikaa voi hyödyntää tutkimaan erilaisia ratkaisuja. Heti tuotteen kehittämisen jälkeen voidaan soveltaa testejä, jotka estävät vikojen muodostumisen.

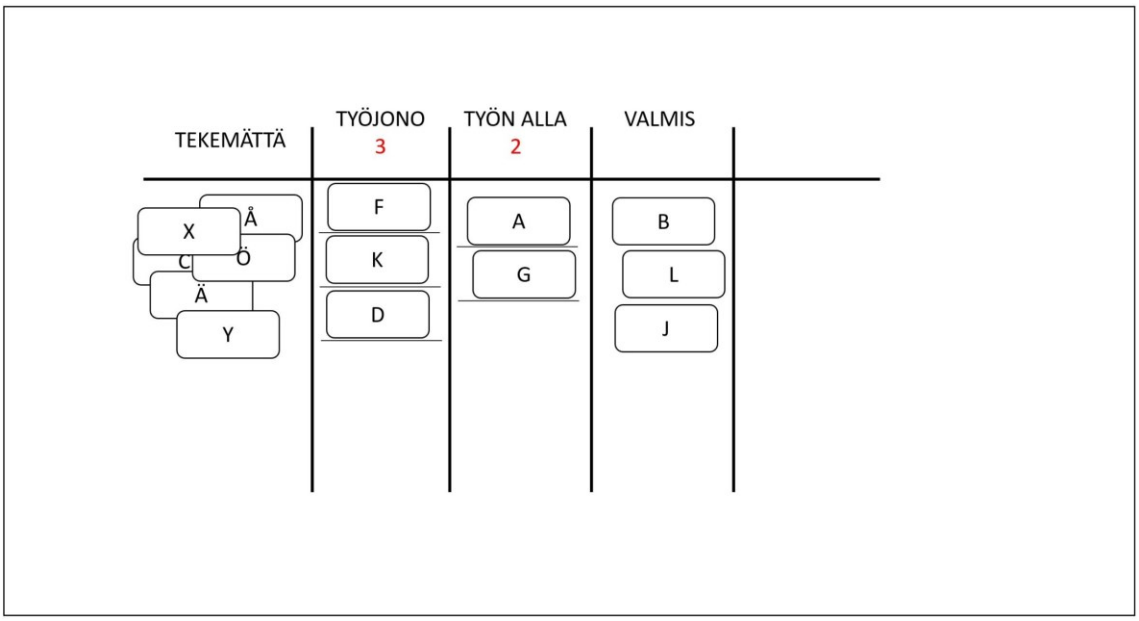
Lean pyrkii turhien päätäntäelinten ja hallinnollisten toimien muodostumisen. Päätetään asioista niin myöhään kuin mahdollista, tuotetaan niin nopeasti kuin mahdollista ja annetaan projektiryhmälle päätäntävalta. Myös asiakkaan tulee kuulua tähän kokonaisuuteen, koska projektiryhmä sekä asiakas tarvitsevat kokonaisvaltaisen kuvan tekemisestä, jotta päätöksiä voidaan tehdä nopeasti ilman ylimääräistä tarvetta kokouksille tai hyväksyntää päätäntäelimeltä. (Lean n.d.)

KUVIO 8. Lean periaatteet (Laaksoharju 2020).

6.4 Kanban

Kanban on Lean johtamisfilosofiasta johdettu aikatauluttamisen järjestelmä, joka vaatii reaaliaikaista kommunikaatiota ja täyttä työskentelyn läpinäkyvyyttä. Työjonon kohdat esitetään visuaalisesti Kanban-pöydällä, jotta jokainen tiimin jäsen näkee jokaisen työjonossa olevan asian ja sen etenemisen milloin tahansa. Pöytä voi olla mikä tahansa visuaalinen alusta, josta voi nähdä selvästi työjonon tehtävien etenemisen. (Koskinen 2021.)

Kanban luotiin alun perin osana Lean menetelmää tarkoituksena maksimoida tuotannon tehokkuus. Sen juuret ylettyvät 1600-luvulle, jolloin termi luotiin osana japanin kaupungillistumista. Sana tuleekin japanin sanoista ja voidaan suoraan pilkkoa kahdeksi sanaksi merkki ja lauta. Sen jälkeen siitä on tullut erittäin hyvä työkalu ketterään ohjelmistokehitykseen. Keskeisenä ajatuksena on digitalisoida visuaalisesti Kanban-pöytä, johon lisätään kortteja. (Kanbantool n.d.)



KUVIO 9. Kanban-pöytä työjonona (Hietaniemi 2020).

Projektiryhmä voi milloin tahansa nähdä pöydän kortit, jonka ansiosta projektiryhmän on helppo visualisoida työjonon virtaus. Jokainen kortti edustaa yhtä tehtävää työjonossa, josta pyritään vähentämään jonossa samanaikaisesti olevia kortteja. Kanban johtamisfilosofiassa tuotannon on tarkoitus keskittyä asiakkaan tarpeisiin ja odotuksiin, hallita työjonoa eikä työntekijöitä ja säännöllisesti tarkastella palvelun verkostoa. Kanban-käytäntöjen tulisi perustua nykyhetkestä lähtevään toimintaan, sitoutua tekemään inkrementaalisia ja kehittyviä muutoksia sekä kannustaa johtamiseen kaikilla työskentelyn tasoilla. Konsepti on joustava ja erittäin inhimillinen. (Kanbantool n.d.)

Kanban ei käske ihmisiä tekemään työtä tietyllä tavalla, vaan arvostaa työjonon nykyistä tilannetta ja antaa visuaalisen kuvan kehittää omaa työskentelyä. Menetelmällä keskeneräistä työtaakkaa vähennetään ja ymmärretään ihmisten työtaakkaa, josta visuaalisesti huomataan aikaisessa vaiheessa ylikuormitukset ja

mahdolliset työjonon ongelmatilanteet. Ongelmatilanteisiin voidaan reagoida nopeasti ja antaa palautetta prosessista, jotta työskentelyä voidaan muokata projektissa työskenteleville toimivaan tapaan. (Kanbantool n.d.)

7 PROJEKTITYÖN KÄYNNISTÄMINEN

Projektiryhmän ensimmäinen iteraatio alkoi kehittämään ketterää menetelmän ohjekirjaa kuuden vapaaehtoisen voimin. Projektia lähdettiin edistämään kerran viikossa kokouksella, jonka pohjalta käytettiin toinen tunti projektin edistämiseen. Ensimmäisen projektin kestoksi sovittiin kuukausi. Tavoitteena oli tehdä projektin aloitussuunnitelma eli mitä halutaan saavuttaa projektin elinkaaressa kokonaisuudessaan ja määrittää ensimmäisen projektin tavoitteet. Projektiryhmän elinkaaren pilkkominen useampiin projekteihin antoi mahdollisuuden saada tuotettua jotain konkreettista nopeasti ja saada tuotteeseen palautetta.

Ensimmäisestä projektista haluttiin lopputuloksena Haltun käsitys ketterästä ohjelmistokehittämisestä. Scrum-ohjelmistokehittämisen menetelmä on vain kuitenkin ohjenuora, jota haluttiin soveltaa toimivaksi omiin käytäntöihin. Scrummallia ei kuitenkaan koettu parhaimmaksi mahdolliseksi tavaksi tuottaa ketterästi ohjelmistoa, joten menetelmää piti soveltaa toimivaksi, jotta Haltun työntekijöiden olisi helpompi ottaa se käyttöön ilman tarvetta koulutukselle, joka kestää kuukausia.

Ensimmäisen projektin tavoitteeksi kuukaudessa asetettiin Ken Schwaberin ja Jeff Sutherlandin Scrum-oppaan lukeminen loppuun projektin henkilöiden kesken, Haltun Scrum-työkalujen valinta ja Haltun käsitys ketterästä ohjelmistokehittämisestä.

7.1 Projektin määrittely

Oletuksena projektiryhmän jäsenillä tuli olla luettuna Scrum-opas ja perustietämys, miten Scrum toimii ja mitä se vaatii toimiakseen. Ensimmäisen kokouksen aikana hahmoteltiin tapahtumia ja miltä tiivistettynä Haltun oma ketterä menetelmä tarvitsee toimiakseen Haltulla. Haluttiin tiivistettynä hahmotella, miltä yksi iteraatio voisi näyttää. Iteraation haluttiin olevan mahdollisimman selvä, koska oletuksena ohjeiden lukija ei välttämättä ymmärrä ketterästä ohjelmistokehittämisestä yhtä paljon kuin työryhmän jäsenet.

Haltun strategiapäivillä nousi tarve projektien parempaan hallintaan, jotta projektit eivät ole yhden henkilön vastuulla, josta nousi ketterän ohjelmistokehittämisen tarve projekteihin. Työryhmässä haluttiin käydä läpi esille nousseita ongelmia, joita ketterä menetelmä voisi ratkaista. Haasteena kuitenkin on menetelmän käyttöönottaminen pitkälle tulevaisuuteen, koska uusien työntekijöiden tulee olla vakuuttuneita sen tuottamasta arvosta projektille. Tämän takia ongelmien läpikäynti ja ketterän ohjelmistokehittämisen mallin tarjoaminen ratkaisemaan nämä ongelmat olisivat käytännöllinen tapa vakuuttaa sen tuottamasta arvosta.

