Modum tuotantoympäristön tuotekehitysversioiden A/B vertailu

Jori Nevalainen

2020 Laurea

Laurea-ammattikorkeakoulu

Modum tuotantoympäristön tuotekehitysversioiden A/B vertailu

Jori Nevalainen
Tietojenkäsittelytradenomi
Opinnäytetyö
Marraskuu, 2020

Laurea-ammattikorkeakoulu **Tiivistelmä**
Liiketalouden ammattikorkeakoulututkinto
Tradenomi (AMK)

Jori Nevalainen

Modum tuotantoympäristön tuotekehitysversioiden A/B vertailu

Vuosi 2020 Sivumäärä 41

Tämän työn toimeksiantaja oli Modum Oy, jonka päätoimiala on käytettyjen autojen kuvaus. Tutkimuksen tavoitteena oli havainnoida autojen kuvaus ja editointi prosessin kestoa. Tarkoitus oli selvittää vertailuarvo sekä löytää ongelmia ja ratkaisuja prosessiin, jota on tehty useamman vuoden ajan. Tarkoituksena oli tuoda päivitys editointisovellukseen sekä testata uusi kuvausstudio.

Teoriassa käydään läpi alalla vallitsevista ohjelmistokehityksen metodeista. Eri metodit pitää sisällään erilaisia kehyksiä, työkaluja ja rooleja. Teoria osuudessa selitetään eri metodit ja niiden ominaispiirteet.

Empiirinen tutkimus oli triangulaatio tutkimus ja siinä keskityttiin kyseisen työvaiheen tarkasteluun. Tiedonhankintamenetelminä toimivat havainnointi sekä haastattelut. Pääpaino on havainnoinnilla ja haastattelu on taka-alalla. Näiden avulla kerättiin tietoa prosessin ongelmista, sen kehittymisestä sekä parannusehdotuksista.

Tuloksena saatiin tarkkaa tietoa prosessin lähtökohdasta sekä sen kehittymisestä. Näiden ohessa syntyi kehitysideoita, joiden avulla prosessia voisi jatkokehittää. Tutkimuksen tulosten arviointi oli helppoa, sillä se esitettiin suoraan numeroilla sekä kaavioita apuna käyttäen. Prosessivaiheissa tapahtunut kehitys oli positiivista ja selkeästi huomattavissa. Editointiin kulunut aika väheni 40 % ja sen lisäksi myös työvaiheen käyttäjäystävällisyys kehittyi.

Asiasanat: ohjelmistokehitys, havainnointi, tutkimus triangulaatio, tilastollisesti kuvaava

Laurea University of Applied Sciences **Abstract**
Business Information Technology
Bachelor of Business Administration

Jori Nevalainen
Modum Production Environment Development Versions A/B Comparison
Year 2020 Pages 41

The client of this bachelor's thesis study was Modum Ltd, whose main business is photographing used cars. The aim of the research was to observe the duration of the photograph and editing process. The aim was to examine the benchmark and to find problems and solutions to a process that has been going on for several years. Intend was to bring an update to the editing application and test the new photograph studio.

In theoretical section, prevailing methods in the software development field are reviewed. Different methods involve different frameworks, tools, and roles. The theoretical section explains the different methods and their characteristics.

The empirical study was a triangulation study and focused on examining that work stage. The methods used to obtain information were observation and interviews. The main focus was on observation and the interview was in the background. These were used to gather information about the problems of the process, its development and suggestions for improvement.

The result was accurate information about the starting point of the process and its development. Alongside these development ideas, emerged ideas that could be used to further develop the process. The evaluating the results of the study was easy as results were presented directly with numbers as well as with the help of diagrams. Developments in the process stages was positive and clearly significant. The time spent on editing was reduced by 40%, and in addition, the user-friendliness of the work stage was also improved.

Keywords: software development, observation, triangulation research, statistically descriptive

Sisällys

1 Johdanto

Ohjelmistokehitysmetodeja on monia erilaisia. Moni niistä tulee projektinhallinamenetelmistä. Ohjelmistokehitysmenetelmiä tutkitaan paljon ja puntaroidaan mikä menetelmä sopii mihinkin projektiin. Näitä metodeja muokataan ajan saatossa sopimaan paremmin nykypäivän haasteisiin. Teoriaosuudessa tutkitaan eri ohjelmistokehityksen metodeja ja niiden ominaispiirteitä.

Ohjelmiston kehittäminen on tärkeätä monesta syystä. Yleisimpiä ovat käyttäjäkokemuksen parantaminen, ominaisuuksien lisääminen tai työprosessin nopeuttaminen. Modum Oy:llä on tarkoituksena tuoda laaja päivitys editoinnissa käytettyyn sovellukseen. Päivityksen tavoitteena on kehittää näitä aikaisemmin mainittuja asioita. Päivitys tuo sovellukseen kuvien editointiominaisuuden. Tämä yhdistää kaiken editoinnin yhteen sovellukseen. Tarkoituksena on myös rakentaa uusi kuvausstudio, johon kaikki kuvaus siirretään. Modum haluaa selvittää työvaiheen keston ja kuinka se kehittyy päivitysten ansiosta.

Tutkija tulee testaamaan sovelluspäivitystä kesken kehityksen ja myös sen jälkeen bugien varalta ja antaakseen palautetta päivityksestä. Palautteen avulla voidaan vielä tehdä muutoksia ennen kuin lopullinen versio julkaistaan. Julkaisun jälkeen tarkkaillaan kuinka uusi sovellusversio vaikuttaa työprosessin kestoon ja käyttäjäystävällisyyteen. Saatua dataa käsitellään, jotta saadaan tarkkaa tietoa siitä mitkä osa-alueet kehittyivät ja kuinka paljon. Tämän tiedon perusteella yritys voi tehdä päätöksiä tulevista päivityksistä prosessiin. Tutkimuksen avulla on tarkoitus selvittää prosessin kesto ennen päivitystä sekä siinä piileviä ongelmia ja parannusideoita prosessiin. Päivityksen jälkeen on tarkoitus päivittää prosessin kesto sekä selvittää sen kehittyminen.

2 Lähtökohdat

Tutkija on perehtynyt asiakasyritykseen työharjoittelussaan sekä sen jälkeisessä työssä vuoden 2018 lopusta vuoden 2020 puoliväliin saakka. Tutkijan mielenkiinto aihetta kohtaan kehittyi työharjoittelun toisen vaiheen alussa. Tutkija sopi asiakasyrityksen kanssa, että hän voi kirjoittaa tästä aiheesta opinnäytetyön. Asiakasyrityksen edustaja sanoi jo odottavansa innoissaan saadakseen lukea kyseisen opinnäytetyön. Aika on ajankohtainen sillä yrityksessä tulee tapahtumaan paljon muutoksia lähitulevaisuudessa.

2.1 Tutkimuksen kohde

Tutkimus suoritettiin yrityksessä Modum Oy. Modum on suomalainen kasvuyritys, joka on perustettu vuonna 2017. Yrityksen kotipaikka on Espoo. Modum on yritysmuodoltaan osakeyhtiö. Yrityksen päätoimiala on käytettyjen autojen kuvaus.

Modum tarjoaa uudenlaisia video- ja valokuvauspalvelukokonaisuuksia. Palvelut ovat korkealuokkaisia ja ne on tarkoitettu verkkokauppatuotteiden kuvaamiseen ja esittelyyn. Palvelun tuotteisiin kuuluu korkealaatuiset still -kuvat, 360° kuvat ja videot, nämä yhdistettynä laajaan ohjelmistokokonaisuuteen. Tämä ohjelmistokokonaisuus mahdollistaa laadukkaan ja tehokkaan medianjakelun loppukäyttäjille. Modumin tehtävä on kehittää ammattimaista ja korkealaatuista kuvauspalvelua ja kuvausteknologiaa. Yrityksen näkemys on tuotteiden realistinen havainnointi verkossa.

Modumin liikevaihto oli perustamisvuonna 24 000 €. 2018 liikevaihto nousi 43 000 €:on. Viime vuonna liikevaihto oli jo 170 000 €. Liikevoittoa ei ole kertynyt ensimmäisenä kolmena vuotena. Liiketappiota tuli vuonna 2017 15 000 €, 2018 61 000 € ja 2019 14 000 €.

Modumilla on tällä hetkellä vain yksi täysipäiväinen työntekijä. Yritys hyödyntää työtehtävissään paljon freelancereita. Kuvaajat, editoijat, myyjät, koodaaja, tilintarkastaja sekä palkanlaskija ovat freelancereita. Modum on yritysmuodoltaan yksityinen osakeyhtiö ja sen hallitukseen kuuluu kolme jäsentä.

2.2 Työprosessi ja sen ongelmat, tutkimus kysymykset ja aihealueen rajaus

Modum Oy kuvaa asiakkaiden autoja heidän toimitilassaan. Asiakasyritys valmistelee autot kuvattavaksi, jonka jälkeen kuvaajaa hakee auton ja vie sen Modumin rakentamaan studioon. Studiossa kuvaaja asettelee auton kuvauskuntoon ja ottaa autosta ulkokuvat, sisäkuvat ja videon. Tämän jälkeen kuvaaja vie auton pois ja hakee uuden auton kuvattavaksi.

Autojen kuvaamisen jälkeen, kuvaaja tai editoija editoi kuvat ja leikkaa videon. Kuvien editointi tapahtuu Windowsin omalla editointiohjelmalla. Kun kuvat on editoitu, viedään kaikki media Modumin sovellukseen. Sovelluksessa tapahtuu videon leikkaus ja median lähettäminen Modumin omalle palvelimelle. Sovellus luo kuvista myös kevyemmät versiot, jotka editoija itse lähettää asiakasyrityksen verkkosivuille.

Ongelmana oli, ettei Modumilla ollut tarkkaa tietoa, kuinka kauan mikäkin prosessivaihe kestää. Henkilöiden, jotka eivät kuvaa tai editoi autoja oli vaikea tietää mitkä vaiheet vievät paljon aikaa ja missä kohdissa olisi parannettavaa. Se oli ainakin tiedossa, että editointivaiheessa useamman sovelluksen käyttäminen on huono asia ja tarkoituksena oli, että kaikki vaiheet tehtäisiin yhdessä sovelluksessa.

Tutkimuksen peruskysymys on, miten autojen kuvaus ja editointiprosessit kehittyvät? Apukysymyksiä ovat, miten selvitetään prosessivaiheiden kestot, käyttäjäystävällisyys sekä ongelmat? Tavoitteena oli jakaa prosessi mahdollisimman pieniin vaiheisiin ja selvittää jokaisessa vaiheeseen kuluva aika, sen mahdolliset ongelmat ja mahdolliset parannuskeinot.

Tavoitteena oli kasvattaa kuvauksen volyymia; kuinka monta autoa kuvaaja voi kuvata ja editoida päivässä. Oli jo tiedossa, että kuvattavien autojen määrä tulee kasvamaan lähikuukausina. Jotta yksi henkilö pystyy vielä hoitamaan tämän työtehtävän, oli työprosessia kehitettävä. Työprosessista oli tarkoitus tehdä nopeampi, sujuvampi ja käyttäjäystävällisempi sekä selkeämpi, jotta uudet työntekijät omaksuisivat prosessin nopeammin. Työprosessin sujuvuutta tarkkailtiin ennen ja jälkeen muutosten. Tarkasteltiin mitkä prosessivaiheet kehittyivät eniten ja missä olisi vielä kehitettävää. Aihetta tutkittiin kuvaajan näkökulmasta itse tekemällä kyseistä työtä ja haastattelemalla työntekijää.

3 Ohjelmistokehityksen metodit

Projektinhallintamenetelmiä on paljon erilaisia. Eri menetelmät toimivat paremmin eri projekteissa siksi onkin tärkeää löytää hyvä menetelmä omalle projektille. Toiset menetelmät rytmittävät työn erittäin tarkasti, kun taas toiset ohjeistavat vain hieman oikeaan suuntaan ja antavat joitakin hyviä sääntöjä. Eri menetelmiä on mahdotonta laittaa paremmuusjärjestykseen, mutta niissä kaikissa on omat sääntönsä, jotka soveltuvat toisiaan paremmin eri tilanteissa. Joskus yksi yksittäinen menetelmä ei ole paras vaihtoehto, silloin voi kokeilla yhdistää useampaa menetelmää. Valitsemalla eri menetelmistä niiden parhaat puolet kyseiseen projektiin voi tarjota paremman työnjaottelun kuin mitä yksi menetelmä pystyisi tarjoamaan. (Pulkkanen 2020.)

3.1 Historia

1940-luvulla otettiin toimintaan ensimmäiset digitaaliset tietokoneet. Kyseisiä tietokoneita operoitiin vaihtamalla niiden sähköisiä kytkentöjä. 1940-luvun loppupuolella kehitettiin ensimmäiset ohjelmistokehitys- ja tietojärjestelmähankkeet. Tämän mahdollisti, Von Neumannin luoman arkkitehtuurin mukaisesti kehitetyt ohjelmoitavat tietokoneet. Aikaisemmin digitaalisten tietokoneiden ohjelmat luotiin konekielellä. Tämän takia ohjelmat olivat kahlehdittu laitteeseen, jolle ne oli luotu. Ohjelmointikielten ilmestymisen jälkeen, ohjelmia voitiin aloittaa kehittää eri laitteille. (Haikala & Mikkonen 2011.)

