Haaga-Helia
ammattikorkeakoulu Oy

# Vaatimushallintaprosessin uudistaminen

Merja Saarinen

Opinnäytetyö
Tietojenkäsittelyn koulutusohjelma
2016

Haaga-Helia ammattikorkeakoulu Oy

**Tiivistelmä**

22.10.2016

**Tekijä(t)**
Merja Saarinen

**Koulutusohjelma**
Tietojenkäsittelyn koulutusohjelma

**Opinnäytetyön otsikko**

Vaatimushallintaprosessin uudistaminen

**Sivu- ja liitesivumäärä**
27

Tämän tutkimuksen tavoitteena on kuvata, miten toimeksiantajayrityksessä määritelty vaatimushallintaprosessi eroaa kirjallisuudessa yleisesti esitetyistä malleista. Lisäksi se tarkastelee, mitä ongelmia toimeksiantaja pyrkii uudella määrittelyllä ratkaisemaan sekä miten vaatimushallinnan osa-alueet (muutoshallinta, versionhallinta, vaatimusten tilojen seuranta ja vaatimusten jäljitettävyys) prosessissa toteutuvat. Toimeksiantajan pyrkimyksenä on korvata yrityksessä käytössä olevat erilaiset tuotekehitysprosessit yhdellä prosessilla vuoden 2016 aikana.

Määrittelytyö on opinnäytetyön kirjoittamisen aikana vielä kesken. Lopullisia tuloksia ei siis voida kertoa. Tutkimuskysymyksiin vastataan työn aikana saadun aineiston ja haastattelujen avulla.

Tutkimuksen alussa tutustutaan vaatimusmäärittelyyn, vaatimushallintaan ja vaatimukseen käsitteinä. Sen jälkeen kuvaillaan lyhyesti vesiputousmallia ja ketterään menetelmään perustuvaa prosessimallia.

Empiirisessä osassa kuvataan muutoksen syitä, taustaa sekä itse muutosprojektia. Lisäksi tutustutaan toimeksiantajan määrittämään prosessimalliin ja tarkastellaan pilottihanketta. Empiriaosuus pohjautuu toimeksiantajan materiaaliin sekä haastatteluihin yrityksen sisällä.

Tutkimus pyrkii selvittämään muutosprosessin eroja ja yhtäläisyyksiä verrattuna alan kirjallisuuteen. Mitkä ovat ne ongelmat, joita prosessilla pyritään ratkaisemaan?

Tutkimuksen tuloksia voidaan hyödyntää samankaltaisia prosessimuutoksia harkitsevissa yrityksissä. Toimeksiantajalle tutkimus tarjoaa tietoa prosessin määrittelyn kattavuudesta sekä tarjoaa jatkotutkimusaiheita.

**Asiasanat**
vaatimushallinta, ketterä menetelmä, Scrum, käyttäjätarina, BDD, tuotteen kehitysjono

## Sisällys

# 1 Johdanto

Tämän toiminnallisiin menetelmiin perustuvan opinnäytetyön aiheena on vaatimushallintaprosessin uudelleen määrittely toimeksiantajayrityksessä. Aihe on varsin ajankohtainen erityisesti ohjelmistotoimialalla, jossa sekä vaatimukset että niihin liittyvä ympäristö muuttuvat nopeasti. Ohjelmiston on täytettävä erilaisten sidosryhmien odotukset, ja samaan aikaan kitkattomasti nivouduttava muihin järjestelmiin. Yhtäältä sen tulee täyttää käyttäjien odotukset, toisaalta sen on oltava helposti konfiguroitavissa ja muokattavissa useiden käyttäjäryhmien tarpeisiin. Sen on myös mukauduttava ympäristön asettamiin ehtoihin. Vaatimushallinnan avulla tunnistetaan ohjelmistoihin kohdistuvia muutospaineita, ja hallitaan ja seurataan muutoksia ohjelmiston elinkaaren aikana. Onnistunut vaatimushallinta toimii tehokkaasti, ja sen avulla luodaan jatkuvasti uusia, positiiviseen käyttäjäkokemukseen johtavia tuotteita. Näin sitä voidaan kiistatta pitää yhtenä merkityksellisimpänä menestystekijänä ohjelmistoalan yrityksissä.

Opinnäytetyön käsittelemän prosessimuutoksen taustalla on pyrkiä ymmärtämään asiakkaiden vaatimuksia entistä paremmin. Tähän pyrittiin rakentamalla varsinaisen prosessin myötä käytännöt, jossa liiketoiminta on lähempänä tuotekehitystä, ja jossa vaatimukset kirjoitetaan asiakkaan ”kielellä” eli käyttäjätarinoina. Muutoksia hallitaan ja priorisoidaan työjonojen avulla. Näin muutospyyntöihin on helpompi ja nopeampi reagoida mahdollisimman varhaisessa vaiheessa. Aiemmista tuotekehitysmalleista ja prosesseista luovuttiin. Ne korvattiin prosessimuutoksen myötä luodulla yhdellä prosessilla, jota kutsuttiin yksinkertaisesti vaatimushallintaprosessiksi.

Prosessia tarkasteltaessa ilmenee, että se on ottanut vaikutteita sekä perinteisestä että erityisesti ketterästä menetelmästä. Perinteisen mallin mukainen hallittu päätöksentekomekanismi, jossa vaiheesta toiseen siirrytään tarkkojen suunnitelmien ja analyysien jälkeen, on korvattu mallilla, jossa päätöksenteko-organisaatiot tekevät päätöksiä tietyissä prosessin vaiheissa liiketoiminnan tavoitteisiin perustuen. Prosessista ovat löydettävissä perinteisen menetelmän vaiheet, mutta ne toteutuvat ketterän mallin mukaisena. Esimerkiksi perinteisen määrittelyvaiheen tarkoitus on kerätä ja ymmärtää vaatimukset, myös tietokantaa, arkkitehtuuria ja algoritmeja koskevat vaatimukset. Toimeksiantajan prosessissa vaatimukset pyritään kuvaamaan käyttäjätarinoina, jotka analysoidaan nopeasti. Vaatimuksia ei määritellä algoritmien tai tietokannan tasolla, vaan lähtökohtana ovat asiakkaan ilmaisemat toiminnalliset vaatimukset. Nämä vaatimukset priorisoidaan, ja tärkeimpien perusteella määritellään MVP (Minimum Viable Product). Suunnitteluvaiheessa tutkitaan toteuttamisen kannalta tärkeimpiä asioita (resurssit, markkinat jne.). Perinteisen mallin mukainen toteutusvaihe eli koodaus alkaa prosessissa

suunnittelun jälkeen. Toteutus tapahtuu ketterällä menetelmällä (Scrum). Tällainen ohjelmistokehitys sopii hyvin ympäristöön, joka operoi nopeasti muuttuvilla markkinoilla kansainvälisessä ympäristössä.

Tämä opinnäytetyö pyrkii kuvaamaan uudistamisprojektin tuloksena syntynyttä vaatimushallinnan prosessia.

Opinnäytetyön tutkimuskysymyksinä ovat

1. mitä eroja ja yhtäläisyyksiä on vaatimushallintaprosessissa käytettäessä perinteistä tai ketterää menetelmää
2. mitä ongelmia prosessilla pyritään ratkaisemaan
3. miten muutosten hallinta, versiointi, tilojen seuranta ja jäljitys käytännössä tulevat toteutumaan

Työn alkuosassa määritellään, mikä on vaatimus ja mitä on vaatimushallinta. Näiden jälkeen kuvaillaan lyhyesti vesiputousmallia sekä Scrum-menetelmää, ja esitellään, miten vaatimushallinta näissä malleissa toteutuu. Lopuksi esitetään yhteenveto mallien välillä. Tutkimuksen keskivaiheilla tutustutaan prosessimuutosprojektiin. Sen jälkeen kuvataan toimeksiantajan määrittämää vaatimushallintaprosessia, ja verrataan sitä tietoperustaan. Lopuksi tehdään yhteenveto ja arvioidaan prosessin sisältöä suhteessa tietoperustassa käsiteltyihin teoreettisiin malleihin. Viimeisenä kerrotaan päätelmiä siitä, miten muutosten hallinta, versiointi, tilojen seuranta ja jäljitys käytännössä tulevat toteutumaan.

Toimeksiantajalle työ tarjoaa näkemyksen, jonka avulla se voi paneutua mahdollisiin puutteisiin sekä havaittuihin ongelmakohtiin, ja toisaalta kehittää edelleen jo toimivia asioita. Muille mahdollisesti prosessimuutosta harkitseville yrityksille työ antaa tietoa, jota voidaan ottaa huomioon muutoksen suunnittelussa.

Työ rajautuu vaatimushallinnan tarkasteluun. Empiirisessä osuudessa tarkastellaan toimeksiantajan määrittämä vaatimushallinnan prosessia, sekä itse vaatimuskäsittelyä.