Pohdimme projektiryhmässä ohjelmistoalan ongelmia ilman ketterää mallia. Isoimpana ongelmana koettiin tapahtumien puute eli asiakkaaseen ei olla yhteydessä tarpeeksi. Lisäksi asiakas ei usein osaa toimia tuoteomistajan roolissa, mikä ohjelmistokehityksessä olisi tärkeää projektin elinkaarelle. Vaihtoehtoisesti asiakkaana toimii usein komitea, mikä hidastaa päätöksentekoa ja päätöksenteko ei ole virtaviivaista. Kommunikaatio on kriittistä projektissa ja ongelmakohtiin voidaan tarttua heti niiden ilmaantuessa, jos asiakkaaseen voi olla yhteydessä mahdollisimman nopeasti. Sama pätee toiseen suuntaan eli tuoteomistaja voi olla nopeasti yhteydessä kehittäjiin.

Ideaalinen tilanne olisi asiakkaan toimiminen mukana projektiryhmän työskentelyä. Kommunikaation puute ja tuoteomistajan toiminnan puute johtaa kokonaisuuden käsityksen menettämiseen, vastuun välttelyyn ja turhaan selvittelyyn projektin kulusta, koska yhteistä käsitystä tavoitteista ei ole. Yhteisen tavoitteen puutteen takia lopputulos on mahdotonta saavuttaa ja hallinnan puuttumisen seurauksena ei ole mahdollista seurata projektin edistymistä tai tavoitteiden täyttymistä.

Toisena isona ongelmana koettiin olevan yksin toimiminen. Projekteissa ei ole minkäänlaista hallintaa, tiimityöskentelyä tai sitoutumista yhteiseen tavoitteeseen, jos projekteja tehdään vain tehtävä kerrallaan. Projektien ollessa yhden ihmisen harteilla, sitoudutaan liikaa yhteen ihmiseen.

Kolmantena isona ongelmana koettiin olevan suunnitellun arvon tuottaminen eli selkeän tekemisen puute kohti jotain tavoitetta. Käytännössä kehittäjät tekevät

asioita ja tuoteomistaja antaa hyväksynnän isompaa kokonaisuutta tietämättään. Ongelmakohta johtuu osittain tuoteomistajan kokemattomuudesta, mutta yhteisen suunnittelun lopputuloksena voitaisiin saada suunniteltua arvon tuottamista kohti järkevää kokonaisuutta, jotta tuoteomistaja voi saada sijoituksestaan irti enemmän.

Ohjelmistoalan ongelmien pohdinnan jälkeen haluttiin vastaus edellä mainittuihin ongelmiin Haltun käsikirjan ohjeissa. Projektiryhmässä alettiin suunnittelemaan itse sisältöä, joka ratkoo näitä ongelmia. Sisältöluetteloon haluttiin ainakin vastuut ja roolit, tapahtumat, miksi valittiin Scrum-malli ja ehdottomat asiat ketterän menetelmän toimimiseen.

Uudestaan huomioidaan, että ketterän ohjelmistokehittämisen menetelmää Haltulla ei kirjoiteta uudelleen tyhjästä, koska siinä ei olisi mitään järkeä kustannusten kannalta. Ohjekirjan lisäksi luodaan sen ulkopuolelle käytäntöjä, jotka auttavat ketterän ohjelmistokehittämisen menetelmän seuraamista ja toteuttamista. Esimerkiksi automatisointi vähentää projektinhallintaan käytettyä aikaa projektijohtajan tai projektissa työskentelevien tahojen toimesta. Automatisoinnin avulla ei ole tarvetta henkilölle, joka fasilitoi tehtäviä enempää kuin on tarve.

Tärkeimpänä tavoitteena koettiin saada menetelmä käyttöön jokaiseen projektiin ja tätä tietoa halutaan ylläpitää yhteisessä koko Haltua koskevan palaverin aikana. Tarvitaan siis jonkinlainen prosessi, johon ihmiset voivat käydä merkitsemässä sovitut prosessit ja niiden toteutuminen sekä merkitsemässä projektijohtajan kyseiseen projektiin, joka vastaa menetelmän seuraamisesta.

Tuloksena projektijohtaja on kirjattu ylös jokaiseen projektiin ja hän käy vuorollaan Haltun yhteisessä kokouksessa projektinsa tapahtumat läpi ja tarkastelee menetelmän toteutumista projektissa. Tapahtumien puuttumiseen voidaan siten puuttua ja käydä katsomassa projektin vointia tarkemmin, jos menetelmä ei jostain syystä toteudu kyseisessä projektissa.

8 TYÖN VAATIMUKSET

Ensimmäisen projektin tavoitteisiin aloitettiin uusi projekti, jolle valittiin ensimmäisen projektin tavoin budjetin sekä tavoitteet. Samat jäsenet jatkoivat projektiryhmässä työskentelyä kohti uutta tavoitetta eli itse sisältöä. Tavoitteena on päättää mitä sisältöä tarvitsemme tuottaa, jotta niiden pohjalta osattaisiin toimia projektijohtajana tai osana projektia.

Lisäksi sovittiin käyttäjätutkimuksen toteuttamisesta. Haluttiin tietää osaavatko lukijat ja tulevat projektijohtajat oikeasti toimimaan ohjeiden perusteella. Omia ohjeita ja käytäntöjä on helppo seurata. Tarvitaan siis käyttäjiä, jotka tietävät aiheesta mielellään todella vähän, jotta voi saada parhaan mahdollisen hyödyn ja palautteen ohjeista.

Keinoksi käyttäjätutkimuksen toteuttamiseksi sovittiin laittaa käyttäjät suoraan projektijohtajiksi fasilitoimaan projektin tapahtumia. Se on parhain mahdollinen tapa seurata onnistumista ja toteuttaa ohjeita jokaisen parhaaksi nähdyllä tavalla. Tämän projektin aikataulusta se kuitenkin jätettiin pois tarkoituksella, jotta projektilla on selkeä tuotettu hyöty ja asiat liikkuvat nopeasti eteenpäin, joten käyttäjätutkimus päätettiin hoitaa joskus myöhemmässä vaiheessa. Vasta oltiin kuitenkin päättämässä ohjeiden sisällöstä.

Projektin aikatauluksi kaksi kuukautta, jolloin on tuotettuna pääotsikot eli ohjeiden vaiheet, mistä halutaan kirjoittaa ja mitä asioita halutaan projektijohtajien tietävän ennen toimintaa.

8.1 Vastuut ja roolit

Projektijohtajan tulee hahmottaa omat vastuunsa ja tietää Haltun kontekstissa, mitkä roolit ovat työntekijöiden muodossa ja mitä rooleja on asiakkaan puolelta. Roolitus tulee tärkeäksi, jotta voimme auttaa toisiamme ja ymmärtää omat vastuualueet, jotta kehitystyö pysyy mahdollisimman sulavana ja samaa työtä ei tehdä useaan kertaan.

8.2 Menetelmän tapahtumat

Projektijohtajan tulee ymmärtää mitä tapahtumia ketterän ohjelmistokehittämisen menetelmät perinteisesti sisältävät. Tarkoituksena ei ole ottaa käyttöön suoraan olemassa olevia menetelmiä ja pitää jokainen tapahtuma säännöllisesti, vaan hyödyntää näitä tapahtumia ja ottaa niitä käyttöön tarvittaessa.

Lähdimme siltä kannalta, että esitämme vaihtoehtoja tapahtumien pitämiseksi. Hyväksyimme kuitenkin tosiasiat, että jokainen projekti on erilainen. Tulimme johtopäätökseen, että tapahtumien ei tule olla samanlaisia projektista toiseen, koska myöskään projektit eivät ole koskaan samanlaisia. Jokainen tapahtuma tai pidetty palaveri on projektinhallinnan työkalu, joka voi antaa arvoa ja parantaa työskentelyä sekä kommunikaatiota projektissa. Kaikkiin projekteihin ei kuitenkaan tarvitse jokaista tapahtumaa, kuten esimerkiksi retrospektiiviä, jossa tarkastellaan kuluneen iteraation onnistumisia ja epäonnistumisia.

Jos kehitystä tapahtuu muutaman kerran kuukaudessa tai tekijöitä on vain yksi, koetaanko silloin hyödylliseksi pitää retrospektiiviä? Asiaa pohdittiin projektiryhmässä ja tultiin johtopäätökseen, että tapahtumat ovat valinta, jonka jokaisen tulee valita ja ottaa käyttöön, jos siitä koetaan saavan arvoa tarpeeksi.

8.3 Perustelut mallille ja mitä ongelmia ratkotaan

Koimme tarpeelliseksi selventää projektikäytäntöjen hyödyn ja sen tuottaman arvon, jotta jokainen huomaa tarpeelliseksi käyttää tapoja sekä käytäntöjä, joita menetelmässä ollaan luomassa. Ihmistä on vaikea pakottaa toimimaan käytäntöjen pohjalta, joihin hän ei itse usko.