Noin 10 vuotta ohjelmointikielten rantautumisen jälkeen ohjelmistokehitys kohtasi ohjelmistokriisin. Koska alalla oli pulaa selkeistä työmenetelmistä sekä ohjelmoijista. Tämän takia ohjelmat sisälsivät paljon virheitä, tuottavuus laski ja kustannukset nousivat. Näiden ongelmien seurauksena ohjelmistotuotantoa kehitettiin ratkaisemaan alan ongelmat. Tavoitteena oli nostaa ohjelmien laatua ja minimoimaan virheet. Ohjelmien kehittäminen voi olla mittava ja mutkikas hanke. Jotta voidaan maksimoida tuotannon tehokkuus, tulee kehitystyö eritellä osiin. Ohjelmien suunnittelun jakamista osiin kutsutaan vaihejakomalliksi. (Haikala & Mikkonen 2011.)

3.2 Ketterät menetelmät

Ketterä ohjelmistokehitys on kyky luoda ja vastata muutoksiin. Se on tapa käsitellä epävarmaa ja sekasortoista ympäristöä, lopulta onnistuen siinä. Ketterän ohjelmistokehityksen manifestin kirjoittajat valitsivat ketterän koko ketjun etiketiksi, koska tämä sana edustaa heidän lähestymistapaansa sekä tärkeää sopeutumiskykyä, että vastausta muutokseen. Tarkoituksena on todella miettiä kuinka voit ymmärtää ympäristön tilanteen, tunnistaa minkälainen epävarmuus on hyväksyttävää ja selvittää, kuinka voit mukautua siihen edetessäsi. (Agile Alliance.)

Ketterä ohjelmistokehitys ei ole vain kehys; Extreme Programming (XP), Scrum tai Dynamic Systems Development Method (DSDM). Ketterä ohjelmistokehitys sisältää myös käytäntöjä, kuten stand-up kokoukset, sprintit, pariohjelmointi, suunnitteluistunnot ja testiohjattu kehitys. Ketterä ohjelmistokehitys on kattotermi käytännöille ja kehyksille perustuen arvoihin ja periaatteisiin, jotka on ilmaistu ketterän ohjelmistokehityksen manifestissa ja sen taustalla olevissa 12 periaatteessa. (Agile Alliance.)

Ohjelmistokehitystä lähestyessä, on yleensä hyvä elää paikallisten arvojen ja periaatteiden mukaisesti ja käyttää niitä apuna kyseisten tehtävien selvittämisessä. Ketterä kehitys erottuu muista lähestymistavoista ohjelmistokehitykseen, keskittymällä työntekijöihin ja yhteisiin työtapoihin. Ketterässä ohjelmistokehityksessä kiinnitetään paljon huomiota itseorganisointiin ja yhteistyöhön. Projektiin kuuluu silti johtaja, mutta hänen työtehtävänsä eroavat verrattuna muihin projektinhallintamenetelmiin. Itseorganisoinnin ansiosta ryhmällä on mahdollisuus yhdessä selvittää miten he aikovat lähestyä tilannetta. Tämä tarkoittaa, että ryhmä on omavarainen. Ryhmän jäsenille ei anneta rooleja. Projektin johtaja vain varmistaa, että ryhmällä on tarvittavat tietotaidot projektin suorittamiseksi. (Agile Alliance.)

Projektinjohtajia tarvitaan vielä kuitenkin. Johtajat tehtävä on varmistaa, että tiimin jäsenillä on tai he saavat tarvittavan tietotaidon. Johtajat luovat ympäristön, jossa ryhmällä on paras mahdollisuus menestyä. Johtajat eivät yleensä sekaannu projektin suorittamiseen vaan antavat ryhmälle mahdollisuuden selvittää, miten he toimittavat tuotteita. Johtajien tehtävänä on tulla apuun, kun ryhmä yrittää ratkaista ongelmia onnistumatta niissä. (Agile Alliance.)

Kun organisaatio tai ryhmä aloittaa ketterän ohjelmistokehityksen, on heillä tapana keskittyä käytäntöihin, jotka auttavat työn organisoinnissa ja yhteistyössä. Tämä on hyvä asia, mutta toinen keskeinen käytäntöjoukko, jota taas ei noudateta usein, on tiettyjen teknisten käytäntöjen käyttäminen. Nämä käytännöt voisivat auttaa ryhmää organisoimaan toimintaa ja käsittelemään ryhmälle vaikeaa aihetta. Kyseiset tekniset käytännöt ovat hyvin tärkeitä eikä niitä kannata jättää pois. (Agile Alliance.)

3.2.1 Scrum

Scrum on yksi ketterän ohjelmistokehityksen prosessikehyksistä. Scrumia käytetään tietotyön johtamisessa sekä tuotekehityksessä. Scrum on kehitetty tavalla, joka mahdollistaa muiden prosessikehyksien käytäntöjen lainaamisen, edellyttäen että ne sopivat projektin kontekstiin. Scrum on kehyksenä kokeellinen. Tämän ansiosta se tarjoaa käyttäjille hypoteesin siitä, kuinka heidän mielestään kyseinen asia toimii. Tämän ansiosta he voivat toteuttaa hypoteesin. Saadun kokemuksen perusteella he voivat tehdä hypoteesiin tarvittavat säädökset. Mutta kyseinen operaatio toimii vain, jos kehystä käytetään oikein. (Agile Alliance.)

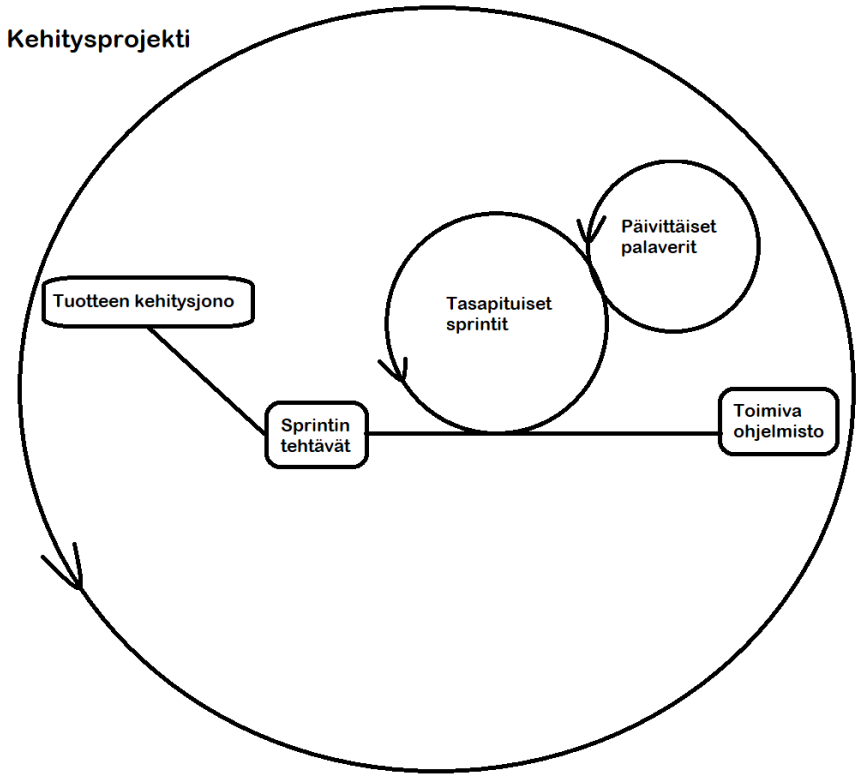
Scrum kehitettiin 1990-luvulla monimutkaisten tuotteiden työn hallintaan. Scrum on ominaisuuksiltaan kevyt, helppo ymmärtää, mutta vaikea hallita. Scrum itsessään ei ole tekniikka tai prosessi vaan kehys, jossa voi käyttää erilaisia tekniikoita tai prosesseja. Scrumin avulla pystyy selvittämään työtekniikoiden sekä tuotejohtamisen tehokkuuden. Tämän ansiosta käyttäjät voivat jatkuvasti parantaa tiimiään, työympäristöään sekä tuotettaan. (Scrum Guide.)

Scrum -kehys koostuu Scrum -tiimeistä sekä niihin liittyvistä tapahtumista, säännöistä, rooleista ja esineistä. Scrumin käytölle on välttämätöntä, että jokainen komponentti kehyksessä palvelee tiettyä tarkoitusta. Scrumin ytimenä on pieni tiimi ihmisiä. Pieni tiimi on hyvin joustava ja mukautuva. Tiimi koostuu kehitysryhmästä, scrummasterista sekä tuotteen omistajasta. Itsenäisesti toimivat tiimit voivat itse päättää, mikä on paras tapa työn suorittamiseen, sen sijaan että joku ulkopuolinen ohjaisi ryhmän toimintaa. Monialaiset tiimit ovat omavaraisia eivätkä ole riippuvaisia tiimin ulkopuolisista toimijoista. Scrumin ryhmämalli on suunniteltu optimoimaan tuottavuus, luovuus ja joustavuus. Tiimit tuottavat tuotteita, jatkuvasti ja asteittain maksimoidakseen tuotteesta saatavan palautteen. (Scrum Guide.)

Kehitysryhmälle optimaalinen kokoa on kolmesta yhdeksään jäsentä. Jos ryhmässä on alle 3 jäsentä, laskee ryhmän tuottavuus ja vuorovaikutus. Mutta jos ryhmässä on yli 9 jäsentä, ryhmän koordinointi hankaloituu. Pienempien ryhmien ongelmana voi olla tietoa ja taitoja koskevia rajoituksia, jonka seurauksena kehitysryhmä ei pysty tuottamaan tuotetta. Suuret ryhmät saattavat hyödyntää liikaakin kokeellista prosessia ja sen takia monimutkaistaa projektia. (Scrum Guide.)

Scrummasterin tehtävänä on tukea ja edistää projektia. Hän toteuttavat tätä auttamalla ryhmän jäseniä ymmärtämään Scrumin arvoja, sääntöjä, teoriaa ja käytäntöjä. Hänen tehtävänsä on myös ohjeistaa ryhmän ulkopuolisia toimijoita hyödylliseen kommunikaatioon ryhmän kanssa, jotta vältytään turhalta vuorovaikutukselta. Scrummasteria voisi kuvailla palvelijajohtajaksi. Hän on ryhmän johtaja, mutta hänen työtehtävänsä on suurimmaksi osaksi ryhmän palveleminen sen tarpeiden mukaisesti. (Scrum Guide.)

Scrumissa käytetään ennalta määrättyjä tapahtumia kokouksien vähentämiseksi ja säännöllisyyden luomiseksi. Kaikki Scrumin tapahtumat on aikataulutettu, niin että niillä on aina enimmäiskesto. Scrumissa paljon käytetty sprintti on projekti, jonka kesto on enintään kuukausi. Sprintillä on aina tavoite mitä valmistetaan, miten suunnitellaan ja joustava suunnitelma, joka ohjaa työtä. Sprintin alkaessa sen kestoa ei voida enää pidentää tai lyhentää. Jos sprintin tavoite on saavutettu etuajassa, jäljelle jääneet tapahtumat voivat päättyä, jotta ei turhaan tuhlata aikaa prosessissa. Tämän jälkeen aloitetaan uusi prosessi alusta (Kuvio 1). (Scrum Guide.)



Kuvio 1: Scrum projektin rakenne

Päiväpalaveri on aikataulun mukainen tapahtuma, joka pidetään Scrumin jokaisena päivänä. Se on 15 minuutin pituinen palaveri, johon osallistuu kehitysryhmä. Päiväpalaverissa kehitetään työsuunnitelma seuraavalle 24 tunnille. Palaverilla pyritään optimoimaan tiimin yhteistyötä ja suorituskykyä tarkastelemalla aikaisemman päivän työtä sekä ennustamalla tulevaa työtä sprintissä. Työnkulun yksinkertaistamiseksi päiväpalaveri pidetään aina samaan aikaan samassa paikassa joka päivä. Kehitysryhmä käyttää päiväpalaveria tarkistaakseen kehityksen kohti sprintin tavoitteita sekä tarkistaakseen kehityksen trendin kohti kehitysjonon valmistumista; kehitysjono on lista kaikesta mitä tuotteessa saatetaan tarvita, se on myös ainut lähde muutoksille ja vaatimuksille. Päiväpalaverin tarkoituksena on parantaa, kehitysryhmän todennäköisyyttä saavuttaa sprintin tavoitteet. (Scrum Guide.)

3.2.2 Extreme Programming

Extreme Programming eli XP otettiin käyttöön 1990-luvulla. XP on yksi monista ketteristä ohjelmistokehityksen kehyksistä. XP on ilmennyt olevan hyvin tuloksellinen ympäri maailmaa erilaisissa yrityksissä. Menestyksen salaisuus on ollut asiakkaiden kuunteleminen. Sen sijaan että toimitettaisiin lopullinen tuote joskus pitkän ajan kuluttua, XP toimittaa tarvittavat ohjelmistot silloin kun niitä tarvitaan. XP:n ansiosta kehittäjillä on hyvä mahdollisuus vastata muuttuviin asiakasvaatimuksiin. (Wells 2013.)