# 2 Vaatimus

Kun tarkastellaan vaatimusta käsitteenä, huomataan, että sitä voidaan tulkita monin eri tavoin, tulkitsijasta riippuen. Vaatimus voi yhdelle merkitä isomman toiminnallisuuden kuvaamista, kun taas toiselle se on tuon isomman joukon sisältä löytyvä pienempi osakokonaisuus, vaikkapa väri, jolla valittavissa oleva vaihtoehto näytetään. Vaatimukset luokitellaan yleisesti toiminnallisiin ja ei –toiminnallisiin vaatimuksiin. Toiminnalliset kertovat siitä, miten ominaisuudet on toteutettu käyttäjän kannalta katsottuna. Ei-toiminnalliset kuvaavat ne reunaehdot, jotka liittyvät toiminnallisiin vaatimuksiin (suorituskyky, integraatio jne.). Ne voivat esittää myös reunaehtoja toiminnallisille vaatimuksille (esimerkiksi käyttöliittymän asettamat rajoitukset).

Kansainvälisesti tunnustettu ISO –standardi on listannut vaatimuksille asetettavia ominaisuuksia. Ominaisuudet ovat yleisesti tunnustettuja, ja niitä tulisi pyrkiä noudattamaan vaatimuksia esitettäessä ja kuvattaessa. Vaatimushallintaan selkeästi liittyvänä ominaisuutena standardi mainitsee jäljitettävyyden. Sen mukaan vaatimus tulee voida jäljittää paitsi eteen- ja taaksepäin, myös ylemmän tason vaatimuksiin siten, että sidosryhmän tarve tulee selvennetyksi, ja myös alaspäin lähdekoodiin asti (ISO/IEC/IEEE 29148:2011, 11). Edelleen standardi luettelee mm. tarpeellisuuden, toteutusriippumattomuuden, yksikäsitteisyyden, ristiriidattomuuden, täydellisyyden, yksilöllisyyden, toteutettavuuden ja todennettavuuden.

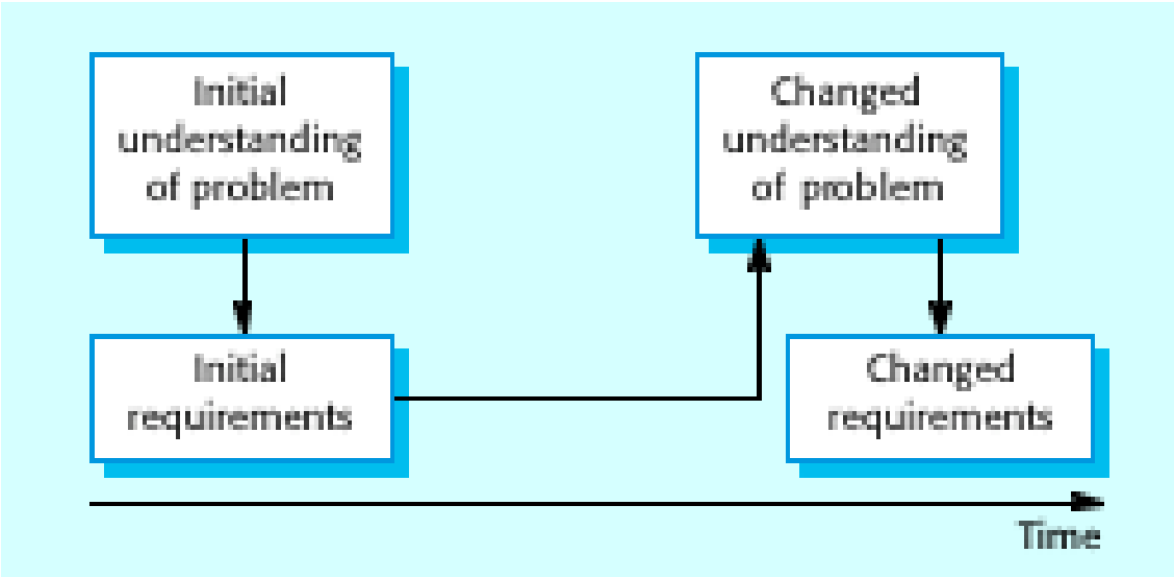
# 3 Vaatimushallinta

Vaatimushallinnan käsitettä on määritelty kirjallisuudessa hieman eri tavoin. Sommerville (2004) kuvaa vaatimushallintaa sanoen, että se on prosessi, jonka avulla hallitaan muuttuvia vaatimuksia vaatimustenmäärittelyprosessin ja järjestelmän kehityksen aikana. Vaatimushallinnan toteutus voi poiketa riippuen siitä, mitä tuotekehitysmallia seurataan. Vaatimushallintaan kuuluu vaatimuksista sopiminen, vaatimusten analysointi, dokumentointi, jäljitys, versiointi ja priorisointi.

Laamswerde (2009) viittaa artikkelissaan *Engineering Requirements for System Reliability and Security* erääseen USA:ssa tehtyyn tutkimukseen, johon osallistui 350 yritystä, ja jossa paljastui, että kolmannes aloitetuista yli 8000 tuotekehitysprojekteista jäi tekemättä, ja puolet niistä onnistui vain osittain. Syy oli huonosti määritellyissä vaatimuksissa. Euroopassa vastaavanlainen tutkimus, johon osallistui 3800 organisaatiota 17 maasta, osoitti samankaltaista tulosta. Ohjelmistoissa havaittujen ongelmien todettiin olevan huonojen vaatimusmäärittelyjen ja vaatimushallinnan seurausta.

Miksi vaatimusten määrittely on sitten niin vaikeaa? Laamswerde (2009) toteaa artikkelissaan, että vaatimuksia esittävillä sidosryhmillä on osittaisia ja ristiriitaisia toivomuksia. Ohjelmistoa suunniteltaessa olisi osattava huomioida nekin vaatimukset, joilla estetään mahdollinen ohjelmiston väärinkäyttö. Lisäksi vaatimusmäärittelyn tulee kattaa paitsi ylätason strategiset tavoitteet, myös aivan pienet, lähdekoodiin saakka ulottuvat vaatimukset. Sommerville (2004) puolestaan on todennut, että vaatimukset ovat väistämättä epätäydellisiä, ja uusia vaatimuksia syntyy kehitystyön edetessä, koska liiketoiminta muuttuu ja järjestelmän toimintaa aletaan ymmärtää entistä paremmin. Myös Sommervillen mielestä sidosryhmien erilaiset näkökannat aiheuttavat usein ristiriitaisia vaatimuksia. Teknisen käyttäjän vaatimukset eroavat loppukäyttäjän vaatimuksista. Sommervillen mukaan ei ole olemassa yhtä ainoaa oikeaa tapaa analysoida järjestelmän vaatimuksia, vaan vaatimukset tulisi kategorisoida sen mukaan, millaisia katsantokantoja tai näkökulmia ne tuotteeseen tai palveluun nähden edustavat. Kategorisoinnissa tulee ottaa huomioon järjestelmän palvelujen tuottajat ja vastaanottajat, insinöörit, jotka kehittävät ja ylläpitävät järjestelmää, erilaiset liiketoiminnalliset ja ei-toiminnalliset vaatimukset, markkinointi, määräykset ja säännöt ja epäsuorasti tuotteeseen liittyvät muut järjestelmät. Kehitystyön aikana alkuperäiset painoarvot saattavat muuttua.

Kuva alla hahmottaa Sommervillen näkemystä siitä, miten vaatimus muuttuu ongelman ymmärtämisen myötä (Sommerville 2004).




Kuva 1: Ian Sommerville (2004): Requirement evolution

Sommerville (2004) on listannut neljä erilaista vaatimustyyppiä, ja kuvannut, miten ne tyypillisesti muuttuvat.

Taulukko 1: Vaatimustyypit (Sommerville 2004)

| | |
|---|---|
| Muuttuvat vaatimukset | Vaatimukset muuttuvat ympäristön muuttumisen seurauksena. Esimerkiksi sairaalassa potilasrahoituksen muuttuessa on tarpeen kerätä erilaista hoitoon liittyvää tietoa kuin aikaisemmin. |
| Ajan kuluessa kehittyvät vaatimukset | Järjestelmän kehittyessä ymmärrys kasvaa ja vaatimuksia osataan kuvata paremmin. |
| Merkitykselliset vaatimukset | Esimerkiksi uuden järjestelmän mukanaan tuomat uudet käyttötavat synnyttävät uusia vaatimuksia. |
| Yhteensopivuuteen liittyvät vaatimukset | Prosessien, sääntöjen jne. muuttuessa on vaatimuksiakin muutettava vastaavasti |

Sommerville painottaa kolmea vaatimushallinnan suunnittelun kannalta olennaista pääelementtiä: vaatimusten tunnistaminen, muutosten hallinta ja jäljitettävyys. Suunnitteluvaiheessa täytyy päättää, miten vaatimukset yksilöllisesti tunnistetaan, ja mitä prosessia tullaan noudattamaan muutospyyntöjä analysoitaessa. Lisäksi on päätettävä, mihin asti vaatimuksia tulee jäljittää. Jäljitettävyydellä tarkoitetaan vaatimusten välisiä suhteita, niiden lähteitä sekä järjestelmän suunnittelua. Vaatimusten riippuvuudet toisiinsa on pystyttävä jäljittämään. Vaatimukset on voitava yhdistää niiden esittäjään eli sidosryhmään, samoin kuin itse järjestelmäsuunnitteluun.

Vaatimukset on talletettava tietokantaan. Muutoshallinnan eri vaiheet on pystyttävä määrittämään ja tieto, joka näiden vaiheiden välillä kulkee, on osittain automatisoitava. Vaatimukset tulee linkittää toisiinsa automaattisesti.