Ketterät ohjelmistokehittämisen menetelmät ovat tutkitusti hyödyllisiä ohjelmistoalalla. Tämän tiedon pohjalta päätettiin valita Scrum-ohjelmistokehittämisen malli pohjalle. Projektiryhmässä koettiin tarpeelliseksi avata tätä näkökulmaa ja avata projektiryhmän pohdintaa ongelmiin, joita on havaittavissa ohjelmistokehittämisen alalla, mitkä ratkaistaan ketterillä menetelmillä.

8.4 Jalkautus

Tultiin johtopäätökseen seurannan tarpeelle, jotta menetelmän jalkautus lähtisi odotetusti liikenteeseen. Jokaiselle projektille haluttiin projektijohtaja fasilitoimaan projektin kulkua ja huolehtimaan kommunikaatiosta projektin sisällä. Tarvittiin seurantaa, jotta projektijohtajien toimintaa voitiin seurata. Lopputuloksena Haltun yhteisen viikoittaisen palaverin aikana projektijohtajat käyvät läpi projektinsa tapahtumat ja kulun, jotta menetelmän toteutumista voidaan seurata.

8.5 Ehdottomat asiat

Aikaisemman pohdinnan tuloksen tultiin lopputulokseen, että ketterän ohjelmistokehittämisen tapahtumat tulisivat olla vapaaehtoisia ja käyttöönotettavissa tarpeeseen perustuen. Haltun kontekstissa ketterä ohjelmistokehittämisen menetelmä toteuttaa vain tapahtumat, joista koetaan oikeasti olevan hyötyä. Projektiryhmässä mietittiin, mitkä ovat vähimmäisvaatimukset, jotta voidaan todeta menetelmän toimivan, jos tapahtumat ovat vapaaehtoisia. Tulimme lopputulokseen, että vähintään Haltun GitLab-työkalussa on tehtävät listattuna.

GitLab on palvelu, joka tarjoaa versionhallintaa ja tehtävienhallintatoimintoja ohjelmistokehittäjille. Haltun ketterässä ohjelmistokehityksen menetelmässä tavoitteena on tehtävien listaaminen GitLab-työkaluun, josta näkyy aina, kuka on osallisena projektia, mikä on nykyisen iteraation tavoite, mikä on iteraation aikaväli ja miksi iteraatio on olemassa. Tämä koettiin ehdottomaksi, jotta projektissa voidaan toimia ketterästi.

9 PROJEKTITYÖN HAASTEET JA RETROSPEKTIIVI

Kolmannen projektin käynnistämisen aikana retrospektiivissä huomattiin ongelmia projektiryhmän toiminnassa. Huomattiin työn vievän liikaa aikaa ohjelmistokehityksestä ja kokoontumisten järjestäminen ajankohdallisesti sopivaksi kaikille oli haastavaa. Työn edistyminen oli hidasta ja tekeminen koettiin raskaaksi. Aikaisemmat projektit kestivät aikataulullisesti suunniteltua pitempään ja suunnitteluun käytettiin liikaa aikaa verrattuna tuotettuun sisältöön.

Projektiryhmän kokoa pienennettiin kolmeen henkilöön, jotka jatkaisivat viikoittaisilla kokouksilla. Jatkoin projektijohtajana toimimista ja lähdettiin kolmannessa projektissa tuottamaan sisältöä sovittujen otsikoiden alle. Viikoittain projektiryhmän tavoitteena oli luoda kahdessa tunnissa tiivistetysti sisältöä jonkin otsikon alle ja kuukauden päätteeksi valmis sisältö oli julkaisu valmis. Tuotettu sisältö hyväksyttiin teknologiapäällikkö Ilkka Hakkarin kautta muutosten kera, jos tämä koettiin tarpeelliseksi. Valmiin määritelmä projektille oli tuottaa lopullinen teksti Haltun käsikirjaan ja luoda konkreettiset käytännöt menetelmälle, jotka sopivat Haltun toimintatapoihin.

9.1 Käyttöönotto

Kolmannen projektin tavoitteena on luoda Haltun käsikirjaan ketterän ohjelmistokehittämisen menetelmän osio, johon on tuotettu teksti Haltun tavasta toteuttaa ketterää ohjelmistokehitystä, hyödyntäen aikaisempien projektien pohdintoja. Aikaisemmista vaiheista tuotettu arvo ja hyöty olivat vain projektiryhmän jäsenten päässä tässä kohtaa. Haluttiin tuoda teksti kaikille saavutettavaksi, jotta palautetta ja hyötyä saisi mahdollisimman pian.

Päätettiin projektin kestävän kuusi kuukautta kesään asti, jolloin Haltun käsikirjaan oli tuotettu ensimmäinen versio Haltun ketterän ohjelmistokehittämisenmallista. Projektiryhmässä lähdettiin kirjoittamaan sisältöä tavoitteena jokaisen kuukauden aikana saada tuotettua sisältö yhden ison otsikon alle ja saada siihen hyväksyntä. Hyväksynnän jälkeen teksti voitiin kirjoittaa Haltun käsikirjan

kokeelliseen versioon ja työntekijöiden hyväksymisen jälkeen itse oikeaan käsikirjaan luettavaksi julkisesti työntekijöiden kesken.

Hyväksynnän kautta työntekijät pääsivät käsiksi tuotettuun tekstiin ja pystyivät antamaan palautetta sekä toimimaan tekstin ohjeiden mukaisesti. Näin toteutetusta tekstistä saatiin aikaisin hyötyä liiallisen suunnittelun sijasta.

10 TYÖN TULOKSET

Edeltävät kappaleet sisältävät koko prosessin elinkaaren aikana tuotetut sisällöt. Tekstit kirjoitettiin kolmannen projektin aikana ja muokattiin soveltuvaksi Haltun työntekijöille palautteen pohjalta. Kappaleet löytyvät Haltun sisäisestä käsikirjasta ja jokainen kappale on allekirjoitettu Haltun työntekijöiden toimesta hyväksyntänä kirjoitetulle sisällölle. Lopullisen hyväksynnän on antanut Haltun teknologiajohtaja Ilkka Hakkari, joka esikatseli jokaisen kappaleen ennen työntekijöille julkaisua, mutta sisältö on täysin projektiryhmän sisällä tuotettua. Otsikkojen pääsanat löytyvät käsikirjasta sellaisenaan, mutta sisältöä on muokattu opinnäytteeseen viitekehyksen mukaisesti virallisemmaksi.

10.1 Haltun ketterä ohjelmistokehittäminen

Ketterä ohjelmistokehittäminen mahdollistaa projektityöksi kutsutun kaaoksen kesyttämisen. Samaa mallia käytetään yrityksen jokaisessa toiminnossa, niin myynnissä, kehittämisessä, viestinnässä kuin käyttäjätukeenkin liittyvissä toiminnoissa.

Käytäntöjen noudattaminen projektissa luo hyötyä esimerkiksi projektista toiseen vaihtuvalle projektijohtajalle, projektiin uutena jäsenenä tulevalle, projektiryhmästä toiseen hyppivälle ohjelmistokehittäjälle ja projektiryhmän nykyisille jäsenille helpottamalla asioiden muistamista. Projektijohtajan tehtäviin kuuluu huolehtia, että projektissa on Haltun ketterä menetelmä käytössä.

Haltun ketterän ohjelmistokehittämisen menetelmän pohjana toimii kansainvälinen Scrum-opas, joka kuvaa toimintatavat, joita kokonaisuutena kutsutaan Haltun ketteräksi menetelmäksi. Haltulla menetelmää sovelletaan projektien tarpeisiin, eli projektiryhmä itse valitsee tarkoituksenmukaiset työkalut, joita hyödynnetään projektissa. Scrum-opas toimii pohjatietona Haltun käytännöille. Jokaisen tulee perehtyä ja omaksua kansainvälinen Scrum-opas.

Scrum valittiin, koska se tarjoaa välineet ja keinot parantavat meidän ja asiakkaidemme välistä kommunikaatiota, sekä tekee projektihallinnasta selkeämpää. Ketterää ohjelmistokehitystä toteutetaan, koska halutaan reagoida jatkuvasti muuttuviin tilanteisiin nopeasti ja tehokkaasti.

Haltun ketterästä ohjelmistokehittämisen menetelmän projektista löytyy aina vähintään tuoteomistaja, projektijohtaja ja kehittäjä.

Menetelmään kuuluu erinäisiä tapahtumia, joilla edesautetaan tiimin päätöksentekoa ja rytmitetään yhteistä tekemistä. Yleisiä työkaluja, joita usein hyödynnetään projekteissa, on päivittäinen tekemisen avaamisen palaverin, iteraation suunnittelu, iteraation esittely ja retrospektiivi tekemisille. GitLab-työkalusta täytyy löytyä työjono, josta löytyy iteraatiossa toteutuvat tehtävät ja niiden tekijät. Iteraation aikataulu lisäksi GitLab-työkalusta tulee käydä ilmi miksi ja mitä ollaan toteuttamassa.

Vähintään jokaisessa projektissa käytössä työjono ja sieltä löytyy tehtäviä. Käytännössä yksinkertaisemmin tämä tarkoittaa sitä, että jokaisessa projektissa on tavoite, jota kohtaan työskennellään ja kirjataan mitä asioita sen eteen on tehty, jotta päästään kohti sitä tavoitetta. Gitlabissa tämän asian ajaa työjono, johon meillä on kirjattu iteraation tavoitteet ja sinne valittu työjonosta tehtävät, jotka tulee tehdyiksi sillä iteraatiolla.