Extreme Programming tiimissä kehittäjät, asiakkaat ja johtajat ovat tasa-arvoisia. XP:n tarkoitus onkin korostaa tiimityöskentelyä. XP luo tehokkaan, mutta yksinkertaisen työympäristön, joka parantaa ryhmän tuottavuutta. Ratkaistakseen ongelman mahdollisimman tehokkaasti, ryhmä organisoi itsensä ongelman ympärille. (Wells 2013.)

Extreme Programming kehittää ohjelmointiprojekteissa näitä osa-alueita: palaute, viestintä, rohkeus, kunnioitus ja kommunikointi. Palautetta saadaan testaamalla ohjelmistoa heti ensimmäisestä päivästä saakka. Ohjelma toimitetaan asiakkaille mahdollisimman aikaisin ja siihen tehdään asiakkaan pyytämät muutokset. Näiden perustuksien avulla ohjelmoijat pystyvät rohkeasti reagoimaan muuttuvaan teknologiaan ja vaatimuksiin. Jokainen pienikin onnistuminen, vahvistaa tiimin jäsenten välistä kunnioitusta, toistensa ainutlaatuisista työpanoksistaan. Kommunikointia tapahtuu ohjelmoijien ja asiakkaiden kanssa jatkuvasti. (Wells 2013.)

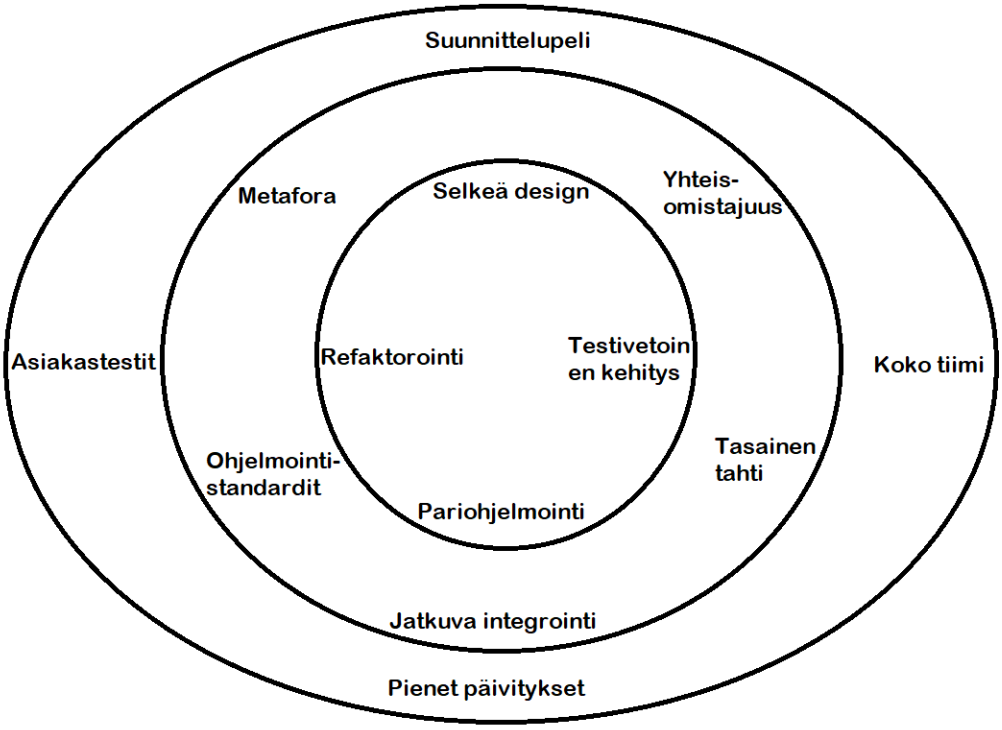
Yksinkertaiset säännöt saattavat yllättää Extreme Programmingissa. Ne voivat vaikuttaa naiiveilta tai hankalilta, mutta ne on luotu vakaisiin periaatteisiin ja arvoihin perustuen. Säännöt itsessään eivät ole päämäärä, vaan niiden on tarkoitus asettaa odotuksia tiimin jäsenten välillä. Vaikutusmahdollisuuksien ja ryhmätyön edistämiseksi nämä säännöt määrittelevät työympäristön. Kun tämä on saavutettu, sääntöjä voidaan muokata yrityksen omien tarpeiden mukaisiksi, pysäyttämättä tuottavaa tiimityöskentelyä. (Wells 2013.)

Extreme Programmingissa asiakas on vastuussa kaikesta projektia koskevasta liiketoimintapäätösten tekemisestä. Asiakkaan toivotaan liittyvän osaksi tiimiä ja osallistumaan projektiin aktiivisesti. Aikaisemmin on oletettu, että asiakkaana toimii yksi henkilö. Kokemus on todistanut, ettei yksittäinen henkilö pysty tarjoamaan kaikkea liiketoimintaan liittyvää tietoa, projektia varten. Tiimin tehtävänä on varmistaa, että projektin johtaja ymmärtää täysin liiketoiminnallisen osan projektista. (Agile Alliance.)

Extreme Programmingissa ei ole tarvetta tarkempaan roolimäärittelyyn, jonka takia suurin osa tiimin jäsenistä on rooliltaan kehittäjiä. Kehittäjien tehtävänä on tunnistaa asiakkaan tarpeet, asiakkaan antamasta kuvauksesta. XP kehys perustuu monialaiseen tiimiin, jossa on sopiva yhdistelmä eri taitoja. Projektit edellyttävät erilaisten taitojen yhdistelemistä, jonka takia kehittäjille ei tarvitse luoda tarkempia rooleja. (Agile Alliance.)

Mittaaja on valinnainen rooli Extreme Programming projektissa. Mittaajan tehtävänä on seurata tiimin määrittämiä mittareita. Niistä selviää projektin eteneminen sekä parannettavat alueet. Joitakin tiimille tärkeitä mittareita voivat olla: ylityön määrä, onnistuneet sekä epäonnistuneet testit, kehityksen vauhti sekä siinä tapahtuvat muutokset. Mittaajana toimii yleensä yksi kehittäjistä, joka kuluttaa osan työajastaan täyttämään tämän roolin tehtävät. Tosiaan mittaajan rooli ei ole pakollinen ja se yleensä määritetään, jos tiimillä on tarve seurata useita eri mittareita. (Agile Alliance.)

Valmentaja on myös valinnainen rooli Extreme Programmingissa. Valmentaja on hyödyllinen rooli silloin kun XP:tä ollaan vasta ottamassa käyttöön työympäristössä. Valmentaja on joku henkilö, joka on toiminut aikaisemminkin XP projektissa. Hänen pääarvonsa onkin kokemus kyseisistä toimintatavoista, joka auttaa tiimiä välttämään aloittelijoille sattuvia virheitä. Valmentajana yleensä toimii, joku tiimin ulkopuolinen henkilö on se sitten yrityksen työntekijä tai konsultti. Valmentajan tehtävänä on opettaa tiimille XP:n käytäntöjä (Kuvio 2), sekä auttaa ylläpitämään tiimin kurinalaisuutta. (Agile Alliance.)



Kuvio 2: Extreme Programming käytäntöjä

3.3 Lean IT

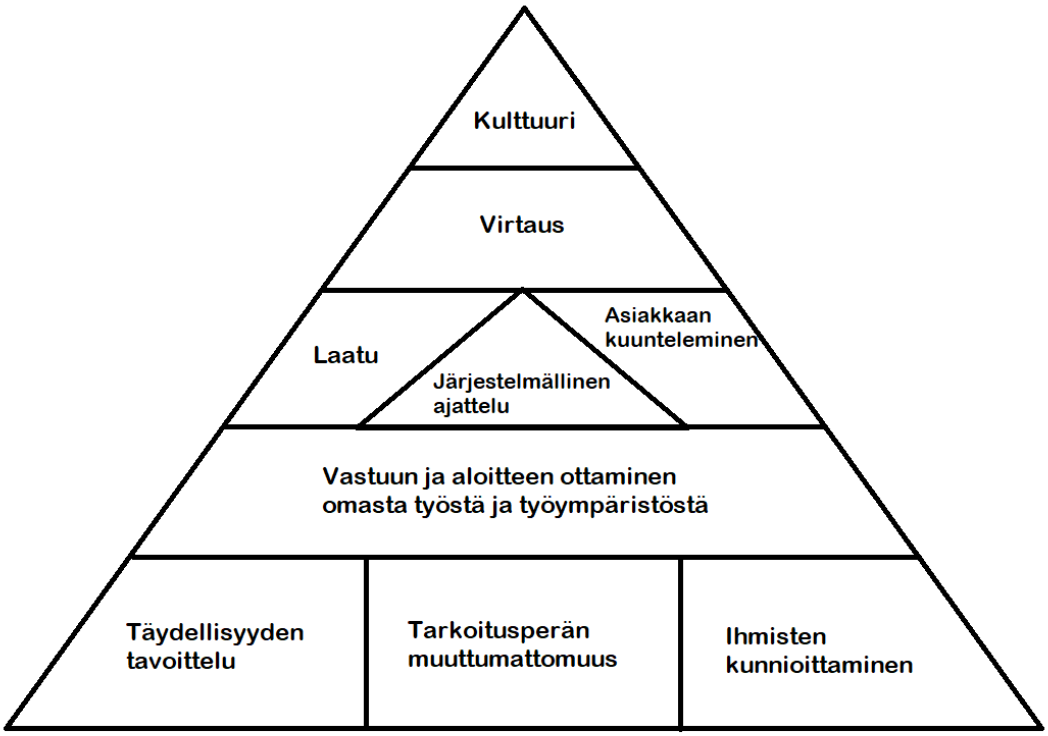
Leania voi pitää toimintastrategiana. Se on tapa hoitaa, muuttaa ja suunnitella yritystä, maksimoimalla asiakkaan arvo ja vähentämällä jätettä; jätteellä tarkoitetaan työtä, joka ei tuota arvoa tuotteelle tai palvelulla. Lean sai alkunsa 1900-luvun alkupuolella Henry Fordin liikkeellä olleesta kokoonpanolinjasta. Myöhemmin Lean sai myös vaikutteita Toyotan tuotantojärjestelmästä sekä Edwards Demingin laatujohtamisen mallista. Lean kehittyi Lean IT:ksi 90-luvulla, kun sitä alettiin käyttämään muillakin aloilla. Lean jatkaa kehittymistään ja sitä sovelletaan uusiin konteksteihin. IT-ala kehittyy todella nopeasti, jonka takia Lean IT:lle voi vain olla alustava määritelmä. On joukko hypoteeseja korostamaan Lean IT:n kehittyvää luonnetta. Nykyisten hypoteesien merkitystä on venytetty niin paljon, että on nähtävissä paljon erilaisia tulkintoja eri tasoilla (työkalut ja tekniikat, periaatteet, menetelmät, organisaation perusarvot). (Lean IT Association 2015.)

Lean IT tarjoaa perusteelliset konseptit nykyajan liiketoiminnalle, operaatioiden ja muutoksien aloittamisesta suunnitteluun. Nämä konseptit perustuvat kestävän menestymisen jatkuvaan parantamiseen ja kulttuurillisiin mahdollisuuksiin, koko yrityksen kattavan prosessikapasiteetin rakentamiseen ja yhdenmukaistamiseen sekä tulosten luomiseen pitkällä aikavälillä. Lean yritysten sanotaan olevan menestyneempiä säilyttämään kykyjä, kestävässä menestyksessä ja yrityksen arvossa, kuin yritykset, jotka eivät käytä Leania. (Lean IT Association 2015.)

Lean IT keskittyy operatiivisen osaamisen kehittämiseen parantamalla prosessitehokkuutta, ketteryyttä ja palvelun laatua. Tämän saavuttaakseen tulee rakentaa arvo- ja asiakaskeskeinen kulttuuri, jossa työntekijät osallistuvat Lean IT prosesseihin. Toimintastrategiaan kuuluu myös kriittisimpien palvelujen ja sovelluksien kehittäminen, optimoimalla IT-prosessit ja toiminnot. Lean IT vaikuttaa suuresti organisaation kulttuuriin. Se valtuuttaa työntekijät osallistumaan prosessien optimointiin. Tämän tavoitteena on saavuttaa tiukka ongelmanratkaisuprosessi, saavuttaakseen suuremman taloudellisen ja strategisen arvon. (Heunks 2014.)

Lean IT:ssä on monia eri näkökantoja kahdessa perspektiivissä. Perspektiivit ovat sisäänpäin ja ulospäin suuntautuva Lean IT. Sisäänpäin suuntautuva Lean IT soveltaa jatkuvan parantamisen työkaluja ja periaatteita projekteihin, palveluihin, IT-toimintaan ja ohjelmistokehitykseen. Tarkoituksena auttaa IT-organisaatiota saavuttamaan toiminnallinen huippuosaaminen. Toinen perspektiivi eli ulospäin suuntautuva Lean IT, joka parantaa ja innovoi jatkuvasti hallintajärjestelmiä ja liiketoimintaprosesseja tekemällä yhteistyötä tieto, tietojärjestelmien ja IT-organisaatioiden kanssa. Nämä perspektiivit eivät ole erillään toisistaan vaan ne täydentävät toisiaan. Niiden tarkoitus on palvella Lean muutosten päätavoitetta eli arvon luomista asiakkaille ja yritykselle. (Heunks 2014.)