Muutoshallinnan tärkeimmät vaiheet ovat ongelmien analysointi, muutoksen analysointi, sen aiheuttamat kustannukset ja vaikutukset muihin vaatimuksiin sekä muutoksen toimeenpano, jonka myötä vaatimusdokumentti ja muut dokumentit päivitetään vastaavasti.

Laamswerde (2009) tukee näkemystä. Hänen mukaansa vaatimushallintaa on harjoitettava systemaattisesti, jotta toivottuihin tavoitteisiin päästään. Ensinnäkin, vaatimushallinnan on oltava tavoitehakuista. On tiedettävä, mitä järjestelmältä halutaan. Järjestelmää tulee toteuttaa pienin, inkrementaalisin askelin. Mitä myöhemmin virheitä löydetään, sen hankalammaksi ja kalliimmaksi niiden korjaaminen käy. Prosessin tulee olla muodollinen, silloin kun se on tarpeen, ja kevyempi käytännön tilanteissa.

Yhteenvetona voidaan todeta, että vaatimushallinta pyrkii hallitsemaan muutoksia, jotka kohdistuvat aiemmin kerättyihin vaatimuksiin. Puhutaan vaatimusten tunnistamisesta, jolla tarkoitetaan vaatimusten numeroimista tai muuten ”korvamerkitsemisestä” siten, että vaatimukset yksilöidään ja voidaan löytää myöhemminkin. Tunnistamisessa voidaan käyttää esimerkiksi numerointia, tai numero-kirjainyhdistelmiä. Vaatimukset talletetaan yhteisesti sovittuun paikkaan, josta ne poimitaan analysoitaviksi. Yleisesti hyväksytty paikka on tietokanta.

Vaatimushallinnalla seurataan, miten vaatimus ”elää”, eli miten se polveutuu ja miten sitä voidaan jäljittää. Kuten aiemmin kerrottin, ISO –standardin mukaan yksi vaatimukselle asetettu ominaisuus on jäljitettävyys. Sen avulla nähdään, mihin ylemmän tason vaatimukseen vaatimus liittyy ja missä kohden lähdekoodia vaatimus toteutuu. Näin siis voidaan seurata vaatimusta eteen- ja taaksepäin.

Vaatimukset versioidaan, jotta voidaan myöhemmin nähdä, miten ne alkuperäinen vaatimus muuntuu matkan varrella, tai millaisia eri toteutustapoja vaatimuksella on.

Vaatimuksiin liittyviä muutospyyntöjä tulee yleensä ottaen runsaasti. Muutoshallinnalla pyritään kontolloimaan vaatimusten toteuttamista edellä kerrotun mukaisesti (tunnistaminen, analysointi, versiointi).

# 4 Vesiputousmalli

Vesiputousmalli on tunnettu ohjelmistokehitysmalli, joka syntyi Winston E. Roycen 1970 – luvulla kirjoittaman artikkelin pohjalta. Mallista tuli hyvin suosittu. Ajan myötä siitä kehittyi uusia versioita, jotka eivät olleet niin ehdottomia kuin alkuperäinen malli.

Vesiputousmallissa vaiheita tunnetaan yleisesti viisi. Ne ovat määrittely, suunnittelu, toteutus, testaus ja ylläpito (Sommerville, 2011). Vaiheet toteutetaan järjestyksessä, ja kunkin vaiheen päättyminen on alku seuraavalle vaiheelle. Näin esimerkiksi muutoksia ei sallita enää sen jälkeen, kun suunnittelu on tehty, joskin nykyisissä versioissa sekin on tietyin ehdoin sallittua.

Vaatimushallinnan kannalta merkityksellisintä vesiputousmallissa on tapa, jolla muutospyyntöjä hallitaan. Alkuvaiheessa kerätyt vaatimukset kuvataan kattavasti vaatimusdokumenteissa. Muutospyynnöt kerätään, ja käsitellään alkuperäisistä vaatimuksista irrallaan. Niistä etsitään samankaltaisuuksia, joiden aikaansaamia hyötyjä tutkitaan suhteessa toteuttamisen kustannuksiin. Vesiputousmallissa työmäärät lasketaan konkreettisina, absoluuttisina arvoina, kuten esimerkiksi henkilötyöpäivinä tai tunteina, jolloin kannattavuuden arvioiminen tulee ainakin teoriassa matemaattisesti todistetuksi. Hyötyvaikutuksia voivat toki olla muutkin kuin taloudelliset mittarit, esimerkiksi strategiset tavoitteet ja halu päästä markkinoille varhaisessa vaiheessa tai haastajana.

Vesiputousmallissa asiat dokumentoidaan. Näin myös vaatimukset dokumentoidaan yksilöllisesti ja selvästi tunnistettavasti. Sekä vaatimukset että muutokset kuvataan tarkasti ja kattavasti. Dokumentointi on hidasta, ja muutosten aikaansaama uudelleen dokumentonti hidastuttaa prosessia entisestään.

Vesiputousmallia on verrattu raudan tuotantoprosessiin. Se sopii ohjelmistotuotantoon, jossa muutokset ovat vähäisiä, ja valmistettava ohjelmisto voidaan miettiä tarkasti etukäteen.

# 5 Ketterä menetelmä

Ketterä menetelmä perustuu Agile -manifestiin, jonka on laatinut Agile-allianssi (2001). Se sisältää neljä arvoa ja 12 periaatetta. Menetelmä arvostaa ihmisläheisyyttä enemmän kuin prosesseja ja työkaluja. Niinpä esimerkiksi yksilöiden keskeisellä vuorovaikutuksella on suuri painoarvo. Ketterä menetelmä kyseenalaistaa asioita, jotka ovat perinteisen mallin kulmakiviä. Näitä ovat mm. sopimusneuvottelut, suunnitelmien seuraaminen ja kattava dokumentonti. Menetelmässä todetaan, että muutoksiin reagoiminen on arvokkaampaa kuin orjallinen suunnitelman seuraaminen, ja tärkeämpää kuin sopimusneuvottelut, on saada aikaan toimiva ohjelmisto yhteistyössä asiakkaan kanssa.

Manifeston mukaan liiketoiminnan ammattilaisten ja kehittäjien täytyy työskennellä yhdessä päivittäin koko projektin ajan. Tällä tavalla varmistetaan, että tieto välittyy kehittäjille oikealla tavalla ja nopeasti. Kasvokkain keskustelu on tehokkaampaa kuin kirjalliset selvitykset. Menetelmä luottaa siihen, että tiimi kykenee toimimaan itsenäisesti, ja tuottamaan jatkuvasti nopeassa tahdissa uusia tuoteversioita. Muutoksia vastaanotetaan kesken kehitystyön, jopa aivan viime hetkillä. Tärkeämpää kuin säilyttää alkuperäinen tuotteeseen määritetty sisältö, on osata priorisoida vaatimuksia sitä mukaa, kun niitä vastaanotetaan, ja pysyä aikataulussa. Tämä voi tarkoittaa sitä, että tuotteen sisältö muuttuu matkan varrella. Sillä ei kuitenkaan ole merkitystä, koska priorisointi tapahtuu asiakkaan ehdoilla.

Ketterä menetelmä uskoo, että kehitystiimi osaa itse määritellä, miten työ kannattaa toteuttaa. Se siis itse määrittää arkkitehtuurit, kuvaa vaatimukset ja suunnittelee työn toteutuksen. Mittarina käytetään toimivaa ohjelmistoa. Sen sijaan tarpeetonta työtä, kuten esimerkiksi kattavaa dokumentointia tulee välttää. On hyve osata karsia sellaista työtä, jolla ei ole merkitystä asiakkaalle.

Siinä, missä perinteinen menetelmä luottaa vaatimusten kattavaan määrittelyyn, ja vaatimusdokumentteihin, ketterä menetelmä tallettaa vaatimukset ja muutospyynnöt työjonoihin priorisoitaviksi. Työjonoissa vaatimukset yksilöidään tunnistettavasti. Vaatimusten kuvaaminen tapahtuu epiikkeinä (laajempi toiminnallinen kokonaisuus) tai yksittäisinä käyttäjätarinoina, joilla kuvataan toiminnallisia vaatimuksia käyttäjän kannalta. Vähemmälle huomiolle yleensä jäävät ei-toiminnalliset vaatimukset. Muutosten vaikutusta kannattavuuteen arvoidaan yhdessä liiketoimintajohdon kanssa. Työmäärät voidaan mitata erilaisin mittarein, esimerkiksi S-M-L (Small, Medium, Large) –mittarilla, josta yleensä käytetään nimitystä T –paitamalli, tai tarinapisteinä, joissa isoimmat pisteet

saadaan, kun toteutus arvioidaan laajaksi. Näissä hankaluus on se, että pitää löytää vertailukohde, jolla abstraktit arvot saadaan muunnettua laskettaviksi arvoiksi.

Ketteristä menetelmistä on olemassa monia eri muunnelmia. Perusperiaatteet niissä ovat kuitenkin samat.

# 6 Vaatimushallinnan erot vesiputousprosessin ja ketterän menetelmän välillä

Vaatimushallinnan eri osa-alueet toteutuvat vesiputousprosessissa ja ketterässä menetelmässä eri tavoin.

Perinteisessä menetelmässä **muutoksia** hallitaan siten, että niitä ei oteta mukaan enää sen jälkeen, kun vaatimusten määrittely on tehty. Muutospyynnöt kerätään, ja niiden samankaltaisuudet tutkitaan. Lopuksi lasketaan, miten paljon kustannukset olisivat, ja mitä hyötyä toteutuksesta tulisi olemaan.