Haltun menetelmän näkökulmasta tarvitaan vähintään nämä kaksi asiaa, jotta meillä on selvää mitä projektissa ollaan tällä hetkellä tekemässä ja mitä meidän pitää työstää, jotta pääsemme tavoitteeseen. Lisäksi konsistentti työjono ja tehtävä työkalujen käyttö jättää aina jäljen ja merkin projektin historiaan, mikä on tärkeää selvitettäessä mitä projektiin on tehty, ja kuka siihen on koskenut viimeiseksi.

Työjonon ja siihen liittyvien tehtävien auki kirjoittamiseen kannattaa käyttää ajatusta ja aikaa, jotta myös muilla olisi mahdollisimman selvä käsitys ilman syvempää kontekstia, miksi ne on ollut olemassa ja mitä niissä on tehty. Jo hyvä otsikko voi selventää esimerkiksi tehtävän tarkoituksen lukematta syvemmällä ohjelmistoprojektin koodia.

Projekteissa vaihtuu tekijät ja joihinkin projekteihin kosketaan hyvinkin harvakseltaan. Jatkuvasti tapahtuu tilanne, jossa tekijät tarvitsevat ensimmäiseksi pohjatietoa, kuka on koskenut projektiin viimeiseksi ja minkä takia, jos meillä ei ole dokumentoitua historiaa projektin elinkaaresta. Pahimmassa tilanteessa kellään ei ole mitään tietoa projektista ja joudutaan lähtemään nollasta selvittämään tilannetta.

Kaikkiin projekteihin on valtava apu, kun niihin dokumentoidaan historiaa työjonon ja tehtävien muodossa. Uusi tekijä projektissa voi käydä lukemassa edeltävän työjonon, lukea kuvauksesta minkä takia siihen on koskettu viimeksi, mitä siellä on toteutettu ja minkä takia juuri ne asiat on tehty. Projektin dokumentoitu historia säästää aikaa ja hermoja, koska historiaa ei tarvitse selvittää muilta ihmisiltä. Työjonon luominen ja kirjoittaminen miksi olet koskemassa projektiin, vie murto-osan ajasta mitä siihen käytetään, kun seuraava tekijä lähtee selvittämään projektista jotain. Ei ole yhtään hyvää syytä olla jättämättä jälkeä omasta tekemisestä projekteissa.

Visio tuotteesta auttaa projektiryhmää ymmärtämään tavoitteet ja hallitsemaan väärinymmärrysten riskiä. Visiota luotaessa riskejä käsitellään hyvin korkealla tasolla. Tämä auttaa yhteisen tavoitteen ymmärtämisessä. Visio muodostetaan tuoteomistajan ja projektitiimin yhteistyöllä.

Etenemissuunnitelma tarjoaa yleisen kuvan projektin vaatimuksista ja niiden tärkeysjärjestyksestä. Projektiryhmä voi käyttää tätä työkaluna hallinnoimaan projektia yhtenä kokonaisuutena.

Tuotteen työjono luodaan yhdessä tuoteomistajan kanssa. Tuotteen visio ja suunnitelma muutetaan tärkeysjärjestyksessä olevaksi työjonoksi. Työjonolla hallitaan projektissa tapahtuvia muutoksia. Projektia ei suunnitella tarkkaan alusta loppuun asti, vaan työjono elää projektin aikana sitä mukaa kun näkymä suunnitelman seuraavaan vaiheeseen tarkentuu.

Tuotteen työjonosta otetaan tehtäviä ja ne pilkotaan osaksi iteraation työjonoa paljon tarkemmalle tasolle. Iteraation aikana kehitystiimi toteuttaa iteraation työjonossa olevia tehtäviä. Iteraation aikana työjonosta nähdään, miten projekti etenee ja mitä ongelmia ja riskejä tulee vastaan.

Jokaisen sprintin alussa päätetään, mitä työtä tuotteen työjonosta otetaan projektin työjonoon. Projektin sisältöön liittyviä riskejä voidaan käydä läpi hyvinkin tarkkaan ja miettiä, miten ne vaikuttavat kokonaisuuteen. Iteraation aikana kehitystiimi toteuttaa työjonossa olevia tehtäviä. Iteraation aikana työjonosta nähdään, miten projekti etenee ja mitä ongelmia tai riskejä voi tulla vastaan.

Jokaisen iteraation päätteeksi työn tilaajalla, asiakkaalla, on mahdollisuus tutustua toteutettuun toiminnallisuuteen. Asiakas hyväksyy jokaisen iteraation tuotoksen ja näin pääsee projektin aikana varmistamaan, että koko projektin lopputulos vastaa toivottua. Asiakkaalla on mahdollista seurata työn edistymistä myös iteraation aikana.

Lopuksi käydään läpi, mitä iteraatiossa tapahtui, millaisessa toimintaympäristössä työskentely tapahtui, ja mietitään mitä hyvää ja huonoa aikavälillä oli. Retrospektiivissä tiimi tutkii iteraation onnistumista, ja pohditaan parannuksia tiimin työtapoihin ja tuotteen kehittämiseen liittyviin asioihin tulevia tehtäviä varten.

11 VASTUUT JA ROOLIT

Projektityötä tehtäessä on tärkeää, että jokainen tietää oman roolinsa, jotta tehokas työskentely projektiryhmässä on mahdollista. Vastuut ja roolit osiossa kuvataan Haltun ketterän ohjelmistokehittämisen roolit ja millaisia vastuita ja tehtäviä kullekin roolille kuuluu. Projektista löytyy aina vähintään roolit: tuoteomistaja, projektijohtaja ja kehittäjä.

11.1 Tuoteomistaja

Ideaalimaailmassa tuoteomistaja vastaa tuloksekkaasta tuotteen kehitysjonon hallinnasta. Käytännössä ohjelmistoalan asiakaskunnalla ei yleensä ole tuoteomistajuuteen koulutusta tai kokemusta. Tämän on koettu johtavan projektin sekavuustilanteeseen, jossa suuri visio on selvillä, mutta yksityiskohdat ovat täysin hämärän peitossa, eikä asiakkaalla välttämättä ole itsellään kykyä ja osaamista määritellä yksityiskohtia. Tuoteomistaja itse on viimekädessä vastuussa siitä, että nämä alla listatut asiat toteutuva ja tuoteomistaja antaa viimeisen sanan ja hyväksynnän työjonolle.

Haltulla projektiryhmä on yhdessä vastuussa tuoteomistajan auttamisesta. Käytännössä auttaminen voi sisältää esimerkiksi alla listattujen kehitysjonon hallintaan liittyvien toimenpiteiden tekemistä yhdessä tuoteomistajan kanssa.

Tuotteen tavoitteen luominen ja siitä viestiminen täsmällisesti, tuotteen kehitysjonon sisällön valmistelu ja sen viestiminen selkeästi, tuotteen kehitysjonon sisällön järjestäminen, tuotteen kehitysjonon läpinäkyvyyden, saatavuuden ja ymmärrettävyyden varmistaminen

Tuoteomistajan auttaminen ja kommunikaation ylläpitäminen on äärimmäisen tärkeää yhteisen tavoitteen ymmärtämisen kannalta, jotta voidaan tuottaa iteraatioissa arvokkaaksi koettuja tehtäviä.

11.2 Projektijohtaja

Projektijohtajan tehtäviin kuuluu huolehtia, että projektissa on menetelmä käytössä. Projektijohtaja vastaa Haltun ketterän ohjelmistokehittämisen menetelmän toteutuksesta. Tämä tapahtuu auttamalla kaikkia projektiryhmässä sekä ympäröivässä organisaatiossa ymmärtämään Scrum-mallin teoria ja käytännöt.

Projektijohtaja valmentaa projektiryhmän jäseniä itseohjautuvuuteen ja monialaisuuteen. Projektijohtaja auttaa ryhmää keskittymään käyttäjille arvokkaiden ja valmiin määritelmää noudattavien tehtävien toteutukseen. Projektissa tulee seurata menetelmän asettamia tavoitteita ja käytäntöjä, joista raportoidaan yhteisessä viikoittaisessa palaverissa.

Projektijohtaja poistaa ryhmän esteitä edistää työjonon tehtäviä ja huolehtii projektissa pidettävien tapahtumien aikataulusta sekä fasilitoinnista, jotta tapahtumat pysyvät annetun ajan puitteissa. Aikataulussa pysymisen lisäksi projektijohtajan tulee varmistaa osallistujat ja antaa jokaiselle oman puheenvuoron sekä pitää huolta tapahtumaan asetetusta agendasta. Tapahtuman aikana muodostetut muistiinpanot kirjataan ylös projektiryhmän sopimaan helposti löydettävään sijaintiin, joka on helposti saatavilla. Pyydettäessä projektijohtaja huolehtii sidosryhmien välisestä yhteistyöstä.

11.3 Kehittäjä

Kehittäjät ovat projektiryhmän ihmiset, jotka ovat sitoutuneet työskentelemään yhdessä kohti iteraatiolle määritettyjä tavoitteita. Haltulla kehittäjiin kuuluvat ohjelmistokehittäjät, järjestelmän ylläpitäjät, UX-suunnittelijat, graafikot, ja kaikki muut tarvittavat tekijät, jotta iteraation tehtävistä muodostuu projektille arvokasta tavaraa.