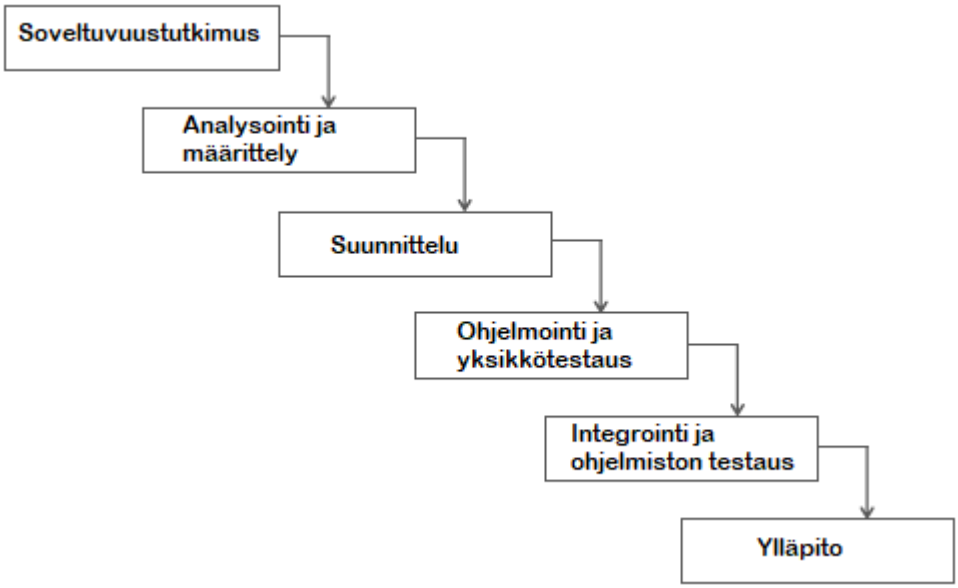
Lean IT perustuu yrityskohtaisiin Lean-periaatteisiin, joista koostuu Lean IT pyramidi (Kuvio 3). Pyramidin alimmalla eli pohjatasolla ovat: täydellisyyden tavoittelu, tarkoitusperän muuttumattomuus ja ihmisten kunnioitus. Seuraava taso on varattu kokonaan ennakoituvalle käyttäytymiselle, se tarkoittaa vastuun ja aloitteen ottamista omasta työstä ja työympäristöstä. Kolmas taso on tietoisuus. Siihen vaikuttavat lähteen laatu, järjestelmällinen ajattelu sekä asiakkaan kuunteleminen. Neljättä tasoa hallitsee virtaus. Virtauksella tarkoitetaan palveluiden, tiedon ja materiaalien jatkuvaa eteenpäin kulkemista. Viides ja viimeinen kerros on Lean IT:n periaatteiden kulmakivi eli kulttuuri. Kulttuuri pitää sisällään organisaation arvot ja vaatimukset. Näiden periaatteiden tarkoituksena on säästää aikaa ja kuluja, nostaa moraalia sekä parantaa laatua. (Heunks 2014.)



Kuvio 3: Lean IT pyramidi

3.4 Vesiputous

Vesiputousmalli oli ensimmäinen ohjelmistotuotannossa yleisesti käytössä ollut prosessimalli. Se jäljet johtava 1970 julkaistuun dokumenttiin, jonka kirjoitti Winston Royce. Vesiputousmalli oli ennen erittäin suosittu ohjelmistosuunnittelussa, nykyään sen käyttö on vähentymässä. Vesiputousmalli kuvaa ohjelmistokehitystä lineaarisesti ja askelittain. Mallin jokaisella kehitysvaiheella on selkeä tavoite. Vaiheet toteutetaan yksi kerrallaan ja seuraava vaihe alkaa vasta kun aikaisemman vaiheen tavoitteet ovat saavutettu (Kuvio 4). (UKEssays 2018.)



Kuvio 4: Vesiputousmallin rakenne

Vesiputousmallin ensimmäinen vaihe on soveltuvuustutkimus. Tämän tutkimuksen tavoitteena on selvittää, onko taloudellisesti ja teknisesti kannattavaa kehittää kyseistä sovellusta. Soveltuvuustutkimuksessa kerätään kaikki tuotteeseen liittyvä tieto ja ongelmat. Tämän tiedon pohjalta analysoidaan, jotta saadaan selville: asiakkaalle tärkeät vaatimukset, kaikki ongelmanratkaisutavat ja ratkaisustrategiat. Ratkaisustrategioiden arviointi yleensä vaatii arvioin jokaisen strategian kehittämiskustannuksista, kehitysajasta ja tarvittavista resursseista. Kun paras mahdollinen ratkaisu on identifioitu, loput kehitysvaiheet suoritetaan tämän pohjalta. Soveltuvuustutkimuksessa tehdään siis päätökset ja hyväksytetään tarkka ratkaisustrategia. Koko ohjelmistokehitys voi jo pysähtyä soveltuvuustutkimukseen, jos siinä ei löydy sopivaa ratkaisustrategiaa resurssirajojen, liian korkeiden kustannusten tai teknisten syiden vuoksi. (UKEssays 2018.)

Soveltuvuustutkimuksen jälkeinen vaihe on vaatimusten analysointi ja määrittely. Tässä vaiheessa on tarkoitus selvittää asiakkaan vaatimukset ja dokumentoida ne tarkasti. Tässä vaiheessa on kaksi eri toimintaa: vaatimusten kerääminen ja analysointi sekä vaatimusmäärittely. Vaatimusten keräämis- ja analysointivaiheessa asiakkaalta kerätään kaikki sovellusta koskevat vaatimukset. Nämä vaatimukset sitten analysoidaan ja poistetaan epäjohdonmukaiset ja epätäydelliset vaatimukset. Epäjohdonmukainen vaatimus on vaatimus, jossa osa siitä on ristiriidassa jonkun toisen osan kanssa. Epätäydellinen vaatimus on taas vaatimus, jossa osa todellista vaatimusta on jätetty pois. Keräämis- ja analysointivaiheen jälkeen suoritetaan vaatimusmäärittely. Vaatimusmäärittelyssä kirjataan keräämis- ja analysointivaiheessa tunnistetut vaatimukset ohjelmiston vaatimusmäärittely -asiakirjaan. Tämä kyseinen dokumentti toimii sopimuksena asiakkaiden ja kehittäjien välillä. Tärkeimmät osat tätä asiakirjaa ovat toteutuksen tavoitteet, toiminalliset vaatimukset ja ei-toiminnalliset vaatimukset. Jos asiakkaan ja kehittäjien välille tulee riitaa, voidaan se ratkaista tutkimalla ohjelmiston vaatimusmäärittelyä. (GeekTonight 2019.)

Kolmas vaihe on suunnittelu. Tässä vaiheessa päätetään miten sovellus tullaan kehittämään. Suunnitteluvaiheessa on kaksi eri osa-aluetta: looginen suunnittelu ja fyysinen suunnittelu. Looginen suunnittelu on tapahtunut jo analysoinnin aikana. Fyysisessä suunnittelussa toteutetaan yksityiskohtainen kuvaus siitä mitä tarvitaan alkuperäisen ongelman ratkaisemiseksi; syöte, tuloste, lomakkeet, koodausmenetelmät, prosessointimäärittely, tietokannat. Tässä vaiheessa päätetään myös ohjelmointikielestä, laitteisto- ja ohjelmistoalustasta, koulutuksesta, dokumentaatiosta, käyttöliittymästä, ohjausprosessista, tietorakenteesta, henkilöstövaatimuksesta, järjestelmän rajoituksista ja kuormituksesta, varmuuskopioiden ottamisesta ja tarvikkeiden hankkimisesta. (Innorobix Automation.)

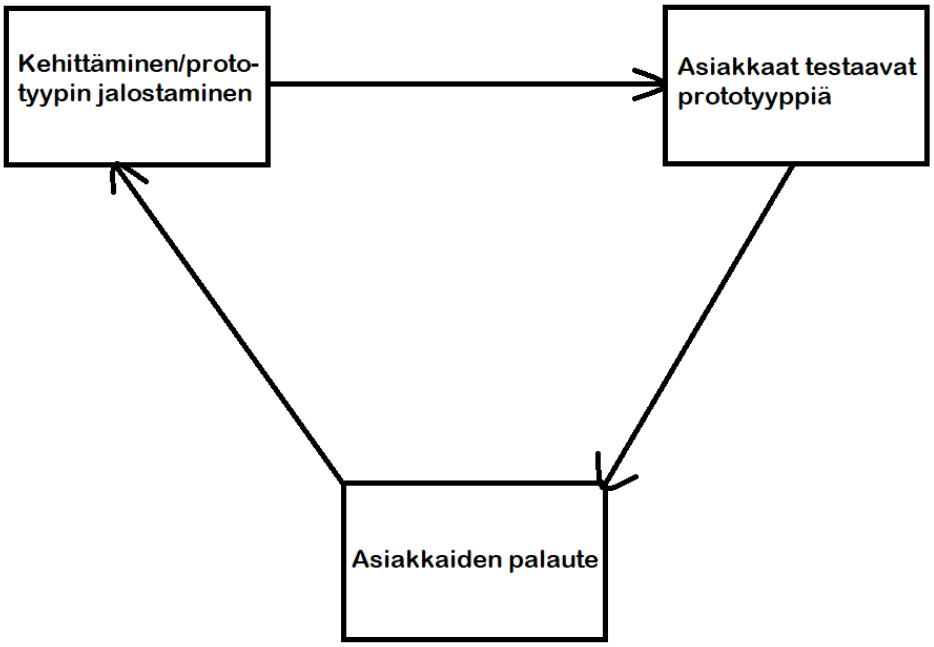
Vesiputousmallin seuraava vaihe on ohjelmointi ja yksikkötestaus. Ohjelmointivaiheessa toteutetaan ohjelmistoarkkitehtuuri, joka suunniteltiin suunnitteluvaiheessa. Tämä vaihe pitää sisällään: ohjelmoinnin, vianmäärityksen ja moduulien testauksen. Ohjelmiston koodaus toteutetaan suunnitteluvaiheessa valitulla ohjelmointikielellä. Ohjelmiston komponentit kehitettään erillään toisistaan. Kun komponentti on valmis, se testaan yksikkötestauksessa, jos komponentti on valmis, voidaan se integroida vähitellen lopputuotteeseen. Ohjelmointivaiheen tavoite on tuottaa kokonainen ohjelmistotuote, joka sitten testataan kokonaisuutena seuraavassa vaiheessa. (IONOS 2019.)

Viides vaihe on integrointi ja ohjelmiston testaus. Integrointivaiheessa; joka aloitetaan pian yksikkötestauksen jälkeen, lisätään ohjelmiston moduulit yksitellen osittain integroituun järjestelmään. Integrointi tapahtuu useassa eri vaiheessa ja järjestelmä testataan integrointivaiheiden välissä. Ohjelmiston testaus koostuu kolmesta vaiheesta: alfa-, beta-, hyväksymistestaus. Alfatestauksessa kehitystiimi itse suorittaa järjestelmän testauksen. Betatestauksessa testauksen suorittavat oikeat käyttäjät, jotka vapaaehtoisesti osallistuvat testaukseen. Lopuksi on vielä hyväksymistestaus, jossa ohjelmisto on jo toimitettu asiakkaalle, joka itse suorittaa testauksen ja päättää hyväksyykö hän ohjelmiston sellaisenaan. (GeeksforGeeks 2019.)

Vesiputousmallin kuudes ja viimeinen vaihe on ylläpito. Ylläpitovaihe on kaikista työläin vaihe, siihen kuluu 60 % kaikesta ohjelman kehittämiseen käytetystä työstä. Ylläpidon voi jakaa kolmeen osa-alueeseen: täydellinen, korjaava ja mukautuva ylläpito. Täydellisessä ylläpidossa asiakkaat antavat rakentavaa palautetta ohjelmasta, jonka seurauksena ohjelmaan lisätään tarvittavia toimintoja tai poistetaan tarpeettomia toimintoja. Korjaava ylläpito on virheiden ja vikojen löytämistä ja korjaamista. Yleensä nämä viat; jotka voivat vaikuttaa designiin, logiikkaan tai koodiin, tulevat ilmi bugiraporteista, jotka ovat asiakkaiden tai käyttäjien tekemiä. Mukautuvan ylläpidon tehtävä on siirtää ohjelma uuteen toimintaympäristöön. Toimintaympäristöön vaikuttavat: laitteistot, käyttöjärjestelmät, pilvipalvelut tai ohjelmariippuvaisuudet. (GeeksforGeeks 2019.)

3.5 Prototyyppimenetelmä

Prototyyppimenetelmällä tarkoitetaan ohjelmistosovelluksen prototyypin rakentamista. Tarkoituksena on siis, rakentaa ohjelmisto, joka demonstroi lopullisen ohjelmiston toiminnallisuuksia. Prototyyppimenetelmässä siis rakennetaan prototyyppi, testataan se ja tehdään tarvittavat korjaukset. Tätä toistetaan niin kauan, kunnes ollaan tyytyväisiä tuotteeseen (Kuvio 5). Prototyyppimenetelmän suosio ohjelmistokehitysmallina kasvaa jatkuvasti. Tämän mallin ansiosta kehittäjät pystyvät ymmärtämään paremmin asiakkaan tarpeet ja vaatimukset jo ohjelmiston kehityksen aikaisessa vaiheessa. Asiakkaat voivat tarjota arvokasta palautetta, jonka avulla ohjelmistokehittäjät ja -suunnittelijat saavat paremman käsityksen mitä heidän tuotteeltaan odotetaan. (Tutorials Point.)