Ketterä menetelmä ottaa muutokset huomioon kesken kehityksen. Muutospyynnöistä keskustellaan saman tien niiden tullessa esille. Muutoksia hallitaan kehitysjonossa, jonka avulla priorisointi tapahtuu. Toteutuksen määrää priorisointi, eli muutos voidaan ottaa mukaan toteutukseen, mikäli se katsotaan riittävän arvokkaaksi. Jokin toinen vaatimus voidaan puolestaan pudottaa pois.

**Versionhallinta** toteutuu perinteisessä mallissa siten, että jokaisella tuoteversiolla on oma tunnisteensa. Muutokset dokumentoidaan ja talletetaan tietokantaan.

Ketterässä menetelmässä käytetään myös tunnisteita, mutta dokumentointi on huomattavasti kevyempää. Kirjallisen kuvauksen sijasta versiomuutos saattaa olla kuvattuna lyhyenä koodinpätkänä, tai parilla rivillä kerrottuna kehitysjonossa, joissa versioita hallitaan.

**Vaatimusten tilojen seuranta** toteutuu perinteisessä mallissa tarkan kontrollin alla. Muutokset analysoidaan, ja jos ne katsotaan toteuttamiskelpoisiksi, ne otetaan mukaan seuraavaan tuoteversioon. Yksittäisten vaatimusten toteutumista seurataan erilaisten määritysten kautta. Näitä tilamäärityksiä voivat olla esimerkiksi ”käynnissä”, ”suljettu”, ”hylätty”.

Ketterässä menetelmässä vaatimusten tiloja seurataan osana kehitysjonon tehtävien toteutusta. Tiloja voidaan kuvata kuten vesiputousmallissa. Priorisointi määrää toteutusjärjestyksen.

**Jäljitettävyys** on ISO –standardinkin mukaan yksi ominaisuus, joka vaatimukselle asetetaan. Se on kuitenkin käytännössä varsin haasteellinen toteuttaa. Perinteisessä mallissa vaatimusten jäljittäminen tapahtuu dokumentoinnin kautta. Vaatimusdokumentit

kuvaavat, mihin vaatimus liittyy, mitä sen toteuttaminen vaatii, ja mitä vaatimus saa aikaan.

Ketterässä menetelmässä jäljitettävyyden tulisi johtaa käyttäjätarinasta aina yksittäiseen tehtävään asti, ja päinvastoin. Näin liiketoimintajohto voi helpommin kommunikoida asiakkaan kanssa, ja kehitystiimi puolestaan nähdä, mitä kokonaisuutta yksittäisillä tehtävillä pyritään toteuttamaan.

Taulukko 1. Vaatimushallinnan osa-alueet vesiputousmallissa ja ketterässä menetelmässä

| **Vaatimushallinnan osa-alueet** | **Vesiputousmalli** | **Ketterä menetelmä** |
|---|---|---|
| Muutosten hallinta | Muutoksia ei pääsääntöisesti sallita vaatimusten määrittelyn jälkeen.<br><br>Muutokset käsitellään erikseen, samankaltaisuudet tutkitaan, ja toteuttamisesta aiheutuvat kustannukset ja hyödyt punnitaan. | Muutokset ovat osa jokapäiväistä kehitystyötä. Muutokset kirjataan kehitysjonoon, josta ne otetaan toteuttavaksi priorisoinnin määräämässä järjestyksessä. |
| Versionhallinta | Vaatimuksilla tulee olla yksilöllinen tunniste ja versiot on kyettävä erottamaan.. Vaatimukset talletetaan tietokantaan. | Vaatimukset yksilöidään ja versioidaan kehitysjonossa. |
| Vaatimusten tilojen seuranta | Tiloja seurataan prosessin edistyessä. | Vaatimusten tiloja seurataan kehitysjonon kautta. |
| Jäljitettävyys | Jäljittäminen tapahtuu vaatimusdokumenttien päivittämisen kautta. | Vaatimuksia jäljitetään kehitysjonon kautta. |

# 7 Vaatimushallintaprosessin määrittely toimeksiantajayrityksessä

Tässä opinnäytetyössä tarkastellaan, miten projekti toteutettiin sekä millainen vaatimushallintaprosessi sen tuloksena syntyi. Prosessia verrataan tietoperustassa esiteltyihin malleihin. Erityistä huomiota kiinnitetään siihen, sisältyvätkö tietoperustassa mainitut vaatimushallinnan osa-alueet (muutoshallinta, versionhallinta, vaatimusten tilojen seuranta, vaatimusten jäljittäminen) prosessin määrittelyyn.

Toimeksiantaja toimittaa IT -ratkaisuja teleoperaattoreille. Toiminta kattaa maailmanlaajuisesti noin 90 operaattoria noin 70 maassa. Pääkonttori on Suomessa, ja myyntitoimistoja on useissa maissa Euroopassa, Lähi-Idässä, Latinalaisessa Amerikassa ja Aasiassa. Työntekijöiden määrä on noin 900.

Yrityksen porfolio sisältää IT-ratkaisuja palvelujen, asiakkaiden ja laskutuksen hallintaan. Strategiana on auttaa palveluntuottajia muuntamaan liiketoimintaansa digitaalisten palvelujen suuntaan. Digitalisointi on muuttanut viime vuosina operaattoreiden kilpailuasemaa siten, että ne joutuvat kilpailemaan paitsi keskenään, myös eri sisällöntuottajien ja palveluntuottajien (Google, Facebook, Amazon) kanssa. Kuluttajat ovat entistä innokkaampia vaihtamaan palveluntarjoajaa, mikäli luvassa on parempia tarjouksia ja käyttökokemuksia. Näin operaattorien on pakko tuoda markkinoille uusia palvelukokonaisuuksia ja investoida asiakaskokemukseen.

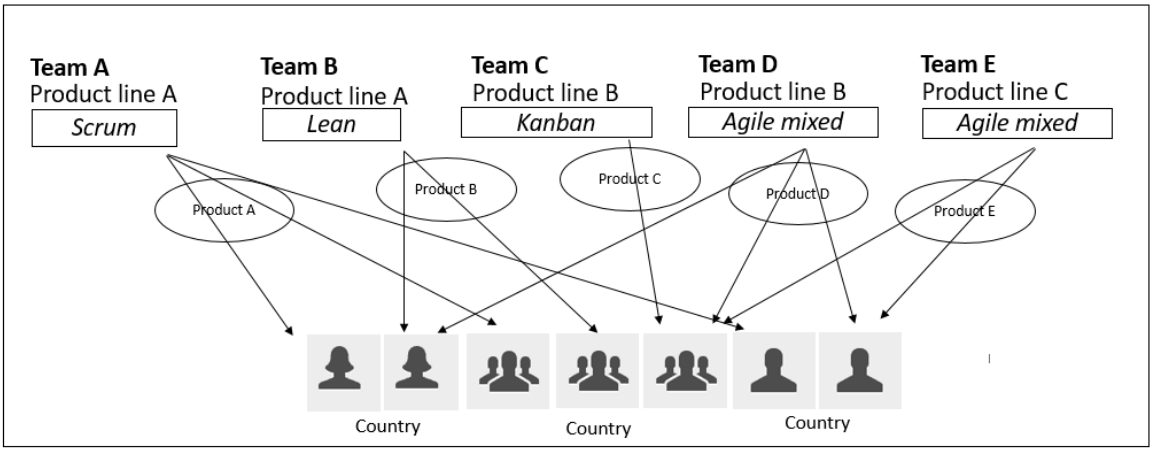
Toimeksiantaja panostaa tuotteisiin, jotka säilyttävät operaattorin aseman uudessa ja nopeasti muuttuvassa markkinaympäristössä. Tavoitteena on ennakoida kuluttajien ja operaattoreiden tarpeita ja luoda niihin sopivia tuotteita. Nopea reagointi muutoksiin on välttämätöntä. Ketterässä menetelmässä liiketoiminta ja kehittäjät toimivat päivittäisessä yhteistyössä ja muutoksiin reagoidaan kesken kehityksen. Se sopi yrityksen strategisiin tavoitteisiin, niinpä strategiaa noudattaen tuotekehitys ja -hallinta koulutettiin Scrum-periaatteiden käyttöön vuonna 2010. Tuotekehitys omaksuikin menetelmän, ja sitä alettiin soveltaa eri kehitystiimeissä. Tuotehallinta pyrki edelleen noudattamaan vesiputousprosessia.

Tuotehallinta vastasi vaatimusten keräämisestä liiketoiminnallisella ja toiminnallisella tasolla. Tuotekehityksessä vaatimukset pilkottiin teknisiksi vaatimuksiksi, ja edelleen tehtäviksi, jotka jaettiin kehitystiimeille. Tiimit priorisoivat vaatimuksia omien näkemystensä mukaisesti.