Kehittäjät laativat iteraation suunnitelman eli iteraation työjonon, jonka tehtävät noudattavat valmiin määritelmää. Suunnitelmat tulee päivittää jokainen päivä iteraation tavoitteen saavuttamiseksi, ja projektiryhmän jäsenet auttavat toisiaan tavoitteiden saavuttamiseksi. Kehittäjien tulee tietää tapahtumien sisällöstä ja

valmistautua tapahtumien vaatimilla tiedoilla sekä muodostaa tapahtumista dokumentoidun historian GitLab-työkaluun.

12 TAPAHTUMAT

Tapahtumat, jotka vähintään on pidettävä, että Scrum-malli voi toteutua: päivittäinen tekemisen suunnittelu, iteraation suunnittelu, iteraation esittely ja retrospektiivi

Näiden tapahtumien lisäksi tiimi voi päättää yhdessä muiden tapahtumien järjestämisestä esimerkiksi retrospektiivistä nousseiden kehitystarpeiden vuoksi. Tapahtumia tulee käsitellä työkaluina, joita voidaan ottaa käyttöön niiden tarpeellisuuden kannalta. Tapahtumien kuvaukset löytyvät Scrum-oppaasta. Seuraavissa kappaleissa käsitellään tapahtumien erityisyyksiä Haltulla.

Projektijohtaja vastaa siitä, että projektiryhmä pitää tapahtuman asiassa, huolehtii aikataulun pitävyydestä ja kalenterikutsun osallistujien asiallisuudesta. Tapahtuman kestäessä pitkään, projektiryhmä järjestää erillisen tapahtuman, missä asiaa käsitellään syvemmin. Tapahtumassa ei esimerkiksi määritellä tehtävien ominaisuuksia, vaan niille järjestetään oma palaveri.

Tapahtumista tallennetaan muistiot ja asiakas pyritään ottamaan mukaan kaikkiin tapahtumiin. Asiakas voi kieltäytyä osallistumisesta, mutta projektiryhmän vastuulla on huolehtia, että asiakas ymmärtää osallistumisen tärkeyden.

12.1 Päivittäinen suunnittelupalaveri

Tarkoituksena on jakaa projektiryhmän sisällä tietoa etenemisestä ja esteistä, jotta ryhmä voi linjata omaa tekemistään sopivaksi iteraation tavoitteen saavuttamiseksi. Projektiryhmä sopii toteutuksesta, mutta suosittuja vaihtoehtoja on säännöllinen kalenterikutsu Google Meet-työkaluun, viestiketju organisaatioiden sisäiseen viestintään suunnatussa ohjelmassa Slackissä tai Gitlab-työkalussa.

Jokainen projektiryhmän jäsen esittelee mitä on tehnyt edellisen päivittäisen suunnittelupalaverin jälkeen, mitä aikoo tehdä ennen seuraavaa palaveria ja ilmaisee mahdolliset esteet, jotka estävät iteraation tavoitteen saavuttamisen.

Päivittäisessä suunnittelupalaverissa ei pitäisi tulla mitään uutta tietoa mitä ei GitLab-työkalusta jo ennestään löydy. Gitlabin automaatio luo projektiryhmälle valmiin muistiinpanopohjan tehtävien muodossa päivittäin. Muistiinpanotehtävät tallennetaan projektille talteen.

Palaveria varten tulee olla luotuna kalenterissa oma päivittäinen tapahtuma, johon projektin jäsenet osallistuvat Google Meet-työkalun avulla. Päivittäisessä suunnittelupalaverissa on tärkeää koko projektiryhmän osallistuminen, joten heidät tulee olla kutsuttuina kyseiseen tapahtumaan kalenterissa. Projektijohtajantehtävä on huolehtia, että kalenterimerkinnän osallistujalista päivittyy projektiryhmän koostumuksen muuttuessa.

Jokainen ryhmän jäsen tulee palaveriin paikalle ajallaan. Tilaisuus kestää ainoastaan vartin, joten jo muutaman minuutin odottelu häiritsee merkittävästi toteutumista. Toisinaan päällekkäisien menojen takia palaverin kokoonpano voi kuitenkin vaihtua. Päivittäistä suunnittelupalaveria ei tule peruuttaa, mikäli henkilöitä jostain syystä ei pääse paikalle. Jos tiimin jäsen ei pääse osallistumaan yhtä aikaa muiden kanssa palaveriin, hän voi esimerkiksi projektin Slack-viestintäkanavalla päivittää tilanteensa.

Joskus on myös järkevää käydä koko palaveri pelkästään Slack-työkalussa. Tällöin projektiryhmä sopii siihen käytännöt keskenään. Samat kolme kysymystä mitkä esitetään tilaisuudessa, voidaan käsitellä myös Slackissä.

Päivittäisessä suunnittelupalaverissa ei ole tarkoitus ratkoa ongelmia, vaan nostaa ongelmat esille, jotta ratkaisulle voidaan järjestää tarvittaessa oma tapaaminen niiden jäsenten kesken, joita ongelma ja sen ratkaisu koskettaa. Ratkaisua voidaan hakea myös tiimin ulkopuolelta muilta Haltun työntekijöiltä.

Tapahtumassa projektijohtaja on puheenjohtajan roolissa, jakaen puheenvuoroja ja pitämällä huolen siitä, että tapahtuma vastaa sen merkitystä. Projektijohtaja avaa palaverin läpikäymällä koko tiimiä koskettavat asiat, kuten asiakkaalta tulleiden viestin läpikäynti ja muistutus julkaistavasta tuotteesta.

Koko projektiryhmää koskettaviin asioihin annetaan jokaiselle ryhmän jäsenelle lyhyt puheenvuoro. Toisinaan yleisiä asioita ei ole, eikä niitä siten myöskään käsitellä. Yleisten asioiden jälkeen projektijohtaja antaa puheenvuoron vuorotellen jokaiselle tilaisuudessa paikalla olevalle henkilölle ja kysyy mitä on tehty, mitä tullaan tekemään ja onko joitain esteitä etenemiselle kohti iteraation tavoitetta.

Päivittäisen suunnittelupalaverin ei kuulu kestää varttia kauempaa. Projektijohtajan tehtävänä on fasilitoida, eli keskeyttää puheenvuorot, jos alkaa näyttämään siltä, että ongelmia aletaan ratkoa tai joku käyttää liikaa aikaa yksityiskohtiin. Tärkeää on saada koko tiimin toimintaa koskettavat merkittävät asiat kaikkien tietoon, ei kuluttaa kaikkien yhteistä aikaa turhaan.

12.2 Iteraation suunnittelu

Tarkoituksena ei ole alkaa tämän tapahtuman aikana suunnittelemaan miten jokin asia toteutetaan, eli koodataan. Tarkoituksena on valita iteraation sisältö, eli valitaan tuotteen työjonolta ne asiat mitkä tullaan toteuttamaan tämän iteraation aikana.

Iteraation työjonon kuuluu pitää sisällään sellaista tekemistä, joka vie projektiryhmää lähemmäs valmiin määritelmää ja projektin tavoitetta. Iteraation suunnittelussa käydään läpi tuotejono ja valitaan yhdessä asiakkaan kanssa seuraavalle alkavalle iteraatiolle suunnitteluun tai kehittämiseen valittavat tehtävät tärkeysjärjestyksessä. Tuloksena on GitLab-työkalussa uusi iteraatio, johon on kirjattu tehtävät työjonoon. Suunnittelun valmistumiseen tarvitaan kaikkien paikalla olevien hyväksyntä ja sitoutuminen.

On tärkeää, että koko projektiryhmä, mukaan lukien asiakas, ja mahdolliset sidosryhmät osallistuvat, joten heidät tulee olla kutsuttuina kyseiseen tapahtumaan. On projektijohtajan tehtävä huolehtia, että kalenterimerkinnän osallistujalista päivittyy ryhmän koostumuksen muuttuessa iteraatiosta toiseen.

Tapahtumassa projektijohtaja on puheenjohtajan roolissa, jakaen puheenvuoroja ja pitämällä huolen siitä, että tapahtuma vastaa sen merkitystä sekä tapahtuman pysymisestä aikataulussaan. Projektijohtaja ei päätä, eikä ohjaa, mitä projektiryhmä valitsee iteraation työjonolle. Projektijohtaja avaa tilaisuuden muistuttamalla projektin tavoitteesta. Hän voi tehdä muistuttamisen kysymällä sitä projektiryhmältä.

Asiakas tuoteomistajan roolissaan määrää tuotteen työjonon, ja tässä tilaisuudessa valitaan tuotteen työjonolta korkeimmalla prioriteetilla olevat tehtävät seuraavan iteraation työjonoon. Usein Haltun asiakkailla ei ole tuoteomistajuudesta kokemusta tai osaamista, joten projektiryhmä auttaa työjonon suunnittelussa. Jos tuoteomistajalla ei ole entuudestaan suunnitelmaa työjonon tehtävistä, voi projektiryhmä asiantuntemuksellaan valita työjonon tehtävät hänen puolestaan ja kysyä mielipidettä suunnitelmasta. Lopullisen päätöksen työjonosta tekee aina tuoteomistaja ja hänen toiveitaan tulee kunnioittaa.