Kuvio 5: Prototyyppimenetelmän perusrakenne

Prototyyppimenetelmästä on neljä erilaissa prototyyppiä: kertakäyttöinen, evoluutio, inkrementaalinen ja extriimi. Näistä selvästi suosituimmat ovat kertakäyttöinen- ja evoluutioprototyyppi. Kertakäyttöinen prototyyppi perustuu alustaviin vaatimuksiin. Sen kehitysaika on lyhyt ja sen tavoite on täyttää visuaaliset vaatimukset. Asiakkaalta saatu palaute prototyypistä auttaa muokkaamaan vaatimuksia, joiden perusteella luodaan uusi prototyyppi. Tämä toistuu niin kauan, kunnes vaatimukset on määritetty. Tässä vaiheessa aikaisemmin kehitetyt prototyypit hävitetään, eivätkä ne ole osa lopullista prototyyppiä. Tämän tekniikan ansiosta kehittäjät ja suunnittelijat saavat välitöntä palautetta asiakkaan vaatimuksista. (Guru99.)

Evoluutioprototyypissä prototyyppiä jalostetaan pikkuhiljaa asiakkaalta saadun palautteen perusteella. Tässä prototyypissä luodaan ensiksi toiminnallinen prototyyppi, jolla on vähän toiminnallisuuksia. Aluksi kehitetty prototyyppi toimii ytimenä koko projektin ajan, ja sen ympärille luodaan lisää toimintoja. Evoluutioprototyypissä ominaista alusta saakka hyvin selkeät vaatimukset, joiden ympärille prototyyppiä aletaan laajentamaan. Tämä menetelmä auttaa säästämään aikaa, rahaa ja vaivaa. Koska prototyypin ydin pysyy alusta saakka samana, ei tarvitse jokaisessa projektin vaiheessa aloittaa prototyypin kehittämistä alusta saakka. Tämä menetelmä toimii myös hyvin projekteissa, joissa käytetään uutta tuntematon tekniikkaa tai projekti on hyvin monimutkainen. (Guru99.)

Sitten on vielä kaksi vähemmän tunnettua prototyyppimallia extriimi ja inkrementaalinen. Extriimiprototyyppiä käytetään lähinnä web-pohjaisien sovellusten kehittämiseen. Sen voi jakaa kolmeen selkeään tasoon matala-, keski- ja korkeataso. Matalassa tasossa luodaan staattinen prototyyppi, jossa kaikki sivut ovat HTML-muodossa. Seuraavassa eli keskitasossa lisätään tähän prototyyppiin toiminnallisuuksia palvelutasolla. Viimeisellä eli korkealla tasolla prototyyppi otetaan käyttöön palvelutasolla ja toteutetaan toiminnallisuudet sen avulla. Inkrementaalisessa prototyypissä taas projekti jaetaan pieniin osiin. Jokaisessa osassa luodaan oma prototyyppi, joka kehitetään eristyksissä muista prototyypeistä, kunnes se on valmis. Kun asiakas on täysin tyytyväinen tuotteeseen, yhdistetään kaikki prototyypit yhdeksi toimivaksi sovellukseksi. Tämä malli on hyvä, jos haluaa luoda sovelluksen nopeasti ja testata sen helposti ja perusteellisesti. (Guru99; Course Hero 2016.)

4 Tutkimusmenetelmät

Työn tutkimuksessa käytettiin sekä laadullisia, että määrällisiä menetelmiä, jonka takia tutkimusstrategiana oli triangulaatio. Tutkimuksessa aineistoa oli laadullisessa sekä määrällisessä muodossa, joten kyseessä oli aineistotriangulaatio. Kyseessä oli prosessi, jonka kestoa ja kehitystä voitiin mitata numeroin. Näiden numeroiden avulla voitiin sitten luoda tilastoja, joista näki kehityksen. Tutkimuksessa keskityttiin tutkimuskohteen seuraamiseen, muutosten havaitsemiseen ja muutokseen vaikuttaneiden tekijöiden analysoimiseen. Kun työprosessin parannukset oli implementoitu, käytettiin myös vertailevaa tutkimusta. Tässä verrattiin lähtötilannetta ja nykyistä tilannetta keskenään ja pyrittiin löytämään eroja työnkulussa sekä käyttäjäystävällisyydessä. (Saaranen-Kauppinen & Puusniekka 2006; Jyväskylän yliopisto 2015c; Jyväskylän yliopisto 2015e.)

Työn aineistonkeruumenetelmänä toimivat havainnointi ja haastattelut. Havainnoinnissa tutkija seurasi työn kulkua ja piti kirjaa siitä, miten sujuvasti kukin työvaihe eteni. Tutkija oli itse fyysisesti paikalla seuraamassa, kun työ suoritettiin. Havainnoinnin tarkoituksena oli nähdä, miten työprosessi tapahtuu työympäristössä. Näin tutkija pystyi itse havaitsemaan mahdollisia ongelmakohtia, joita työntekijä ei itse huomaa tai muista mainita. Haastatteluissa tutkija haastatteli työn tekijää sekä taustatoimien ylläpitäjiä työprosessiin liittyen. Haastattelijan tavoitteena oli ymmärtää työprosessin pullonkaulat sekä hidastetöyssyt. Selvittämällä työprosessin heikot osa-alueet, pystyy niitä parantamaan ja nopeuttamaan työprosessia. Haastatteluissa haastattelija kysyi suoraan mitkä ovat työprosessin hitaat vaiheet ja miten niitä voisi parantaa. Tämän jälkeen hän kertoi omista havainnoistaan ja haki niihin varmistusta. (Jyväskylän yliopisto 2015a; Jyväskylän yliopisto 2020.)

Aineiston analyysimenetelmänä käytettiin määrällistä analyysia. Määrällinen analyysi sopi hyvin tällaiseen tilanteeseen missä vertaillaan lukuja keskenään. Kerätty data käsiteltiin tilastollisesti kuvaavalla analyysillä, data järjesteltiin taulukoihin ja käytettiin laskutoimituksia apuna tuomaan esille eri näkökulmia. Tutkija käytti myös apunaan kaavioita kuvaamaan prosessissa tapahtunutta muutosta. (Jyväskylän yliopisto 2015b; Jyväskylän yliopisto 2015d.)

4.1 Määrällinen tutkimus

Määrällinen tutkimus on empiirinen tutkimus ja se käsittelee numeerista dataa tai dataa, jonka voi muuttaa numeroiksi. Määrällisen tutkimuksen vastakohta on laadullinen tutkimus, joka käsittelee numeerisen datan sijasta ilmiöön liittyviä ajatuksia, vaikuttimia ja tunteita. Perusmenetelmiä, joita käytetään numeeristen tietojen tutkimiseen, kutsutaan tilastoiksi. Erilaisia tekniikoita käytetään tilastoiden esittämiseksi, järjestämiseksi, tulkitsemiseksi ja analysoimiseksi. Tilastotiede on suuri tutkimusalue ja sitä sovelletaan laajasti monella eri tieteenaloilla. Henkilökohtaisten tietokoneiden ja myös muiden erilaisten tietokoneiden yleistymisen seurauksena prosessit, joilla suoritetaan tietojen käsittelyä ja analysointia ovat tulleet helpommin saataville. Tämän seurauksena vaara siitä, että datalle tehdään analyysi, ymmärtämättä sen keruuseen liittyviä testejä ja niiden soveltamista. (Sheard 2018.)

Tutkimusstrategiaksi valittiin triangulaatio, koska tutkimuksen aiheena oli yksinkertainen prosessi, jonka produktiivisuutta ja kehitystä mitattiin laadullisin keinoin, mutta sitä käsiteltiin ja analysoitiin määrällisin keinoin. Määrällisillä keinoilla saatiin hyödyllistä ja helposti käsiteltävää ja ymmärrettävää dataa prosessin ongelmakohdista ja kehityksestä. Tämä data voitiin myös järjestellä monella eri tavalla, jonka ansioista voitiin ymmärtää esimerkiksi prosessin tasaisuutta. Prosessista ei ollut hankittu tämäntapaista dataa aikaisemmin, joten se tulee toimimaan hyvänä pohjana tulevaisuutta ajatellen.

4.2 Havainnointi

Havainnoinnissa eli observoinnissa kerätään tietoa tarkkailemalla henkilön toimintaa. Havainnointia voidaan käyttää itsenäisesti tai tukemaan haastattelua. Havainnoinnin vahvuutena on, että se tarjoaa välitöntä ja suoraa informaatiota tutkittavien toiminnasta. Se mahdollistaa tutkimisen luonnollisessa toimintaympäristössä. Havainnointi sopii hyvin muuttuvien tilanteiden tutkimiseen. Siitä saatu tieto voi liittyä käyttäytymiseen, tapahtumiin tai fyysisiin kohteisiin. Havainnointi voidaan jakaa osallistuvaan tai ei-osallistuvaan havainnointiin. (Saaranen-Kauppinen & Puusniekka 2006.)

Osallistuva havainnointi jakautuu kahteen eri muotoon passiivinen ja aktiivinen havainnointi. Passiivisessa havainnoinnissa tutkija osallistuminen ei vaikuta tilanteen kulkuun. Aktiivisessa havainnoinnissa tutkijan läsnäolo vaikuttaa tutkittavaan. Osallistuvassa havainnoinnissa tutkijan on kuitenkin aina pystyttävä oma roolinsa ja sen vaikutus tutkimukseen. Tutkija voi olla osallisena joko käyttäytymisen seuraajana tai osallistujana. Tutkijan läsnäolo on aina kaikkien tiedossa, joten hän ei voi olla tilanteesta täysin ulkopuolinen. Tavoitteena on tehdä tutkijan läsnäolosta mahdollisimman luontevaa ja huomaamatonta. Tämän takia tutkija ei saa puuttua tutkittavien omiin käytäntöihin ja tapoihin. Osallistuvassa havainnoinnissa kommunikaatio ei ole pelkästään puhetta vaan se sisältää myös liikkeitä, eleitä, kosketuksia ja ilmeitä. Nämä tulisi aina ottaa huomioon, mutta niiden tulkitsemisessa pitää huomioida, että ne tulkitaan oikein eikä niiden painoarvoa liioitella. (Saaranen-Kauppinen & Puusniekka 2006.)

Ei-osallistuvassa eli suorassa havainnoinnissa on tarkoitus tutkia tilannetta, tutkittavien tietämättä sitä. Tätä havainnointitapaa voidaan toteuttaa, joko kentällä tai laboratorio-olosuhteissa. Suorassa havainnoinnissa tutkijan on käytettävä kaikkia aistejaan sekä mahdollisia apuvälineitä havainnoimaan tilannetta. Tässä havainnoinnissa tutkijan läsnäolo voi olla tutkittavien tiedossa, jolloin on kyse avoimesta suorasta havainnoinnista. Toinen vaihtoehto on kätkeä tutkija tutkittavilta, tätä kutsutaan piilohavainnoinniksi. Piilohavainnoinnissa tutkija ei saa vaikuttaa tapahtumien kulkuun vaan hänen on tarkoitus olla täysin ulkopuolinen. Tämän lisäksi tutkijan on oltava mahdollisimman objektiivinen tutkimuskohdetta kohtaan. (Saaranen-Kauppinen & Puusniekka 2006.)

Havainnointitekniikka voi olla, joko systemaattista tai ei-systemaattista. Systemaattisessa havainnoinnissa tutkimus on erittäin yksityiskohtaista, strukturoitua, jäsenneltyä ja joustamatonta. Systemaattinen tekniikka vaatii, että tutkija jäsentää ongelman ja luo sitä varten luokittelun ennen havainnoinnin aloittamista. Tämä vaatii, että tutkittavasta aiheesta on jo ennestään tietoa, jonka avulla voidaan päättää milloin ja mitä havainnoidaan. Ei-systemaattinen tekniikka on taas joustavaa, strukturoimatonta ja väljää. Tämän tekniikan avulla pyritään saamaan paljon ja monipuolista tietoa asiasta. Etukäteistiedon puutteen seurauksena havainnointia ei voida etukäteen ja joudutaan hyödyntää tutkittavan ilmiön teoriaa. Havainnoinnissa tulee aina määrittää tavoitteet sekä vaadittava tarkkuus. (Saaranen-Kauppinen & Puusniekka 2006.)

Jotta havainnoinnin tuloksia voi tulkita oikein on tutkijalla oltava asiasta paljon taustatietoa. Havainnoinnissa voi esiintyä tuloksia, jotka eivät suoraan kerro mitä ne tarkoittavat. Tutkijan tulee ymmärtää miten mikäkin havaittu asia vaikuttaa ilmiöön. Havainnointia tutkimusmenetelmänä onkin kritisoitu siitä, miten tutkija voi vaikuttaa tilanteeseen. Tätä ongelmaa voidaan vähentää niin, että tutkija läsnäolosta tehdään mahdollisimman normaalia ja luonnollista. Tämä voi myös hankaloittaa tutkimusta, jos tutkijan tutustuu tutkittaviin liian hyvin ja heidän välilleen syntyy tunteita. Tämän seurauksena tutkimuksen objektiivisuus huononee. (Saaranen-Kauppinen & Puusniekka 2006.)