Ketterän menetelmän haasteeksi osoittautui erityisesti tiimien kansainvälisyys. Tuotteita kehitettiin kolmessa eri maassa, ja kehitystiimeissä saattoi olla jäseniä kaikista näistä kolmesta maasta. Ketterän menetelmän soveltaminen globaaleissa tiimeissä on vaikeaa jo pelkästään aika- ja kulttuurierojen vuoksi. Eräs kantavista Scrumin periaatteista on, että kasvokkain viestintä on kaikkein tehokkain kommunikoinnin muoto. Globaaleissa tiimeissä ainoa keino olla ”kasvokkain” olivat yhteydenpito nykyaikaisilla viestintävälineillä (Skype), videopuhelut ja tavalliset puhelut, joilla ei kuitenkaan saavutettu aitoa Scrumin tavoitetta nopeasta ja välittömästä kommunikoinnista. Liiketoimintajohto oli myös etäällä tiimeistä.

Tiimejä ei ohjattu tiukasti, vaan niillä oli vapaus soveltaa Scrumin periaatteita omilla säännöillään. Kehitystyö toteutui eri tavoin eri tiimeissä, ja tämä puolestaan teki seurannasta hankalaa.

Kuva 4 havainnollistaa tilannetta.




Kuva 4. Globaalit tuotekehitystiimit

Ketterä menetelmä luottaa siihen, että tiimin sisältä nousevat parhaat arkkitehtuurit, vaatimukset ja suunnitelmat. Osaamisen tason ollessa riittävä korkea, näin tapahtuikin. Aina ei tuloksia kuitenkaan syntynyt toivotulla tavalla.

Edelleen, ketterän menetelmän mukaan motivoitunut tiimi saa työn tehdyksi. Tiimien sisällä olikin usko siihen, että työt onnistuvat. Vaatimusten hallinta, ja erityisesti muutoshallinta, osoittautuivat kuitenkin haasteellisiksi. Muutoksia vastaanotettiin ja yritettiin lisätä tuotteisiin jatkuvasti, jolloin seurauksena alkuperäiset työmääräarviot muuttuivat moninkertaisiksi. Vaatimukset tulivat asiakkailta, ja tuotteita kehitettiinkin asiakkaiden tuomien muutosten ajamana, ei niinkään hallittuna tuotekehitysprosessin mukaisena kehitystyönä. Laatu kärsi jatkuvista aikataulupaineista johtuen. Asiakkaiden

vaatimuksia ei aina ymmärretty oikein alusta lähtien, ja tämä osaltaan toi korjaus- ja muutospainetta kehitystyöhön.

Vaatimuksia analysoitaessa olisi tärkeää nähdä niiden riippuvuudet muihin komponentteihin, myös eri tuotteiden välillä. Tällöin vältetään saman työn toteuttaminen monella eri tavalla, ja ratkaisut voidaan uudelleen käyttää ja viedä hallitusti kehitettävään tuotteeseen. Riippuvuuksien hallinta on haasteellista. Sitä varten on kehitetty mitä erilaisimpia työkaluja. Työkalut eivät kuitenkaan ratkaise ongelmaa, mikäli työtä ei ole organisoitu siten, että riippuvuudet voidaan todeta. Jokaisella yrityksellä on aivan omanlaisensa ympäristö ja omanlaisensa tapa järjestää asioita, jolloin riippuvuuksien hallintaa ei voida suoraan kopioida mistään mallista, vaan se pitää yrityksen miettiä ja organisoida itse. Kun tuotekehitystiimit toimivat fyysisesti etäällä toisistaan, ja kehittivät eri tuotteita ilman yhtenäisiä toimintatapoja- ja työkaluja, ei riippuvuuksia voitu todeta riittävän tehokkaasti, vaan samaa työtä tehtiin hiukan eri tavoin tekijöistä riippuen.

Tuotteiden valmiusastetta seurattiin tuotepäätöprosessin avulla. Prosessi edellytti, että tietyt vaiheet ja asiat olivat tehtynä, ennenkuin voitiin siirtyä seuraavaan vaiheeseen. Vetovastuu tästä prosessista oli tuotehallinnalla, jolla kuitenkaan ei ollut näkyvyyttä tiimien tekemiseen. Kun tuotepäätösprosessi perustui tuotehallinnan vetämänä pääsääntöisesti vesiputousmalliin, ja tuotekehitys noudatti ketterän kehityksen erilaisia menetelmiä ja tekniikoita, oli selvää, että prosessi ei voinut toimia. Päätökset olivat muodollisia, vailla todellista painoarvoa. Prosessi sai alkunsa 2013, ja se lopetettiin 2016.

Tuona aikana vain yksi tuote pääsi koko määritellyn elinkaaren läpi. Tuotteita kuitenkin kehitettiin ja myytiin käytännössä jatkuvasti.

Kun yhtenäistä tapaa tehdä kehitystyötä ei ollut, ei vaatimuksia käsitelty eikä muutoshallintaa tehty yhtenäisellä tavalla. Samojen tai samankaltaisten vaatimusten sekä niiden välisten riippuvuuksien hahmottaminen oli vaikeaa. Ongelmana oli myös se, että yhtenäinen ohjaus liiketoimintatasolta asti puuttui. Kukin tuotelinja vastasi omien tuotteidensa kehityksestä, ja tuotteita kehitettiin pääsääntöisesti laajalla rintamalla ilman liiketoimintajohdon priorisointia. Tuotehallinta oli oma saarekkeensa vailla päivittäistä kontaktia kehitystiimeihin, ja sen näkyvyys ja mahdollisuudet vaikuttaa tiimien tekemiseen olivat melko rajoittuneet.

Johtopäätöksenä todettiin, että vaatimusten käsittelyä ja hallintaa oli muutettava. Sen seurauksena laatu tulisi paranemaan, ja tuotteita saataisiin markkinoille nopeassa tahdissa.

Marraskuussa 2015 julkistettin ensimmäinen konkreettinen vaatimushallinnan muutosehdotus. Esityksen jälkeen käynnistettiin vaatimushallinnan muutosprojekti, jonka esimäärittelyvaihe kesti kolme kuukautta (tammi–maaliskuu 2016).

Projekti oli yksi samanaikaisesti käynnistetyistä prosessimuutosprojekteista. Määrittelyjen pohjatyötä varten käynnistettiin sisäinen tiedonkeruukampanja, jonka avulla oli aktivoida työntekijöitä kertomaan, mitkä olemassaolevista prosesseista koettiin toimiviksi. Palautetta varten jaettiin lomake, joka jaettiin yrityksen työntekijöille intranet -sivuilla. Palauttaneille luvattiin palkinto. Kampanja toteutettiin 1.1. - 31.03.2016 välisenä aikana. Palautteita tuli 10 kpl (vastausprosentti ~1%), joista kaksi koski epäsuorasti vaatimushallintaa. Toisessa ehdotettiin Slackin käyttöä kommunikointivälineenä ja toisessa hyväksyntätestivetoista automaattista testausta työkalun avulla.

Projektin kestoksi arvioitiin kalenterivuosi 2016. Ensimmäisen vuosineljänneksen aikana määriteltäisiin vaatimukset, toisen neljänneksen aikana prosessi pilotoitaisiin. Prosessi määritettäisiin tarkalla tasolla kolmannella vuosineljänneksellä, ja samanaikaisesti alkaisivat henkilökunnan koulutukset. Prosessi olisi käytössä koko yrityksen laajuudella vuoden 2016 loppuun mennessä.

Huhtikuussa 2016 projektiryhmä esitteli johtoryhmälle luonnoksen uudesta prosessista. Prosessin osittainen pilotointi alkoi johtoryhmän hyväksyttyä esityksen. Yksityiskohtaisen määrittelyn tuli valmistua elokuun 2016 loppuun mennessä. Se on kuitenkin viivästynyt, ja tätä työtä kirjoitettaessa näyttää siltä, että määrittelytyö valmistuu vuoden 2016 lopulla. Käyttöönotto tapahtuu vuoden 2017 alusta alkaen.

Projektin tavoitteita ja sisältöä kuvattiin yrityksen intranetissä, Wiki -sivuilla, sekä työntekijöille järjestetyissä online-kokouksissa huhti–toukokuussa 2016. Tiedotus oli yleisluontoista, jolloin kuka tahansa saattoi osallistua mihin tahansa tiedotustilaisuuteen. Tiedottajina toimivat projektiryhmään osallistuneet henkilöt. Projektin valmistuttua siitä tullaan tiedottamaan asianosaisille. Mahdollisia tiedotuskanavia ovat Yammer, intranet ja perinteinen sähköposti.

Projektin tavoitteiksi asetettiin ennen kaikkea asiakkailta ja markkinoilta tulevien vaatimusten oikein ymmärtäminen ja jakaminen tiimien kesken sekä tehokkuuden lisääminen siten, että samoja tai samankaltaisia tuotteita ei kehitettäisi eri puolilla organisaatiota, eikä myöskään kustannussyistä jatkokehitettäisi vanhenevia tuotteita. Tuotteita ja ratkaisuja tuli kehittää ketterästi ja nopeasti, ominaisuuksia priorisoida ja sisällyttää tuotteisiin liiketoiminnan tarpeita silmälläpitäen, lisäksi resurssit tuli priorisoida

liiketoiminnan prioriteetteja ja tavoitteita vasten. Tavoitteisiin sisällytettiin myös tehokkaampi muutosvaatimusten hallinta. Lopuksi mainittiin, että tuotteiden ominaisuuksien hallinta ja päätös niiden lisäämisestä/poistamisesta vaatii kaikkien sidosryhmien panosta. Sidosryhmiä ei tässä kohden tarkemmin määritelty.

Ketterän menetelmän periaatteita noudattaen kehitystiimit sijoittuisivat fyysisesti lähelle toisiaan, jolloin tiedonkulku paranisi, ja kommunikointi olisi päivittäistä ja toteutuisi helpommin.