Projektijohtaja ohjaa keskustelua ja huolehtii, että iteraation työjono tuottaa kokonaisuudessaan jonkinlaista arvoa kohti projektin tavoitetta. Projektijohtajan tehtävänä on huolehtia, että kukaan ei putoa suunnitelmasta ja kaikilla on yhteneväinen käsitys projektin tavoitteesta, iteraation tavoitteesta, ja iteraation työjonon sisällöstä. Jokaisen ryhmän jäsenen pitää ymmärtää mitä ollaan tekemässä, jotta jokainen jäsen voi sitoutua suorittamaan hänelle osoitetut työjonon tehtävät iteraation aikana.

Gitlabissa yhtä iteraatiota vastaa aina yksi työjono. Iteraatiot luodaan GitLabissa palveluryhmään, ei yksittäiseen projektiin, niin että samalle iteraatiolle voidaan lisätä tehtäviä kaikista niistä kokonaisuuksista, joihin tapahtuu kehitystä asiakkaalle.

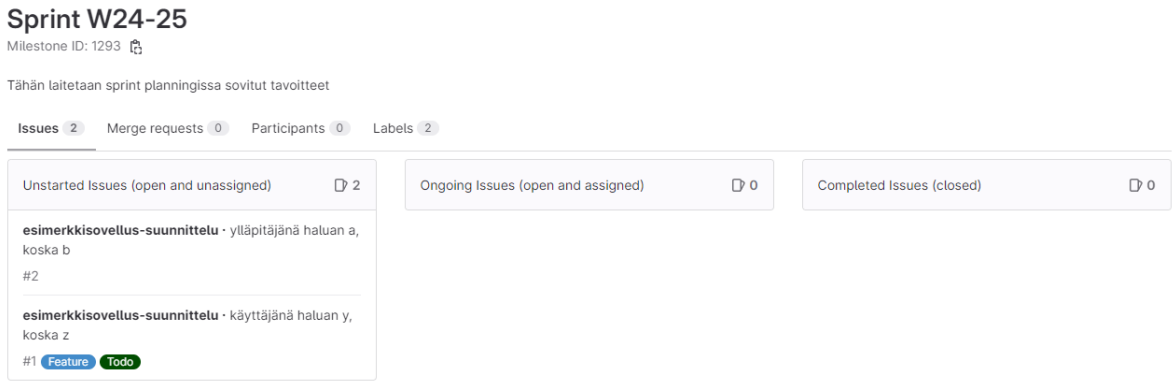
12.3 Iteraation esittely

Iteraation esittelyssä käydään läpi syntyneet tuotokset. Projektiryhmä esittelee asiakkaalle, toisilleen, ja mahdollisille muille sidosryhmille, mitä he ovat tuotta-

neet ja miten he ovat edistäneet projektin tavoitteeseen pääsemistä. Tilaisuuden päätarkoituksena on, että asiakas hyväksyy toimituksen. Jos asiakas ei hyväksy toimitusta, tehdään näistä huomioista muutospyynnöt tuotteen työjonolle mitkä iteraation suunnittelussa priorisoidaan ja päätetään koska ne tullaan toteuttamaan.

Tapahtumassa projektijohtaja on puheenjohtajan roolissa, jakaen puheenvuoroja ja pitämällä huolen siitä, että tapahtuma vastaa sen merkitystä sekä tapahtuman pysymisestä aikataulussaan.

Projektijohtaja avaa ennen tapahtumaa iteraation työjonon eli Gitlabin työjononäkymän. Projektiryhmän jäsenet valmistautuvat esittelemään tuotoksensa. Iteraatiossa tehtyjä tehtäviä aletaan käymään läpi oikeanpuoleisesta sarakkeesta alkaen, josta löytyy suljetut tehtävät, joita on iteraation aikana saatu valmiiksi. Jokaisen yksittäisen tehtävän kohdalla projektijohtaja antaa puheenvuoron tehtävän kehittäjille. Kehittäjät kertovat mitä tehtävä pitää sisällään ja esittelevät toteutuksen toiminnallisuuden. Esittely voi tapahtua palvelimelta, omalta koneelta, lähdekoodia läpikäyden, graafisia kuvia esitellen, tai mitä ikinä onkaan tuotettu. Tärkeää on, että asiakkaalle selvästi esitellään tehtävässä toteutetut asiat.



KUVIO 9. GitLab-työkalun hyödyntäminen työjonon hallintaan.

Vaikka esittely tapahtuu jokaisen tehtävän kehittäjän toimesta, kyseessä kuitenkin on projektiryhmän yhdessä tekemät saavutukset, eli esittelyssä kaikki projektin jäsenet auttavat toisiaan. Esimerkiksi sairaustapauksen sattuessa ei tehtävää ja tuotosta jätetä esittelemättä vaan muut jäsenet paikkaavat poissaolijaa ja käy yhdessä läpi tehtävän.

.

Oikeanpuolimmaisen sarakkeen jälkeen siirrytään keskimmäiseen sarakkeeseen, käydään läpi keskeneräiset tehtävät, joita työstetään vielä ja miten ne ovat edistyneet.

Vasemmanpuoleisessa sarakkeessa olevat aloittamattomat tehtävät on hyvä käydä viimeisenä läpi yleisesti ottaen ja kertoa miksi niitä ei ole aloitettu vielä. Esittely tilaisuudessa ei oteta kantaa seuraavan iteraation sisältöön, ja aloittamatta jääneitä tehtäviä ei siis suoraan tilaisuudessa automaattisesti siirretä seuraavalle iteraatiolla. Tehtävien prioriteetti on saattanut muuttua, joten tuotteen työjono on syytä käydä läpi suunnittelupalaverin ohjeiden mukaisesti.

Tiivistetysti tapahtumien kulku on projektin tavoitteen kertaaminen. Iteraatiolle valitut tuotokset käydään yksitellen läpi ja kirjataan valmiiksi tai tehdään muutospyyntö ominaisuudesta, jolloin aloitetaan uusi tehtävä työjonolle. Aloittamatta jääneet tehtävät palautetaan takaisin tuotteen työjonoon ja pyydetään asiakkaalta hyväksyntä iteraation tuotoksille.

12.4 Retrospektiivi

Projektiryhmä kehittää ja muuttaa omaa toimintaansa niin että se sopii toimintaympäristöön. Retrospektiivissä käydään läpi päättyneen iteraation asioita retrospektiivisessä kulmassa. Tukitaan mikä meni hyvin, missä voitaisiin parantaa ja onko iteraation aikana herännyt kysymyksiä.

Projektiryhmän sisältä nousseita aiheita käydään läpi ja keskustellaan. Retrospektiivin tarkoituksena on saada konkreettisia tehtäviä, että ne toteutuvat. Nousseet tehtävät otetaan ylös projektiryhmän valitsemaan paikkaan ja ryhmä sitoutuu noudattamaan tehtyjä muutoksia.

13 TUOTTEEN TYÖJONO

Tuotteen työjonon tehtävän tulee olla tarpeeksi yksityiskohtainen, jotta se voidaan suorittaa iteraation aikana. Mikäli tehtävän kuvaus jättää tulkinnanvaraa, pitää tehtävä pilkkoa pienempiin osakokonaisuuksiin. Muutospyynnön tullessa asiakkaalta lisätään työjonolle uusi tehtävä. Uuden tehtävän tullessa sen tulee olla otsikoltaan tarpeeksi tarkka, jotta tehtävän voi toteuttaa ilman lisäkysymyksiä. Lisäkysymysten noustessa, luodaan kysymyksistä uusia tehtäviä, kunnes ollaan tarpeeksi matalalla tasolla, jolloin otsikosta saa selvää mitä ollaan tekemässä.

Tehtävien kautta saadaan selvitettyä tarkemmat vaatimukset ja valmiin määritelmät uuden toiminnallisuuden osille. Toteutuksen tehtäviin voidaan liittää suunnittelun tehtäviä, jotta asiakokonaisuudet pysyvät yhdessä. Tällä tavalla koko projektin ominaisuuden etenemistä voidaan seurata paremmin tarkastelemalla asiakokonaisuuksia. Tehtävien osittaminen pieneksi mahdollistaa myös tehtävän valmistumisen iteraation aikana, jotta ei päädytä tilanteeseen, jossa tehtäviä siirretään iteraatiosta iteraatioon.

Iteraatio epäonnistuu aina, kun tehtävät jatkuvat iteraatiosta toiselle. Epäonnistumiseen voi olla useita syitä, mutta yleisin on liian epäselvän tai ison tehtävän nostamista iteraation työjonolle. Kokonaisen uuden toiminnallisuuden kuvaaminen yhdellä tehtävällä voi olla joko mahdotonta tai liian haastavaa yhdellä iteraatiolla, koska projektiryhmällä ei ole tarkkaa kuvaa toiminnallisuuden kaikista vaatimuksista. GitLab-työkalussa tehtäville voidaan antaa otsikoita, joiden avulla ryhmitteleminen helpottuu ja mahdollistaa eri käyttötapauksia.