Tässä tutkimuksessa käytettiin osallistuvaa havainnointia. Tutkijan läsnäolo oli kaikille tutkimuksessa oleville selvää. Tämä menetelmä valittiin, koska se oli kaikista helpoin ja loogisin tähän tilanteeseen. Tutkija oli koko ajan työntekijän lähettyvillä, koska hän seurasi työntekijän tekemistä kentällä. Ei ollut mitään syytä, miksi tutkijan läsnäoloa olisi tarvinnut peitellä. Koska tutkija seurasi työntekijän tekemistä eikä itse osallistunut siihen, oli osallistuva havainnointi passiivista. Työntekijä teki tarkasti määritettyä työprosessia, joten havainnointi oli myös systemaattista. Tutkijan läsnäolo tilanteessa oli jo valmiiksi hyvin normaalia ja luonnollista, koska hän oli myös itse tehnyt samoja työtehtäviä ja juuri kyseinen työntekijä oli kouluttanut tutkijan näihin työtehtäviin. He olivat aikaisemminkin olleet samanlaisessa tilanteessa, joten ei tarvinnut tehdä mitään sen eteen, että havainnoinnista olisi tullut normaalimpaa ja luonnollisempaa. Koska tutkija tunsi työntekijän, oli vaarana, ettei hän pystyisi olemaan tilanteessa täysin objektiivinen. Objektiivisuus ei ollut ongelma, koska tutkija vain kellotti prosessivaiheen kestoa, eikä hän arvioinut työntekijän työskentelyä mitenkään.

4.3 Haastattelut

Haastattelu on yksi suosituimmista menetelmistä hankkia tietoa. Haastattelun tarkoituksena on tutkimustehtävän suorittaminen, joten haastattelumenetelmän valinta perustuu siihen, minkälaista tietoa haastattelija tavoittelee. Haastattelusta on tarkoitus saada tutkimusaineistoa, jota sitten analysoidaan ja tutkitaan tutkimustehtävän täyttämiseksi. Haastatteluissa pyritään saamaan tietoa ihmisistä heiltä itseltään. Haastatteluissa yleinen ongelma on, että niihin suhtaudutaan liian realistisesti. Ihmiset eivät aina kerro asioista niin kuin ne oikeasti ovat. Haastattelu ei siksi ole aina toimiva aineistonkeruumenetelmä. Haastattelun sopivuutta aineistonkeruuseen tulisi aina selvittää. (Saaranen-Kauppinen & Puusniekka 2006.)

Haastattelumenetelmät voi jakaa strukturoituun, puolistrukturoituun tai strukturoimattomaan. Strukturoidussa- eli lomakehaastattelussa on valmiiksi muotoiltu kysymykset ja tietyt vastausvaihtoehdot. Se on formaalisin haastattelumuoto ja vastaa ohjatusti kyselylomakkeen täyttämistä. Kaikille haastateltaville esitetään samat kysymykset, samassa järjestyksessä. Vastauksiin on valmiit vaihtoehdot, joista haastateltava valitsee sopivimman. Tätä menetelmää käytetään silloin, kun halutaan saada paljon aineistoa ja käsitellä sitä tilastollisen analyysin tavoin. Puolistrukturoidussa haastattelussa pysyy aina sama teema. Haastatteluiden välillä kysymykset ja niiden järjestys on likipitäen sama. Tämä menetelmä on tarkoitettu käytettäväksi silloin, kun halutaan tietoa tietyistä aisoista, eikä haastateltavan ole tarpeellista saada ylimääräistä vapautta haastattelutilanteessa. (Saaranen-Kauppinen & Puusniekka 2006.)

Strukturoimaton eli avoin haastattelu on kaikista vapain haastattelumenetelmä. Avoin haastattelu on saanut alkunsa pappien ja lääkäreiden käyttämistä menetelmistä. Avoin haastattelu pyrkii olemaan mahdollisimman avoin ja luonteva. Sitä ei ole käsikirjoitettu vaan se pyörii vapaasti tietyn aiheen ympärillä. Haastattelutilanne kulkee enimmäkseen haastateltavan ehdoilla. Vaikka haastatteluun ei ole valmistettu valmiita kysymyksiä, on tavoitteena kuitenkin saada tietoa tutkijan määrittämistä aiheesta ja teemoista. Haastattelun strukturoimattomuudella pyritään saamaan tietoa haastateltavan kokemuksista, mielipiteistä, muistoista, tuntemuksista tai perusteluista. Haastattelijan on tarkoitus pitää keskustelu hänen haluamassaan aiheessa, sekä tarvittaessa syventyä johonkin haastattelijan vastaukseen. Haastattelijalta vaaditaan herkkyyttä kuunteluun ja kykyä edistää keskustelua oikeaan suuntaan. (Saaranen-Kauppinen & Puusniekka 2006.)

Muita yleisiä haastattelumuotoja ovat teemahaastattelu ja ryhmähaastattelu. Teemahaastattelu on muodoltaan jotain puolistrukturoidun ja strukturoimattoman haastattelun välistä. Tässä haastattelussa keskustelu on hyvin vapaata ja se pyörii määriteltyjen teemojen ympärillä. Haastattelijalla on kuitenkin lyhyet muistiinpanot käsiteltävistä asioista tai kysymyksistä. (Saaranen-Kauppinen & Puusniekka 2006.)

Ryhmähaastattelua voidaan käyttää yksilöhaastatteluiden ohessa tai niiden sijasta. Ryhmähaastatteluista yleisimpiä ovat täsmäryhmähaastattelut ja parihaastattelut. Näissä molemmissa haastateltavat on valittu tarkasti. Ryhmähaastatteluissa haastattelumuoto on melko avointa, sillä erittäin strukturoidusta ryhmähaastattelusta ei olisi hyötyä lomakehaastatteluun verrattuna. Haastattelija kysyy ryhmältä aiheeseen liittyviä kysymyksiä ja antaa ryhmän sitten keskustella. Haastattelija voi myös esittää kysymyksiä ryhmän yksilöille. Ryhmähaastattelussa on yleensä tarkoitus tutkia ryhmän arvoja ja normeja, vuorovaikutusta tai yhteisiä näkemyksiä. Ryhmän koko valitaan sen mukaan, millaista tietoa tavoitellaan. Liian isojen ryhmien ongelmana on keskustelun kulku sekä sen tallentaminen. Haastatteluun voi myös osallistua useampi haastattelija, sen avulla voidaan luoda rennompi ilmapiiri tai saamaan monipuolisempia kysymyksiä. Ryhmähaastattelussa pyritään keräämään tietoa nopeasti useammalta haastateltavalta samaan aikaan. Sen hyvänä puolena on, että ryhmän jäsenet pystyvät auttamaan toisiaan muistamaan asioita, joita he eivät yksin muistaisi. Ryhmähaastattelun on todettu toimivan etenkin lapsia tai vanhuksia haastateltaessa. (Saaranen-Kauppinen & Puusniekka 2006.)

Tässä tutkimuksessa käytettiin avointa haastattelua. Haastatellessa työntekijää, tutkijan tarkoituksena oli saada mahdollisimman paljon tietoa työvaiheiden ongelmista sekä sujuvuudesta työntekijän näkökulmasta. Tavoitteena oli saada selville työprosessin puutteet ja ongelmakohdat. Keskustelua edistettiin kysymällä aiheesta tarkemmin ja välillä myös tuomalla tutkijan oma mielipide esille. Haastateltavalta kysyttiin hänen mielipidettään asioista sekä miksi hän ajattelee tai tekee juuri tietyllä tavalla. Nämä ideat vietiin eteenpäin ja selvitettiin, onko niiden toteuttaminen mahdollista ja onko niitä hyötyä.

4.4 Validiteetti ja reliabiliteetti

Tutkimuksen validiteetista puhuttaessa käydään läpi sitä, kuinka pätevä tutkimus on. Validiteetti pitää sisällään asioita kuten: onko saadut tulokset tai käytetyt menetelmät oikeita sekä kuinka perusteellisesti tutkimus on tehty. Tutkimuksen validiteettia laskevat tutkimuksessa tapahtuneet virheet. Näitä virheitä voivat olla: tutkija kysyy vääriä kysymyksiä, tutkija näkee vääriä periaatteita tai suhteita tai ei näe niitä ollenkaan. (Saaranen-Kauppinen & Puusniekka 2006.)

Reliabiliteetin voin jakaa kolmeen kohtaan: ajallinen-, metodin reliaabelius ja johdonmukaisuus tuloksissa. Ajallinen reliaabeliuus tarkoittaa tulosten muuttumattomuutta eri aikoina. Tämän selvittäminen voi olla ongelmallista sillä tutkimuksissa usein on muuttuvia objekteja, jotka vaikuttavat tulokseen. Metodin reliaabeliuksella tarkoitetaan missä olosuhteissa käytetyt metodit ovat johdonmukaisia ja luotettavia. Sen tarkoitus kertoa onko käytetyt metodit olleet oikeita ja onko niiden avulla saadut tulokset luotettavia. Johdonmukaisuus tuloksissa käsittelee eri välineillä saatujen tuloksien verrattavuutta. Se käsittelee: onko tulokset keskenään verrattavia, vaikka ne on hankittu eri välineitä käyttämällä. (Saaranen-Kauppinen & Puusniekka 2006.)

5 Tiedonkeruu ja analyysi

Tiedonkeruu toteutettiin viikoilla 4 ja 5. Tiedonkeruu tapahtui havainnoimalla sekä haastattelemalla työtä tekevää freelanceria. Havainnointi tapahtui asiakkaan tiloissa, jossa itse työprosessi suoritettiin. Työtä tehtiin sitä mukaa, kun autoja saapui kuvattavaksi, joten silloin, kun autoja ei ollut suoritettiin avointa haastattelua. Tutkija ei kirjannut haastatteluja kokonaisuudessaan ylös vaan ainoastaan yksittäisiä asioita, jotka hänen mielestään kannatti ottaa huomioon ja viedä eteenpäin.

5.1 Havainnointi, lähtökohta

Tutkija toteutti tämän työtehtävän sovelluskehittäjän pyynnöstä. Tutkija ja sovelluskehittäjä yhdessä pohtivat mikä olisi sopiva otanta. He päätyivät siihen, että viisikymmentä toistoa oli tarpeeksi suuri otanta. He myös sopivat, että paras tapa suorittaa havainnointi oli kellottaa työprosessin kesto nykyisessä muodossaan, sen jälkeen tehdä parannukset ja kellottaa parannettu työprosessi. Koska tutkija oli itsekin tehnyt kyseistä työprosessia aikaisemmin, sovittiin että hän ensiksi kellotti itseltään viisikymmentä toistoa ja tämän jälkeen kellotti freelancerin viisikymmentä toistoa. Tutkija itse päätti, että työprosessi kannattaa jakaa pienempiin osiin, jotta tuloksia olisi helpompi analysoida. Työprosessin voitiin jakaa kahteen pääosaan, kuvaus ja editointi. Nämä sitten vielä jaettiin pienempiin osiin. Kuvaus jakautui auton hakemiseen, auton valmistelemiseen, ulkokuviin, videoon, sisäkuviin, lopputoimiin ja auton palautukseen. Editointi puolestaan jakautui kuvien siirtämiseen kansioihin, kuvien editointiin, videon editointiin, odotukseen ja kansioiden purkaminen, kuvien lähettämiseen sekä dokumentointiin.

Tutkija suoritti oman työprosessinsa keston viikolla 3. Hän suoritti kellotuksen puhelimessaan olevalla sekuntikellollaan ja kirjasi kaikki viisikymmentä toistoa tekemäänsä Excel dokumenttiin. Dokumenttiin lisättiin myös erilaisia laskukaavoja kuvaamaan paremmin työprosessin kestoa eri näkökulmista. Seuraavaksi oli freelancerin työn kellottamisen vuoro. Aluksi kellotettiin autojen kuvaus. Tutkija painotti, ettei aikoja käytetä tarkastelemaan kuvaajan ahkeruutta vaan ainoastaan tutkimaan prosessin kehitystä. Tutkija myös muistutti, ettei ole mikään kiire vaan toistot pitäisi pystyä tapahtumaan mahdollisimman tasaisesti, toistettavasti ja realistisesti. Koska autoja oli vähemmän kuin viikolla 3 jouduttiin havainnointia suorittamaan kahdella eri viikolla. Tutkijan muistutuksista huolimatta freelancer välillä teki tarkoituksella työtä nopeampaan tahtiin, kuin hän normaalisti tekisi. Tämä saattoi johtua tutkijan läsnäolosta. Tutkija kuitenkin muistutti kuvaajaa, että tarkoitus ei ole saada mahdollisimman nopeata aikaa vaan realistista ja uskottavaa dataa. Tämän seurauksena freelancer alkoi tehdä taas töitä tasaisempaan tahtiin.

Autojen editointi tapahtui enimmäkseen freelancerin kotona, joten tutkija ei voinut olla läsnä editointihetkellä. Tutkija näytti esimerkkien avulla, miten kellotus tapahtuu, jotta freelancer pystyi itse suorittamaan kellottamisen. Freelancer lähetti kellottamansa ajat tutkijalle WhatsApp sovelluksen kautta. Aikojen ei tarvinnut olla vertailtavissa tutkijan omiin aikoihin, tämän takia tutkija ei uskonut kellottajalla olevan vaikutusta lopputuloksiin. Aikoja oli tarkoitus verrata freelancerin omiin aikoihin, jotka kellotettaisiin parannusten jälkeen. Jotta ajat olisivat keskenään vertailtavissa, oli tarkoitus, että freelancer kellottaisi silloinkin omat aikansa. Tutkija lisäsi nämä ajat samaan Excel dokumenttiin ja suoritti myös niille samat laskukaavat kuten aikaisemminkin.