Prosessin sisältöä määritettäessä pyrittiin vastaamaan seuraaviin kysymyksiin

- Miten vaatimuksia tulisi kehittää, seurata, hallita, validoida ja miten niistä tulisi raportoida
- Mitkä ovat vaatimustenhallintaan osallistuvien henkilöiden roolit ja vastuut
- Miten vaatimuksia priorisoidaan, hyväksytään ja ylläpidetään
- Millä perusteilla vaatimuksia ja ratkaisuja hyväksytään
- Miten vaatimuksia jäljitetään
- Miten vaatimuksia kirjataan ja miten niistä kommunikoidaan sidosryhmille

Prosessi näyttääkin hakevan vastauksia kysymyksiin, jotka kuuluvat pääasiassa perinteisen mallin mukaiseen *vaatimusmäärittelyyn*, joka on laajempi kokonaisuus kuin vaatimushallinta. Vaatimushallinta puolestaan on osa vaatimusmäärittelyä. Sen osa-alueista mainitaan tavoitteiden kohdalla muutoshallinta ja kysymysten kohdalla vaatimusten jäljittäminen. Versionhallintaa ja vaatimusten tilojen seurantaa ei erikseen mainita tavoitteissa tai kysymysten asettelussa. Ne näyttävät kuitenkin toteutuvan Product Engineering -prosessin kautta, joka seuraa tiiviisti vaatimushallinnan prosessia ja osin kietoutuu sen ympärille. Product Engineering -prosessi on vaatimushallinnan ohella yksi niistä kaikista uusista prosesseista, jotka on tarkoitus ottaa käyttöön vuoden 2016 aikana. Product Engineering -prosessi tulee ohjaamaan tuotekehityksen toimintaa globaalisti. Siitä kerrotaan myöhemmin tässä luvussa.

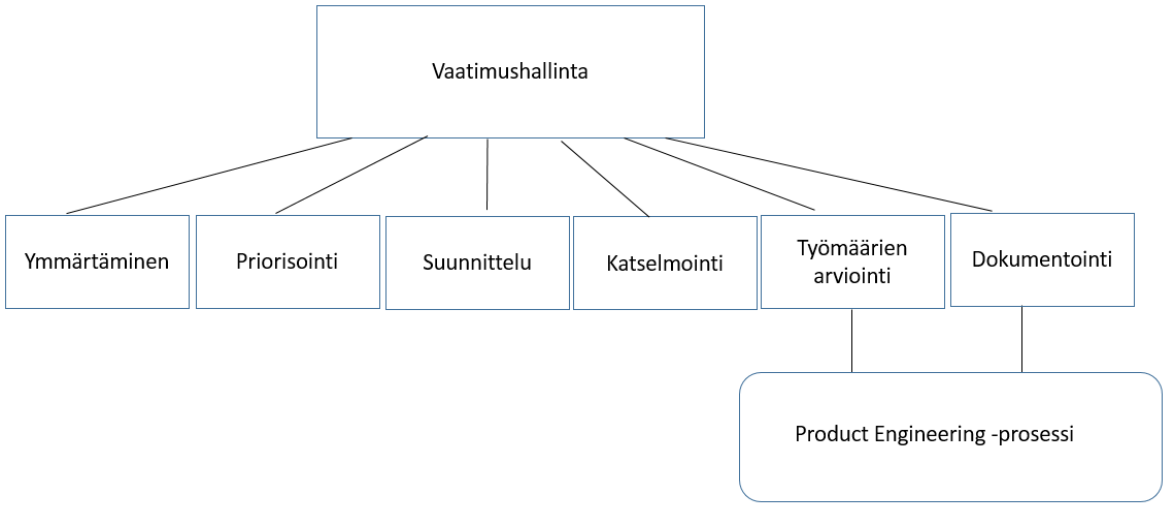
Voidaan todeta, että teoriaperustassa kuvattu vaatimushallinta toteutuu yrityksessä muodollisesti nimetyn vaatimushallinnan prosessin (jatkossa tästä prosessista käytetään nimitystä Toimeksiantajan prosessi = TA-prosessi) sekä tuotekehitystä varten luodun Product Engineering -prosessin kautta. Kokonaisuus kuvaa, miten vaatimusmäärittely, sisältäen vaatimushallinnan, toteutetaan.

Kumpikaan määriteltävistä prosesseista ei ole suoraan verrattavissa teoriaperustassa mainittuihin tuotekehitysmalleihin, joskin ne ovat ottaneet vaikutteita erityisesti ketterästä menetelmästä.

Määrittelytyö on opinnäytetyön kirjoittamisen aikana kesken. Tästä johtuen kappaleen alkuosassa mainitut tavoitteet sekä kysymykset jäävät suurelta osin avoimiksi. Prosesseista voidaan kuitenkin muodostaa käsitys ylätasolla. Seuraavissa kappaleissa käydään läpi niiden määrittely annetun materiaalin ja haastattelujen perusteella.

TA-prosessiin sisältyvät vaatimusten ymmärtäminen, priorisointi, suunnittelu, työmäärien arviointi, katselmointi, dokumentointi ja monitorointi. Teoriaperustan mukaiset vaatimushallinnan osa -alueet toteutuvat Product Engineering -prosessissa, joka puolestaan nojautuu ketterän menetelmän periaatteisiin (Scaled Agile Framework).

Kuvassa 5 esitetään TA-prosessin määrittämät vaiheet.




Kuva 5. TA-prosessin vaiheet

**Ymmärtäminen** on ensimmäinen TA-prosessin vaiheista. Tässä vaiheessa kartoitetaan vaatimuksia keräämällä niitä eri lähteistä – markkinoilta, asiakkaiden tarpeita kartoittamalla, teknologian pohjalta, muutospyyntöjen perusteella jne. Vaatimuksia tuovat esiin organisaation eri edustajat. Edustettuina ovat myynti ja markkinointi, tuotehallinta, toimitus- ja ylläpito-organisaatio, ratkaisukonsultointi sekä ratkaisujen toimitusryhmä. Vaatimuksia selvitetään kuviteltujen käyttäjäpersoonien avulla. Lopputuloksena syntyy tuoteidea/ratkaisu, joka palvelee sidosryhmän tarpeita mahdollisimman hyvin heidän tarpeidensa ja ympäristönsä rajoitustensa mukaisesti.

**Priorisointivaihe** alkaa kartoitettujen vaatimusten kuvaamisella. Ne pitää kuvata joko epiikkeinä (sisältää useita käyttäjätarinoita, ja kuvaa laajempaa toimintoa, esimerkiksi loppukäyttäjälle näkyvissä ja tilattavissa olevat mobiilipalvelut) tai yksittäisinä käyttäjätarinoina (esimerkiksi paljonko on prepaid -saldoa jäljellä) valmista luonnospohjaa käyttäen. Tarinat kirjataan JIRA-järjestelmään. Niiden kirjaamisen yhteydessä on kerrottava tietyt asiat. Näitä asioita ovat mm. itse vaatimuksen kuvaus, luokitus (epiikki, tuotevaatimus, räätälöintivaatimus, muutospyyntö, ei-toiminnallinen vaatimus), raportoija, markkina, käyttäjäpersoona, muut tuotemoduulit, joihin vaatimus liittyy sekä prioriteetti.

Kuvaukset toimitetaan tuotepäällikölle, joka tekee niiden perusteella nopean, 1–2 viikkoa kestävän tuotoskohtaisen analyysin yhdessä ratkaisukonsultoinnin kanssa. Analyysin jälkeen tuotepäällikkö esittelee kuvatun vaatimuksen Change Control Board:ille (CCB) päätöksentekoa varten. CCB päättää, viedäänkö vaatimus eteenpäin edelleen analysoitavaksi, vai hylätäänkö se. Vaatimus voidaan myös pitää odottamassa, kunnes se voidaan sisällyttää seuraavaan tuoteversioon.