14 EROT SCRUM OHJELMISTOTUOTANNON MALLIIN

Kehitetyssä mallissa halutaan korostaa läpinäkyvyyttä ja asiakassuhteen ylläpitoa. Asiakas on keskiössä mukana tekemisessä ja halutessaan voi osallistua jokaiseen tapahtumaan osana projektiryhmää. Asiakasta ei puhutella eri tasolla, kuin muita tiimin jäseniä eli asiakkaat yksilöidään ja yritetään ymmärtää heidän asemaansa yksilöinä omilla rajoitteilla sekä tavoitteilla.

Ohjelmistotuotannossa yksikään projekti ei ole samanlainen, joten projektin käytännöt ja menetelmät ei tulisi olla samanlaisia. Jokaisella projektilla on eri tarve käytännöille. Kehitetyssä mallissa halutaan antaa voima projektiryhmälle päättää millä tavalla projektia kannattaa toteuttaa ja millä tapahtumilla. Malli tarjoaa käytäntöjä sekä esimerkkejä tapahtumista, mutta käytännön tasolla vain harva käytäntö on oikeasti pakollisia, koska halutaan mahdollisimman vähän pakollisia käytäntöjä.

Pakollisia käytäntöjä tarvitaan, jotta voidaan puhua ketterästä menetelmästä, jotta projektissa työskenteleminen on joustavaa ja läpinäkyvää. Ilman minkäänlaisia pakollisia käytäntöjä ei voi varmistaa projektien yhtenäisyyttä. Pakollisia käytäntöjä on vain projektin työskentelyyn liittyvät asiakohdat, josta käy ilmi projektiin sisällä työskentelevät henkilöt, mitä projektiryhmän työjonossa on ja mikä työjonon tavoite on aikataulun kanssa.

Projektin elinkaaren aikana pidettävät tapahtumat on tarkoitettu olevan modulaarisia ja otettavan käyttöön vain tarvittaessa. Tapahtumat ovat neuvoteltavissa asiakkaan kanssa ja otettavan käyttöön, jos tapahtumista koetaan olevan hyötyä projektin hallinnan kannalta. Scrum-mallin kaltainen syklinen tapahtumien pakollisuus ei sovellu jokaiseen projektiin rajoitteiden takia.

Yhtäaikaisten projektien määrän takia nimettyä fasilitaattoria ei haluta nimetä, vaan luodun mallin tarkoitus on mahdollistaa jokaisen työntekijän mahdollisuuden toimia fasilitoijana ja johtaa projektia. Ohjeiden ja käytäntöjen yksinkertaisuudella sekä käytännön esimerkkien kautta projektin johtaminen ei vaadi aikaisempaa kokemusta tai käytännön kokemusta. Ohjeet antavat myös valmiudet

projektiryhmälle osallistua tapahtumiin ja ymmärtää niistä halutun lopputuloksen. Tällä tavalla esimerkiksi uusi työntekijä pääsee nopeasti mukaan projektiryhmän tekemiseen ilman isompaa tarvetta kouluttaa kyseistä henkilöä.

15 KÄYTTÄJÄHAASTATTELU

Ketterän ohjelmistokehittämisen menetelmän jalkautuksen jälkeen eli tekstin löytyessä käsikirjasta, haluttiin saada palautetta ohjeiden selkeydestä ja toimivuudesta oikeissa tilanteissa. Käsikirjan tekstin pohjalta päätettiin luoda käyttäjähaastattelu, jossa haluttiin vastaus kysymykseen, jos ohjeiden pohjalta osataan toimia. (Liite 1.) Haltun työntekijöiden joukosta valittiin muutama vapaaehtoinen projektijohtajiksi omissa projekteissaan ja kirjoittamaan päiväkirjaa jokaiselta päivältä kahden iteraation ajan.

Projektiryhmässä haluttiin saada käsitys siitä, minkä työntekijät kokevat olevan projektijohtamista ja mitä se on heiltä vaatinut. Käyttäjähaastattelussa käytettiin havainnoin menetelmää eli käyttäjien toimien seuraamista aidoissa käyttötilanteissa. Tällä tavalla saadaan yleistuntumaa käyttäjien toiminnasta. Tutkija esiintyy tutkimuksessa passiivisena havainnoitsijana eli seuraa käyttäjän toimia niihin puuttumatta. Tutkijana toimin itse ja vapaaehtoisiksi saatiin hiljattain aloittaneita ohjelmistoalan työntekijöitä.

15.1 Käyttäjähaastattelun toteutus

Käyttäjille kerrattiin käyttäjähaastattelun tavoitteet. Käyttäjät lukivat käsikirjaan kirjoitetun sisällön kokonaisuudessaan itsenäisesti läpi. Tutkija selvitti käyttäjien lähtöpisteen ryhmätyöskentelystä heidän projekteissaan sekä roolin kyseisissä projekteissa, koska roolilla on vaikutusta ketterän ohjelmistokehittämisen näkökulmasta. Projektijohtaja esimerkiksi fasilitoi keskustelua ja pitää huolta työntekijöiden mahdollisista esteistä projektityöskentelyyn, kun taas projektissa oleva kehittäjä osallistuu näihin tapahtumiin osallistujana.

Käyttäjähaastatteluun osallistuneet pitivät päiväkirjaa kahden heidän nykyisen projektin iteraation ajaksi. Iteraation laajuudesta riippuen käyttäjähaastattelu kesti kahdesta neljään viikkoa, jonka aikana kirjattiin päivittäin käyttäjien oman käsityksen mukaisesti päivittäiset tapahtumat, joissa he ovat projektijohtaneet projektissa.

Haasteena käyttäjähaastattelussa oli mittareiden asettaminen ja niiden seuraaminen, koska tutkijana en todellisuudessa kyennyt seuraamaan jokaista käyttäjää yksityiskohtaisesti tapahtumasta tapahtumaan ja selviä mittareita ei asetettu, joten haastattelusta saatavat tulokset ovat täysin tuotetun päiväkirjan varassa.

Tutkijan näkökulmasta käsikirjan sisällön keskeinen tavoite oli saada käyttäjä sisäistämään tärkeimmät kohdat projektinhallinnasta Haltun mukaisesti. Lukijan tulisi ymmärtää projektin iteraation tavoite ja siellä olevan listattuna sen iteraation työjonon tehtävät. Kaikki tapahtumat ja kirjaukset tämän jälkeen ovat ylimääräisiä työkaluja, jotka auttavat projektinhallinnassa tarvittaessa.

15.2 Käyttäjähaastattelun lopputulokset

Suurin osa käyttäjähaastatteluun osallistuneista ymmärsi käsikirjan sisällön keskeisimmät tavoitteet, mutta päiväkirjaa lähdettiin myös täyttämään Scrum-oppaan tapahtumien mukaisesti eli listattiin tapahtumat, joissa oli oltu mukana, mikä ei ollut haastattelun tavoite. Ohjeita kirjoittamiselle osallistuneille ei ollut, joten näkökulma ei ollut väärä ja antoi paremminkin palautetta siitä, että käsikirjan teksteistä tulisi selvemmin käydä ilmi, että työ tapahtumien ulkopuolella on myös projektijohtamista.

Tekstin yhtenäisyyden ja ulkoasun palautteen lisäksi suurin epäkohta oli valmiin määritelmän selkeän kuvauksen puuttuminen ja miten se yhdistetään projektijohtamiseen. Lisäksi asiakkaan ja tuoteomistaja roolien kuvaaminen sekä niiden välisten erojen selventäminen sai palautetta.

Muutokset otettiin käsittelyyn ja tehtiin aikataulu käsikirjan tekstin uudelleen julkaisemisesta. Käyttäjähaastattelua pidettiin onnistuneena kokonaisuutena, josta saatiin arvokasta palautetta, mutta sen lisäksi myös tietoisuutta ja keskustelua sen ympärille. Käsikirjaan tuotettuun sisältöön oltiin myös erittäin tyytyväisiä ja ei koettu tarpeellisiksi jatkaa projektia, vaikka tekstit ja käytännöt elävät aina aikojen sekä tapojen mukaisesti. Tällä hetkellä hyväksi koettu sisältö voi olla myö-

hemmin vanhentunutta, koska ihmisten työtavat muuttuvat ja kehittyvät ajan kuluessa. Onkin tarkoitus palata vuosien päästä kirjoitettuun sisältöön ja tarkastaa sen ajankohtaisuus. Projektiryhmässä nähdään kuitenkin nykyisten käytäntöjen ja tekstien kestävän ajan kulun tarpeeksi pitkälle.

16 POHDINTA

Ohjelmistokehityksen ala on jatkuvasti kehittyvä ja nouseva. Varsinkin pienen yrityksen näkökulmasta on haasteellista vaihtaa käytäntöjä ja kallista kouluttaa henkilöstöä. Sisäinen ja ulkoinen kommunikaatio eri projekteissa on tärkeää, jotta voidaan pysyä aikatauluissa budjetin rajoitteilla. Käytäntöjen puute tai vääränlaiset menetelmät, joista ei ole yhteistä ymmärrystä, vaikeuttaa projektityöskentelyä ja tekee toiminnasta haastavaa. Yhteisesti sovitut käytännöt ja suunniteltu kokonaisuus yhtenäistää toimintatapoja, mikä mahdollistaa vapaan liikkuvuuden sekä oikeasti joustavan tavan tuottaa ohjelmistoa. Joustava malli kestää ajan kulun jatkuvasti vaihtuvan alan alla. Yksinkertaiset ohjeet, jotka on kirjoitettu käytännön tilanteet huomioiden tarpeeksi matalalle tasolle nopeuttaa käyttöönottoa ja antaa pohjan asiasta kokemattomillekin toimia omatoimisesti.