5.2 Haastattelut

Tutkija päätti suorittaa haastatteluja havainnoinnin ohessa viikoilla 4 ja 5. Koska kellottamisen välillä oli hukka-aikaa, oli se järkevää hyödyntää suorittamalla haastatteluja. Haastattelujen teemana oli itse työprosessi, sen hyvät ja huonot puolet sekä parannusehdotukset. Haastattelu oli erittäin avointa eikä sitä varten ollut valmisteltu kysymyksiä ja sitä ohjattiin jatkokysymyksillä sekä haastattelijan omilla näkemyksillä. Haastattelulla ei ollut yhtä suuri painoarvo kuin havainnoinnilla. Haastattelujen tavoitteena oli löytää työprosessista puutteita tai parannusehdotuksia siihen. Tarkoituksena oli myös saada toinen näkökulma asioihin ja kuulla miten joku toinen henkilö näkee samat prosessivaiheet.

Haastatteluista ilmeni, että uusi työprosessiin lisätty seurantasovellus hidasti työnkulkua ja välillä sen käyttö unohtui. Tutkija sai myös vahvistuksen omille näkemyksilleen siitä, että autot oli järjestelty huonosti ja se välillä hidasti työntekoa myös autoissa olleet, ylimääräiset tavarat hidastivat paljon työntekoa ja aiheuttivat turhaa fyysistä rasitusta kuvaajalle. Editoinnista selvisi, että freelancer tykkäsi käyttää paljon enemmän hiirtä, jonka seurauksena käyttäjäkokemus oli erilainen verrattuna henkilöön, joka käyttää paljon pikanäppäimiä. Haastatteluista ilmeni myös muutamia hyviä pieniä parannusehdotuksia sovelluksen kehittämiseksi.

Haastattelut sujuivat mutkattomasti. Keskustelu oli sujuvaa ja aiheessa pysyttiin pääosin. Keskustelu oli rakentavaa ja haastattelija löysi tavoittelevansa toisen ja erilaisen näkökulman asioihin. Haastattelija ja haastateltava olivat usein asioista samaa mieltä mutta myös kyseenalaistivat niitä. Haastattelusta saatiin tavoiteltua aineistoa työprosessiin liittyen. Vaikka haastatteluilla oli vain pieni osuus, silti ne toivat esiin erilaista dataa, mitä havainnoimalla ei olisi saatu. Haastattelujen vaikeutena oli keksiä työprosessin kehitysehdotuksia ja löytää työprosessin heikkoja kohtia. Haastattelut olivat kaikin puolin onnistuneet ja ne sopivat erittäin hyvin havainnoinnin lisäksi tutkimaan työprosessia.

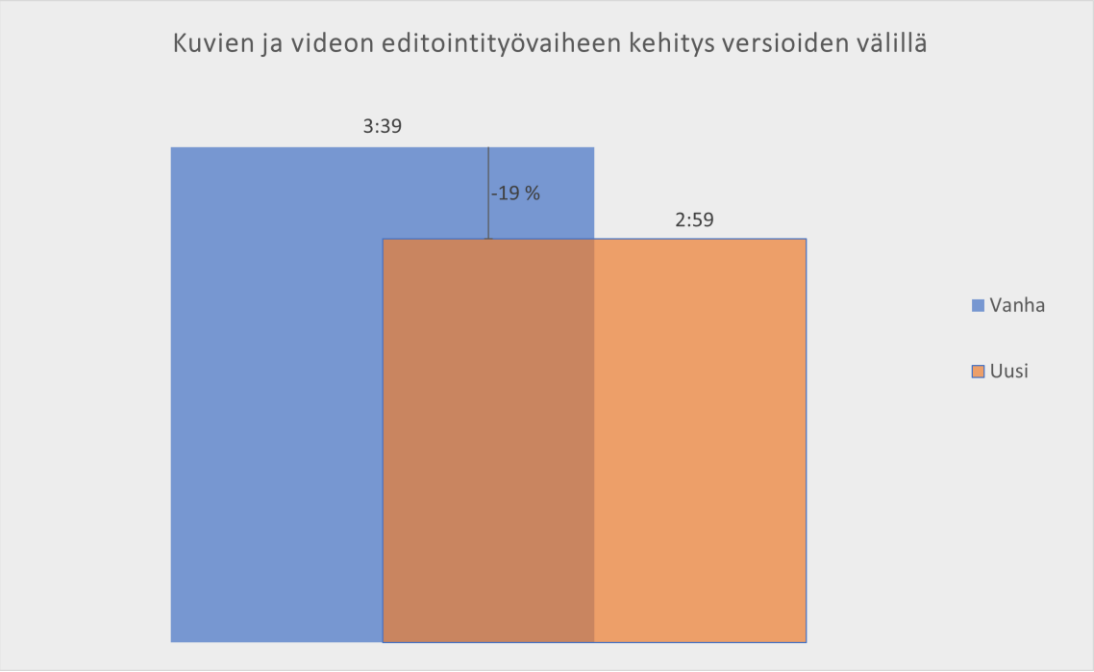
5.3 Havainnointi parannusten jälkeen ja analyysi kehityksestä

Ensimmäiset parannukset kohdistuivat editoinnissa käytettyyn sovellukseen. Tutkijan tehtävän oli kellottaa prosessin kesto uudella sovellusversiolla. Sovelluskehittäjä ei kuitenkaan halunnut julkaista vielä uutta versiota sovelluksesta, joten tutkija ei voinut kellottaa freelancerin editointiaikoja. Tämän seurauksena tutkija kellotti vain omat editointiaikansa uudella sovellusversiolla. Kuvausta varten rakennettiin uusi studio, johon kaikki kuvaus siirrettiin. Modumin asiakasyritys kuitenkin päätti lopettaa yhteistyön Modumin kanssa, ennen kuin uusi studio päästiin ottamaan käyttöön. Näiden tapahtumien seurauksena, tutkija ei pystynyt analysoida kuvausprosessissa tapahtunutta kehitystä alkuperäisen ja uuden studion välillä.

Sovelluspäivityksen tarkoituksena oli poistaa tarve käyttää erillistä sovellusta kuvien editointiin sekä vähentää turhaa odotusaikaa. Kaikki editointi haluttiin yhdistää yhteen sovellukseen. Näin poistettiin riippuvuus ulkopuoliseen sovellukseen ja tarve käyttää useampaa eri sovellusta työprosessissa. Tämän tavoitteena oli parantaa käyttäjäkokemusta, lyhentää prosessivaiheen kestoa sekä tulla omavaraisemmaksi.

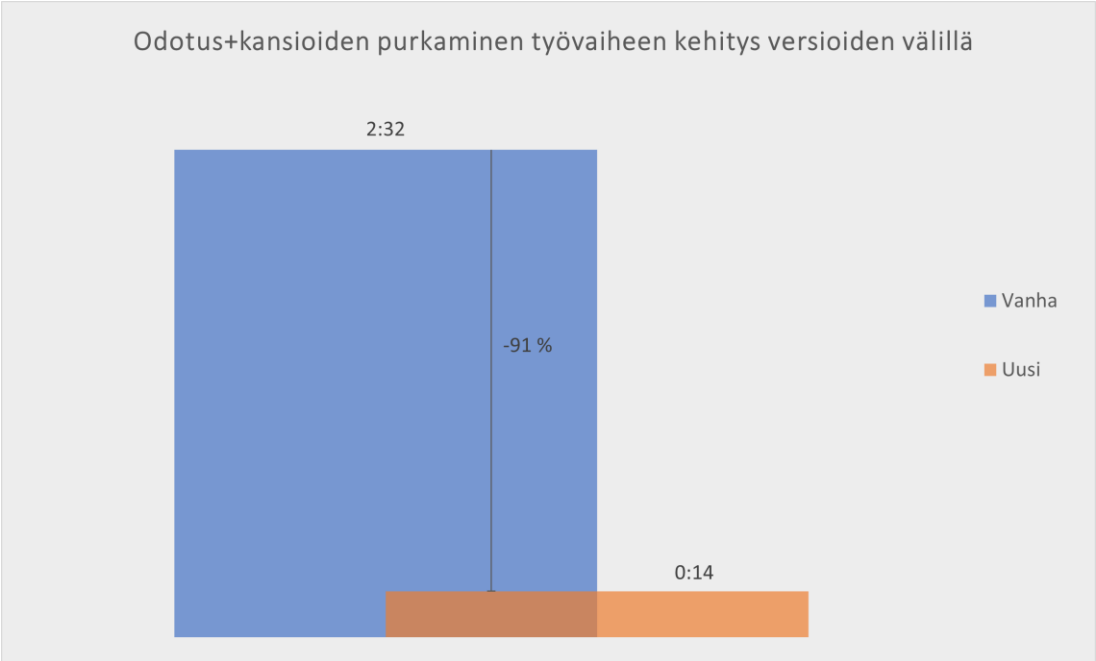
Työprosessi oli jaettu kuuteen eri vaiheeseen: kuvien siirtäminen kansioihin, kuvien editointi, odotus+kansioiden purkaminen, kuvien uploadaus, dokumentointi. Päivitys vaikutti kuvien editointiin, videon editointiin, odotus+kansioiden purkamiseen. Uudessa versiossa kuvien ja videon editointivaiheet yhdistyivät yhdeksi vaiheeksi. Tästä lähtien niitä käsiteltiin yhtenä vaiheena, nimeltään kuvien ja videon editointi. Koska päivitys ei vaikuttanut muihin vaiheisiin, ei niitä tarvinnut käsitellä.

Vanhassa sovellusversiossa kuvien ja videon editointiin kului keskimäärin 3 minuuttia ja 40 sekuntia per auto (Kuvio 6). Uudessa versiossa tämä aika oli vähentynyt keskimäärin 2 minuuttiin ja 59 sekuntiin. Prosessivaiheen kesto oli siis keskimäärin 41 sekuntia vähemmän per auto. Tämä tarkoitti 19 % eroa versioiden välillä. Positiivista kehitystä tapahtui kaikilla mittareilla; huonoin aika, paras aika, mediaani, huonoimman ja parhaimman ajan erotus, keskihajonta. Uuden version ansiosta erillisestä kuvien editointiin käytetystä sovelluksesta voitiin luopua. Päivityksen ansiosta käyttäjäkokemus kehittyi tämän vaiheen osalta, koska käyttäjä voi suorittaa kaiken editoinnin samassa sovelluksessa, jota oli myös mielekkäämpi ja nopeampi käyttää. Tämä osa sovelluspäivityksestä oli erittäin onnistunut.



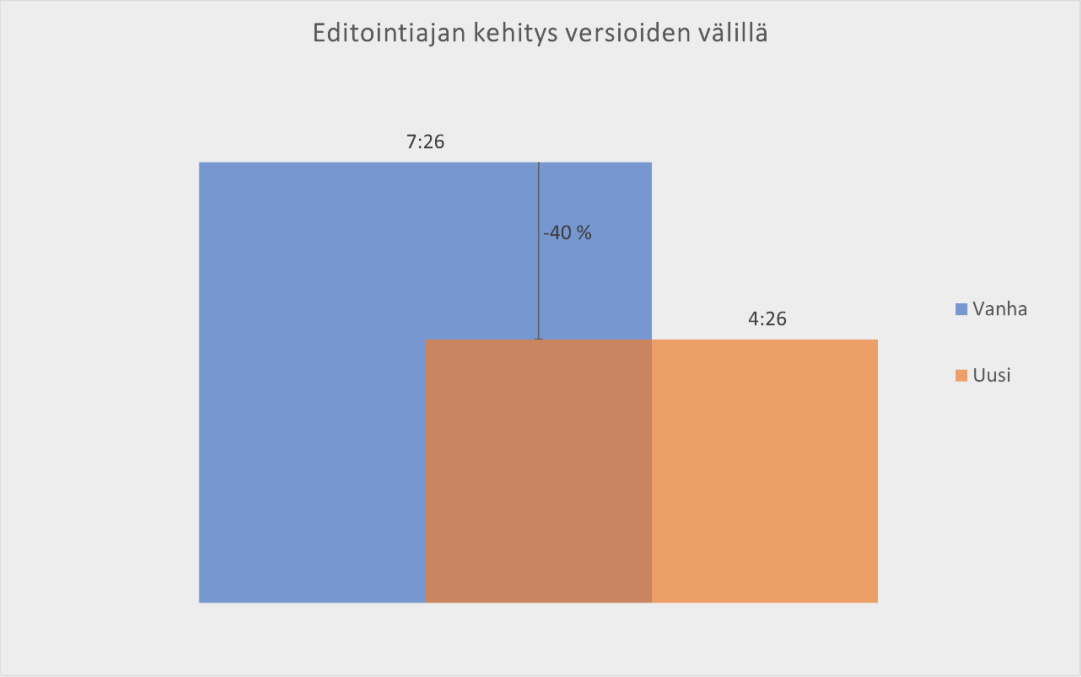
Kuvio 6: Kuvien ja videon editointityövaiheen kehitys versioiden välillä

Odotus+kansioiden purkaminen kesti vanhassa sovellusversiossa keskimäärin 2 minuuttia ja 33 sekuntia per auto (Kuvio 7). Uudessa versiossa sama vaihe oli keskimäärin vain 14 sekuntia. Prosessivaiheen kesto oli keskimäärin jopa 2 minuuttia ja 19 sekuntia vähemmän per auto. Parannusta oli siis tapahtunut 91 % versioiden välillä. Positiivista kehitystä tapahtui taas kaikilla käytetyillä mittareilla. Uudessa versiossa aika piti sisällään käytännössä pelkästään kansioiden purkamiseen kuluvan ajan. Päivityksen ansioista myös tämän vaiheen kohdalla käyttäjäkokemus parantui. Odotukseen kulunut aika oli lähes kokonaan kadonnut. Tämän päivityksen ansioista editoija pystyi tekemään työtä täysin omaan tahtiin, eikä hänen tarvinnut turhaan odottaa tämän vaiheen valmistumista. Tämäkin osa sovelluspäivityksestä oli siis erittäin onnistunut.