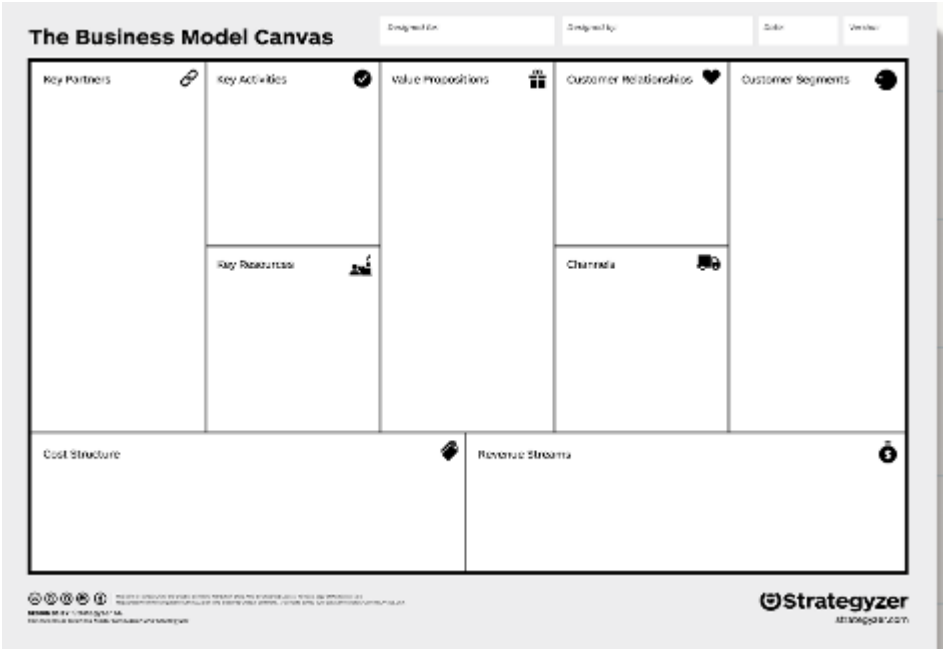
Tässä vaiheessa määritellään MVP (Minimum Viable Product), ja sen sisältämät pakolliset ominaisuudet. Määrittelyn tekee Change Control Board. Päätöstä tehdessään Board tukeutuu strategiaan sekä tarkastelee mm. myynnin ennusteita ja tilauskantaa. CCB:n jäsenet koostuvat seuraavien organisaatioiden edustajista: myynti ja markkinointi, tuotehallinta, tuotekehitys, ratkaisukonsultointi sekä ratkaisutoimitus. Päätöksenä tulisi syntyä nippu priorisoituja ominaisuuksia, joiden perusteella tuotepäällikkö alkaa työstää uutta tuoteversiota (MVP). Tuoteversio voi olla aivan uusi tai olemassaolevan tuotteen päälle tuleva uusi versio. Lisäksi päätetään, mitä dokumentteja tuotteesta tullaan kirjoittamaan.

Dokumentteja kirjoitetaan vain tarpeellinen määrä, ja vain kaikkein tärkeimmät dokumentit. Pääsääntöisesti tuotetaan vain markkinointimateriaalia, teknisiä kuvauksia sekä asennusohjeita.

Change Control Board kokoontuu kerran kahdessa viikossa tai kerran kuukaudessa, tarpeen mukaan.

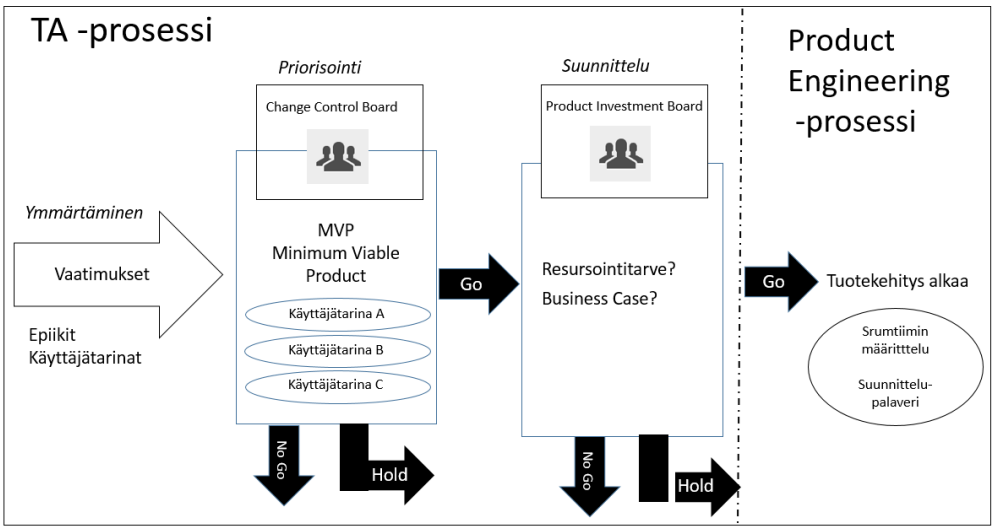
**Suunnitteluvaiheessa** määritellään tuoteversion (MVP) tekemistä varten tarvittavat resurssit sekä valmistellaan tuoteversiokohtainen Business Case. Työmäärien ja edellä mainittujen asioiden perusteella Product Investment Board (PIB) päättää, lähdetäänkö tuotetta kehittämään, hylätäänkö se vai siirretäänkö sen kehittämistä eteenpäin. PIB muodostuu johtoryhmästä ja organisaation ylimmistä johtajista. Päätöstä tehtäessä

tutkitaan, miten tuote sopii yrityksen tuotetarjoomaan ja strategiaan, millainen kilpailutilanne tuotteen kohdalla on, millaista kysyntää sille on, miten tuotekehitysresurssit on mitoitettu, millaista investointia tarvitaan jne. Apuna voidaan käyttää liiketoimintamallilakanaa (Business Model Canvas). Kuvassa 6 näytetään prosessissa esimerkkinä annettu Strategyzer AG:n liiketoimintamallilakana, The Business Model Canvas.




Kuva 6. Business Model Canvas

Mikäli suunnitteluvaiheen tuloksena PIB päätyy siihen, että tuote on kehittämisen arvoinen, se siirtyy tuotekehitykseen (kuva 7 alla).




Kuva 7. TA-prosessista tuotekehitykseen

Tuotekehitystä ohjaa *Product Engineering* -prosessi. Prosessi perustuu ketterään menetelmään (Scrum). Jokaisen tuotteen kehitys alkaa scrumtiimin määrittelyllä. Sitä seuraa tuotteen suunnittelupalaveri. Työ toteutetaan sprinteissä, joiden kestoaika on 14 kalenteripäivää. Työn edistymistä seurataan päivittäisissä scrum -palavereissa ja harvemmin toistuvissa Scrum of Scrum -palavereissa. Scrum of Scrum -palaverien tarkoitus on myös nähdä töiden mahdollinen päällekkäisyys. Jäljellä olevan työn määrää tutkitaan sprinttikohtaisten edistymiskäyrien avulla, joista nähdään, miten paljon työtä on jäljellä verrattuna arvioidun työn määrään. Työ käyttäjätarinan osalta katsotaan tehdyksi, kun koodi on tarkistettu (parityönä tai DevOps -työkalua käyttäen) ja tarinakohtainen testi on hyväksyttävästi suoritettu.

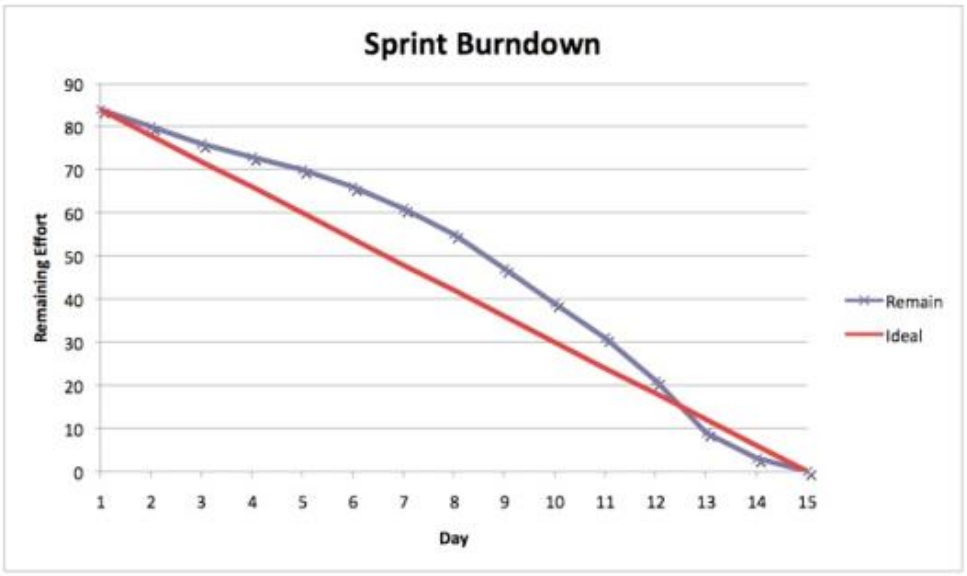
Scrumtiimin koko on 7 – 9 henkilöä. Se koostuu tuoteomistajasta, scrummasterista, kehittäjistä ja DevOps-insinööristä. Lisäksi määrätään arkkitehti. *Tuoteomistajan* tehtävä on edustaa asiakasta kehitysryhmän suuntaan, tulkita käyttäjätarinoita, osallistua päivittäisiin scrum -palavereihin ja vastata kaikkiin kehitysryhmän kysymyksiin kaikkina aikoina. *Scrummaster* puolestaan vartioi tiimin toimintaa, pitää huolta laatuprosessien seurannasta, ja eliminoi mahdollisia haittatekijöitä, jotka voisivat vähentää tiimin tuottavuutta. *Kehittäjien* tehtävä on tuottaa kehitysjonon sisällöstä julkaisukelpoinen, ”valmis” tuote sprintin lopputuotoksena. *DevOps-insinööri* vastaa siitä, että kehittäjien ja järjestelmäasiantuntijoiden vuoropuhelu toimii, ja että käytettävät työkalut ovat tarkoituksenmukaisia. *Arkkitehti* voi olla jaettu resurssi eri scrumtiimien kesken.

Suunnittelupalaveriin osallistuvat scrumtiimin lisäksi tuotepäällikkö ja laadunvalvontatiimin edustaja. Kokouksessa valitaan kehitysjonosta tiimin priorisoimat tehtävät eli luodaan sprintin kehitysjono. Työmäärät arvioidaan Planning Poker -työkalun avulla. Kunkin sprintin pituus on 14 kalenteripäivää. Lopuksi suunnitellaan sprintin tavoite ja aikataulutus. Suunnittelun tulokset välitetään ohjelmiston integrointitiimille, tuotepäällikölle, toimitusyksikölle ja kehityksen johdolle.