Työssä on hyödynnetty laajalti projektijohtamisen teorioita ja hyödynnetty olemassa olevia tutkitusti hyödyllisiä ketterän ohjelmistokehittämisen malleja. Ohessa tutkitaan myös ohjelmistoalan historiaa ja eri menetelmien teoreettisia näkökulmia projektiryhmän toimintaa varten. Lähdemateriaalia on valtavasti eri teoreettisista näkökulmista ja eri vuosikausilta. Teoria on kestänyt ajan kulun, koska pohjateoria ja vanhat päätelmät ovat nykypäivänäkin hyödyllisiä, koska tämän päivän menetelmät on rakennettu vanhojen teorioiden pohjalta. Lisäksi lähdemateriaalina käytetään Haltun sisäistä kirjallisuutta, joka on luotu oikeiden tilanteiden ja tarpeiden pohjalta vuosien ajalta. Vanhoja teorioita hyödynnetään ja ketterän ohjelmistokehittämisen malleista otetaan menetelmiä käyttöön, mutta samalla pohditaan kriittisesti niiden joustavuutta. Käytännön tarpeista ja opinnäytetyön aikana toteutetuista haastatteluista johdettujen johtopäätöksien avulla muokattiin vanhaa teoriaa ja luotiin uudet käytännöt, jotka sopivat tässä asiasisällössä toteutetulle yritykselle.

Opinnäytetyön mallien käyttöönotto osaksi toimintaa on luontaista johtuen pitkäjänteisestä kehittämisestä, joka herätti kiinnostusta ja pohjatietoa tulevalle. Työn aikana on hyödynnetty kaikkia mahdollisia saatavia resursseja yrityksen sisältä ja haastateltu toteutetun mallin käyttäjiä. Tuotettua työtä voidaan hyödyntää pit-

källe tulevaisuuteen ja heti käyttöön projektityöskentelyn vaativalla tavalla. Todellinen hyöty näkyy tulevaisuudessa saataessa enemmän aineistoa käyttäjistä ja asiasta ennestään aiheeseen kokemattomilta tulevaisuuden työntekijöitä.

Opin opinnäytetyön aikana valtavasti projektijohtamisen eri käytännöistä, ohjelmistokehittämisen malleista ja projektityöskentelystä. Työ laajensi omaa käsitystä projektijohtajan roolista ja kuinka tärkeää yhteiset käytännöt ovat ohjelmistokehityksessä. Teorian kautta opin historiaa ja sain lähtötietoa ketterän ohjelmistokehittämisen mallien lähtöteoriasta. Lisäksi sain käytännön kokemusta projektijohtajan roolista sekä vastuista ja sain näkökulmaa ohjelmistokehityksen eri prosesseille sekä asiakassuhteen ylläpitoon. Kokonaisvaltainen ymmärrykseni ja osaamiseni alasta sekä projektitoiminnasta on kasvanut. Työn edetessä kehitin myös omia kommunikaation taitoja sisäisessä ja ulkoisessa viestinnässä.

Katsottuna taaksepäin, projektiryhmän työskentelyssä ja sen toimintatavoissa oli asioita, jotka tekisin nyt toisin. Epäonnistumiset ja työn aikana koetut vaikeudet ovat toisaalta toimineet hyvänä oppina ja muokanneet opinnäytetyötä parempaan suuntaan, kuin mitä se olisi ollut ilman vaikean kautta opittuja oppeja. Projektijohtajana toimiminen mallin kehittämisen työryhmässä on jälkeenpäin katsottuna ollut hyvä kokemus ja näkökulma työlle, vaikka aluksi rooliin asettuminen epäilytti. Kokonaisvaltaisuudessaan olen tyytyväinen opinnäytetyön tulokseen ja siitä opittuihin oppeihin.

LÄHTEET

Cattani, G. Ferriani, S. Frederiksen, L. & Florian, T. 2011. Project-Based Organizing and Strategic Management. Advances in Strategic Management. Vol. 28.

Phillips, Joseph (2004). PMP Project Management Professional Study Guide

Wesland, J, 2022. 15 Essential Project Documents. Projectmanager-blogi 5.8.2022. Viitattu 5.6.2023. https://www.projectmanager.com/blog/great-project-documentation

AdaptiveWork 2021. What Are the Objectives of Project Management? Planview-blogi 13.4.2021. Viitattu 4.6.2023. https://blog.planview.com/objectives-of-project-management

PMI. n.d. What is Project Management. Verkkosivu. Viitattu 5.6.2023. https://www.pmi.org/about/learn-about-pmi/what-is-project-management

Angelo, B. 2006. The Triple Constraint a Triple Illusion. Pdf-dokumentti. Viitattu 5.6.2023. https://www.ucipfg.com/Repositorio/MAP/MAPD-05/BLOQUE-ACADEMICO/UNIDAD5/lectura1.pdf

Kwak, Y. 2021. A Brief History of Project Management. Pdf-dokumentti. Viitattu 5.6.2023. https://home.gwu.edu/~kwak/PM_History.pdf

Cleland, D. & Roland, G. 2006 Global Project Management Handbook.

Azzopardi, S. n.d. The Evolution of Project Management. Projectsmart n.d. Viitattu 5.6.2023 https://www.projectsmart.co.uk/history-of-project-management/evolution-of-project-management.php

Firsthand. n.d. Project-management. Verkkosivu. Viitattu 5.6.2023. https://firsthand.co/industries/project-management/background

Miller, W. 1962. How to Plan and Control With PERT.

Levy, K. & Thompson, L. 1963. The ABCs of the Critical Path Method.

Haughey, D. 2010. History of Project Management. Projectsmart 2010. Viitattu 5.6.2023. https://www.projectsmart.co.uk/history-of-project-management/brief-history-of-project-management.php

Sheldrake, J. 2003. Henry Gantt and Humanized Scientific Management. Management Theory.

Witzel, M. 2003. Fifty Key Figures in Management. Rougbert 2003.

Wysocki, R. 2013. Effective Project Management: Traditional, Adaptive, Extreme. John Wiley & Sons

Kissflow. 2023. 5 Phases of Project Management – A Complete Breakdown. Verkkosivu. Viitattu 5.6.2023. https://kissflow.com/project/five-phases-of-project-management/

U.S Office of Personnel. 2003. Management.Interpretive Guidance for IT Project Manager Psotions. Pdf-dokumentti. Viitattu 5.6.2023. https://www.opm.gov/policy-data-oversight/classification-qualifications/reference-materials/projectmanager.pdf

Agilealliance. n.d. What is Agile? Verkkosivu. Viitattu 5.6.2023. https://www.agilealliance.org/agile101/

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

Sacolick, I. 2022. A Brief History of the Agile Methodology. Verkkosivu. Viitattu 5.6.2023. https://www.infoworld.com/article/3655646/a-brief-history-of-the-agile-methodology.html

Lynn. R. n.d. The History of Agile. Verkkosivu. Viitattu 5.6.2023. https://www.planview.com/resources/guide/agile-methodologies-a-beginners-guide/history-of-agile/

Scrum n.d. What is Scrum? Verkkosivu. Viitattu 5.6.2023. https://www.scrum.org/resources/what-scrum-module

Knowledgehut. 2023. What is Scrum Methodology. Verkkosivu. Viitattu 5.6.2023. https://www.knowledgehut.com/tutorials/scrum-tutorial

Schwaber, K. 2004. Agile Project Management with Scrum. Microsoft Press.

Lean n.d. a Brief History of Lean. Verkkosivu. Viitattu 5.6.2023. https://www.lean.org/explore-lean/a-brief-history-of-lean/

Koskinen, I. 2021. Mikä on Kanban? Katsaus menetelmään ja sen käyttöön ketterässä projektinhallinnassa. Verkkosivu. Viitattu 5.6.2023. https://severa.fi/blogi/mika-on-kanban-katsaus-menetelmaan-ja-sen-kayttoon-ketterassa-projektinhallinnassa/

Kanbantool n.d. Kanban History. Verkkosivu. Viitattu 5.6.2023. https://kanbantool.com/kanban-guide/kanban-history

LIITTEET

Liite 1. Haltun käyttäjätutkimuksessa kysytyt kysymykset

Käyttäjähaastattelun kysymykset

- Saitko sisällöstä selvää, mitä Haltun toteuttama ketterä menetelmä on?
- Mitä käytäntöjä menetelmään kuuluu?
- Ymmärrätkö eri roolien tarkoituksen?
- Mitä tapahtumia voidaan pitää?
- Osaisitko toimia projektijohtajana ohjeiden perusteella?
- Millaisessa projektissa työskentelet?
- Kuinka aktiivinen asiakas projektissa on?
- Oletko toiminut projektijohtajana aikaisemmin?
- Mitä ketterä ohjelmistokehittämisen malli on sinulle?
- Millä tavalla olet toiminut projektijohtajan asemassa?