Kuvio 7: Odotus+kansioiden purkaminen työvaiheen kehitys versioiden välillä

Vanhalla versiolla editointiin kului keskimäärin 7 minuuttia ja 26 sekuntia. Uudella versiolla editointiin kului keskimäärin 4 minuuttia ja 27 sekuntia. Uudessa versiossa editointi siis kesti keskimäärin 2 minuuttia ja 59 sekuntia vähemmän kuin vanhassa versiossa. Kehitystä oli tapahtunut siis 40 % (Kuvio 8). Sovelluspäivitys täytti kaikki sille asetetut tavoitteet. Kokonaisvaltaisesti päivitys oli onnistunut.



Kuvio 8: Editointiajan kehitys versioiden välillä

6 Sovelluskehitysmenetelmien vertaaminen

Käytetty kehitysmenetelmä oli hyvin strukturoimatonta ja vapaata. Projektissa ei tietoisesti käytetty apuna mitään ohjelmistokehityksen metodeja. Ohjelmistokehitys muistutti eniten prototyyppimenetelmää. Aluksi luotiin sovellusversio, jossa oli vähän ominaisuuksia ja ajan myötä niitä sitten lisättiin ja laajennettiin käyttäjiltä saadun palautteen perusteella. Sovelluksen ydin pysyi samana alusta asti ja sen ympärille rakennettuja ominaisuuksia muutettiin. Kuvaus sopii parhaiten evoluutioprototyyppimenetelmän kanssa. Tämän menetelmän ominaisuutena on ajan, rahan ja vaivan säästäminen. Juuri ajan ja rahan säästäminen ovat niitä ominaisuuksia mitä Modum Oy arvostaa.

Nykyisen menetelmän hyvänä puolena oli keskittyminen olennaiseen. Aikaa ei kulunut mihinkään muuhun, kun itse sovelluksen kehittämisen ympärille. Ajankäytön pystyi jakamaan käyttäjien kuuntelemiseen, kehittämiseen, testaamiseen ja käyttöönottoon. Tämä toimi vain sen takia, koska projektilla ei ollut johtajaa, tiimi oli hyvin pieni eikä projektin kehitystä ollut tarpeellista seurata. Jos kyseessä olisi ollut isompi tiimi ja projektin kehitystä olisi tarvinnut seurata olisi järkevää tutkia ketterien menetelmien tarjontaan.

Sovellukselle tapahtunut kehitys ei ollut yhtäjaksoista ja tiimin koko oli edelleen pieni. Senhetkinen kehitysmenetelmä sopi hyvin Modum Oy:n tarpeisiin. Jos tilanteessa ei tapahdu minkäänlaisia muutoksia ei ole järkevää käyttää aikaa eri kehitysmenetelmien tutkimiseen ja implementointiin. Senhetkinen menetelmä tarjosi mahdollisimman suuren osan ajankäytöstä kohdistuvan itse sovelluksen kehittämiseen.

7 Pohdinta ja kehitysehdotuksia

Tutkimuksen toteuttaminen sujui mukavasti. Käynnissä oleva pandemia oli vielä niin alkuvaiheissa, että se ei haitannut tutkimuksen toteuttamista mitenkään. Tutkimuksessa mukana olleet henkilöt olivat jo entuudestaan tuttuja, joten sillä saattoi olla vaikutusta tutkimukseen. Tutkija kuitenkin yritti pitää tilanteen mahdollisimman objektiivisena eikä antanut tämän vaikuttaa tilanteeseen. Tutkimuksessa käytetyt menetelmät olivat oikeita valintoja. Havainnointi oli paras mahdollinen vaihtoehto tutkimaan työprosessin kestoa. Haastattelut toimivat hyvin havainnoinnin tukena. Haastatteluissa kysytyt kysymykset olivat hyviä, mutta haastattelut itsessään olisivat voineet olla strukturoidumpia. Haastattelut olivat ehkä hieman liian rentoja, eikä niille annettu tarpeeksi painoarvoa. Kysymykset ja niihin saadut vastaukset olivat hyviä, mutta vastauksiin olisi voinut syventyä vielä enemmän. Tutkimuksessa saadut tulokset olivat ajallisesti reliaabeleita. Vaikka dataa kerättiin useampana eri päivänä, toimintatavat ja laitteet pysyivät kuitenkin samoina. Tutkimuksen metodit olivat myös reliaabeleita. Oli hyvin johdonmukaista käyttää prosessin kestoa tutkittaessa havainnointia aineistonhankintamenetelmänä sekä tämän tukena haastatteluita, joissa haastateltiin itse työtä tekevää henkilöä.

Tutkimus sujui hyvin mutkattomasti, mutta parannettavaa silti löytyy. Selkein ja tutkimuksessa eniten vaivaa aiheuttanut asia oli kommunikaatio. Kommunikaatiota tapahtui liian vähän ja liian harvoin. Se oli myös aika epämääräistä. Tarkkoja päivämääriä ei lyöty lukkoon ja asioista ei tehty selkeitä lopullisia päätöksiä. Dokumentointi oli toinen selkeä parannuskohta. Ehdotettuja ominaisuuksia, parannuksia ja bugeja ei dokumentoitu tarpeeksi hyvin. Jotkin näistä unohdettiin kokonaan ja osa implementoitiin, mutta tieto siitä ei tullut kaikille käyttäjille heti. Tätä varten oli kylläkin kehitetty oma kanava, mutta sitä ei ollut vielä otettu käyttöön. Uudella sovelluspäivityksellä oli tarkoitus myös lisätä toimintoja, joita voisi suorittaa pelkällä näppäimistöllä käyttämättä hiirtä. Tämä idea kannattaisi viedä vielä pidemmälle, jotta sovellusta pystyisi käyttämään pelkästään näppäimistöllä tai hiirellä. Tämä lisäisi käyttäjäystävällisyyttä, jolloin käyttäjän ei tarvitsisi jatkuvasti vaihdella kumpaa laitetta käyttää. Tutkimuksessa todettiin, että ainakin yksi käyttäjistä suosi sovelluksen käyttöä hiirellä. Tutkija sekä sovelluskehittäjä taas suosivat sovelluksen käyttöä pelkästään näppäimistöllä. Molempien ominaisuuksien parantaminen täyttäisi kaikkien käyttäjien toiveet. Olisi myös kannattavaa tutkia, että voisiko joitakin työvaiheita automatisoida.

Tutkimuksen kannalta olisi ollut hyvä, että tutkija olisi päässyt kellottamaan prosessin kehityksen myös uudessa studiossa. Tämä ei kuitenkaan ollut mahdollista ja se johtui tutkimuksen ulkopuolisista tekijöistä. Tällä datalla ei välttämättä olisi ollut tulevaisuudessa edes käyttöä, koska kyseinen asiakas päätti lopettaa yhteistyön. Se olisi tuonut tutkimukseen lisää laajuutta. Myös editorin uuden päivityksen vaikutusta työvaiheeseen olisi ollut mukava tutkia useammalla eri käyttäjällä. Vaikka tutkimuksessa saatiin tarvittava data, olisi useamman käyttäjän havainnoiminen antanut vielä enemmän tietoa päivityksen vaikutuksesta työvaiheeseen.

Lähteet

Painetut

Sheard, J. 2018. Research Methods (Second Edition). Chapter 18 - Quantitative data analysis. Chandos Publishing.

Haikala, I. & Mikkonen, T. 2011. Ohjelmistotuotannon käytännöt. Helsinki: Talentum.

Sähköiset

Pulkkanen, A. 2020. Sinunkin kannattaa valita: 6 yleistä menetelmää projektityöhön (sis. Agile, Waterfall ja Kanban). Viitattu 13.11.2020. https://www.agendium.com/post/agile-waterfall-kanban-6-projektinhallintamenetelmaa

Agile Alliance. Agile 101. Viitattu 13.11.2020. https://www.agilealliance.org/agile101/

Agile Alliance. Scrum. Viitattu 13.11.2020. https://www.agilealliance.org/glossary/scrum/

Scrum Guide. The Scrum Guide™. Viitattu 13.11.2020. https://www.scrumguides.org/scrum-guide.html

Wells, D. 2013. Extreme Programming: A gentle introduction. Viitattu 13.11.2020. http://www.extremeprogramming.org/

Agile Alliance. Extreme Programming. Viitattu 13.11.2020. https://www.agilealliance.org/glossary/xp/

Lean IT Association. 2015. Lean IT & Lean IT Association Manifesto. Viitattu 13.11.2020. https://drive.google.com/file/d/0ByI-D4zDFqBDYXZad0xLRGx6dEE/view

Heunks, J. 2014. Lean IT In 3 Minutes. Viitattu 13.11.2020. https://www.vanharen.net/blog/lean-it-in-3-minutes/

UKEssays. 2018. The History Of The Waterfall Model Information Technology Essay. Viitattu 13.11.2020. https://www.ukessays.com/essays/information-technology/the-history-of-the-waterfall-model-information-technology-essay.php

GeekTonight. 2019. Classical Waterfall Model | Software Engineering. Viitattu 13.11.2020. https://www.geektonight.com/classical-waterfall-model-software-engineering/

Innorobix Automation. System Development Life Cycle – SDLC. Viitattu 13.11.2020. https://www.innorobix.com/uncategorized/sdlc/

IONOS. 2019. Waterfall methodology. Viitattu 13.11.2020. https://www.ionos.com/digitalguide/websites/web-development/waterfall-methodology/

GeeksforGeeks. 2019. Software Engineering | Classical Waterfall Model. Viitattu 13.11.2020 https://www.geeksforgeeks.org/software-engineering-classical-waterfall-model/

Tutorials Point. SDLC - Software Prototype Model. Viitattu 13.11.2020 https://www.tutorialspoint.com/sdlc/sdlc_software_prototyping.htm

Guru99. Prototyping Model in Software Engineering: Methodology, Process, Approach. Viitattu 13.11.2020. https://www.guru99.com/software-engineering-prototyping-model.html

Course Hero. 2016. Extreme prototyping used at web applications mainly. Viitattu 3.12.2020. https://www.coursehero.com/file/p5pi4vv/Extreme-prototyping-used-at-web-applications-mainly-Basically-it-breaks-down/

Saaranen-Kauppinen, A. & Puusniekka, A. 2006. KvaliMOTV - Menetelmäopetuksen tietovaranto. Tampere: Yhteiskuntatieteellinen tietoarkisto. Viitattu 13.11.2020. https://www.fsd.tuni.fi/menetelmaopetus

Jyväskylän yliopisto. 2015. Havainnointi eli observointi. Viitattu 13.11.2020. https://koppa.jyu.fi/avoimet/hum/menetelmapolkuja/menetelmapolku/aineistonhankintamenetelmat/havainnointi-eli-observointi-osallistuminen-ja-kenttaetyoe

Jyväskylän yliopisto. 2015. Määrällinen analyysi. Viitattu 13.11.2020. https://koppa.jyu.fi/avoimet/hum/menetelmapolkuja/menetelmapolku/aineiston-analyysimenetelmat/maarallinen-analyysi

Jyväskylän yliopisto. 2015. Pitkittäistutkimus. Viitattu 13.11.2020. https://koppa.jyu.fi/avoimet/hum/menetelmapolkuja/menetelmapolku/tutkimusstrategiat/pitkittaistutkimus

Jyväskylän yliopisto. 2015. Tilastollisesti kuvaava analyysi. Viitattu 13.11.2020. https://koppa.jyu.fi/avoimet/hum/menetelmapolkuja/menetelmapolku/aineiston-analyysimenetelmat/tilastollisesti-kuvaava-analyysi

Jyväskylän yliopisto. 2015. Vertaileva tutkimus. Viitattu 13.11.2020. https://koppa.jyu.fi/avoimet/hum/menetelmapolkuja/menetelmapolku/tutkimusstrategiat/vertaileva-tutkimus

Jyväskylän yliopisto. 2020 Haastattelut. Viitattu 13.11.2020. https://koppa.jyu.fi/avoimet/hum/menetelmapolkuja/menetelmapolku/aineistonhankintamenetelmat/haastattelut

Julkaisemattomat

Modum Oy, yritykseltä hankittu tieto.

Freelancer 2020, Modum Oy. Haastattelut 22.1.2020-31.1.2020.

Kuviot

Liitteet

Liite 1: Autojen kuvauksen kellotus

Autojen kuvauksen kellotus.xlsx