Seurantaa suoritetaan päivittäisten, lyhyiden (maksimissaan 15 min) Scrum -palaverien avulla. Palavereissa käydään läpi, mitä on tapahtunut viimeisimmän palaverin jälkeen, mitä on nyt suunnitelmissa ja mitä esteitä toteutukselle mahdollisesti on.

Päivittäisten Scrum-palavereiden lisäksi pidetään Scrum of Scrum -palavereita. Niiden tarkoitus on estää päällekkäisyyksiä tuotekehityksessä, ja parantaa tuotteiden integrointia. Palavereihin valitaan Scrum-tiimeistä edustajat. Palaverien kestoaika on pidempi kuin Scrum-palavereissa, ja niitä pidetään Scrum-palavereita harvemmin.

Sprinttejä seurataan edistymiskäyrien avulla. Niistä nähdään, paljonko on toteutusaikaa jäljellä verrattuna ideaalitilanteeseen.




Kuva 8. Sprintin edistymiskäyrä

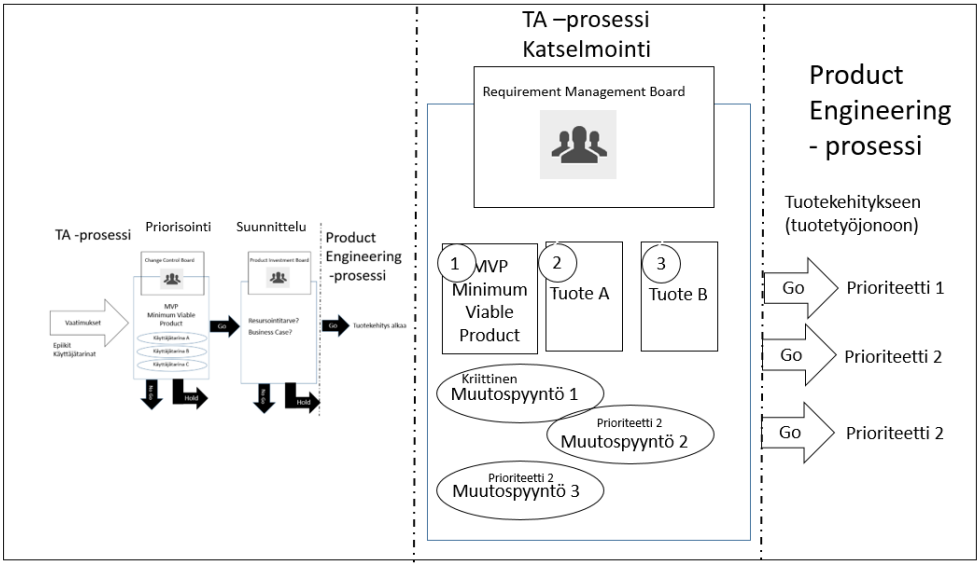
Sprinttien tulokset esitetään demoina. Esitykset ovat epämuodollisia. Esityksen tekoon saa käyttää maksimissaan kaksi tuntia aikaa, eivätkä ne saa olla powerpoint –muodossa. Demojen esittelyyn osallistuvat scrumtiimi, tuotepäällikkö/ratkaisukonsultointi sekä laatupäällikkö. Retrospektiivi tehdään jokaisen tuoteversion valmistumisen jälkeen, ja se sisältää seuraavat kysymykset:

- Mitä teimme hyvin
- Mitä opimme
- Mitä voisimme tehdä paremmin
- Mikä jäi vaivaamaan

Tulokset dokumentoidaan ja talletetaan osaksi kehitysjonoa.

Tuotetta kehitettäessä vastaanotetaan jatkuvasti muutospyyntöjä. Niitä käsitellään TA-prosessin **katselmointivaiheessa**, jossa kohteena ovat sekä varsinaiset vaatimukset että muutospyyntövaatimukset. Katselmointia suorittaa vaatimushallintatiimi (Requirement Management Team), johon kuuluu edustajia ratkaisukonsultoinnista, markkinoinnista, tuotehallinnasta, liiketoimintayksiköistä, tuotekehityksestä (kehittäjät, testaajat) sekä ratkaisutoimituksesta. Tässä vaiheessa MVP:tä tarkastellaan vasten olemassaolevia tuoteprojekteja, ja sen kehitystyö priorisoidaan uudelleen. Tiimi kokoontuu tarpeen mukaan, jopa kerran päivässä.

Kuvassa 9 hahmotetaan TA-prosessiin kuuluvaa katselmointivaihetta. MVP ja muut kehitettävänä olevat tuotteet priorisoidaan uudelleen. Kuvassa on viitteellisesti valittu MVP tärkeimmäksi kolmen tuotteen joukosta (MVP, Tuote A ja Tuote B). Sen jälkeen katsotaan, mitä muutospyyntöjä on saatu ja priorisoidaan ne. MVP:hen voidaan sisällyttää kriittiisiksi katsotut muutospyynnöt. Vähemmän kriittiset muutospyynnöt siirretään toteutettavaksi myöhemmin. Tuotteita voidaan kehittää rinnakkain, ja niiden prioriteetit voivat olla samanarvoisia. Kuvasta näkyy, että tuotekehityksen kehitysjonoon on siirretty kaksi samalla prioriteetilla arvioitua tuotetta. Tarkemmin sanoen tehtävät, jotka on johdettu käyttäjätarinoina esitetyistä vaatimuksista tai epiikeistä, on siirretty tuotekehitykseen. Muutosvaatimusten priorisoinnista sekä tuotteiden uudelleen priorisoinnista vastaa Requirement Management Board. Jäsenet koostuvat ratkaisukonsultoinnin, liiketoimintayksiköiden, tuotekehityksen ja tuotehallinnan edustajista.




Kuva 9. TA-prosessin katselmointivaihe

Katselmointia seuraa **työmäärien arviointi**. Se sisältyy Product Engineering -prosessiin. Tuoteomistaja pilkkoo käyttäjätarinat tehtäviksi, ja samalla listaa ne tehtävät, joilla hän katsoo olevan riippuvuuksia muihin vaatimuksiin. Lopuksi tuoteomistaja määrittelee työmäärät story pointteina.

**Dokumentoinnilla** tässä yhteydessä tarkoitetaan vaatimusten kuvausta. Vaatimukset ja niihin liittyvät muutospyynnöt kuvataan epiikkeinä ja käyttäjätarinoina BDD (Behavioural Driven Development) -tekniikkaa käyttäen, ja talletetaan kehitysjonoihin. Kehitysjonoja on kaksi, toinen geneerisiä tuotteita varten (Product Backlog) ja toinen räätälöityjä tuotteita (Custom Backlog) varten. Muutospyyntövaatimuksia analysoitaessa vaatimuksiin otetaan mukaan ei-toiminnallisten vaatimusten rajoitukset. Tuotehallinta/tuoteratkaisukonsultointi

on vastuullinen selvittämään nämä rajoitukset, joita ovat mm. muutospyyntövaatimuksen konfigurointimahdollisuudet, räätälöintimahdollisuudet, skaalautuvuus, suorituskyky, luotettavuus ja turvallisuus. Prosessissa ei ole otettu kantaa siihen, miten ei-toiminnalliset vaatimukset kuvataan.

Taulukko 2. Vaatimushallinnan osa -alueiden toteutuminen TA-prosessissa ja Product Engineering -prosessissa

| **Vaatimushallinnan osa -alueet** | **TA -prosessi, Product Engineering –prosessi** |
|---|---|
| Muutoshallinta | Toteutuu TA-prosessin katselmointivaiheessa.<br>Saadut muutosvaatimukset priorisoidaan, ja tuotteiden kehitys uudelleenpriorisoidaan, jolloin tehtäville annetaan uusi tärkeysjärjestys niiden siirtyessä kehitysjonoihin. |
| Konfiguraation- ja versionhallinta | Prosesseista ei käy ilmi, miten konfiguraatio- tai versionhallinta toteutuu. |
| Vaatimusten tilojen seuranta | Toteutuu Product Engineering -prosessissa<br>Käyttäjätarinoista pilkotut tehtävät talletetaan kehtiysjonoihin (geneerisille ja räätälöidylle tuotteille omat kehitysjonot), joissa niiden tiloja seurataan. Prosessi ei vielä kuvaa, mitä tilamerkintöjä käytetään (aiemmin on yleisimmin käytetty tiloja Open, Rejected ja Closed). |
| Vaatimusten jäljittäminen | Toteutuu osittain TA-prosessissa sekä Product Engineering –prosessissa.<br>Sekä käyttäjätarinat että niihin liittyvät tehtävät talletetaan kehitysjonoihin. Prosesseissa ei ole kuitenkaan kerrottu, miten ne linkitetään toisiinsa, joten jäljitystä ei voida todentaa. |

Prosessien toimivuutta alettiin kokeilla keväästä 2016 alkaen. Käyttäjätarinoita alettiin kirjoittaa BDD-tekniikalla jo kehityksen alla olevien tuotteiden uusille tuotemoduleille. Kokeilun avulla testattiin myös, miten TA-prosessissa kuvattu päätöksentekomekanismi käytännössä toimi.

Tarkoituksena oli kokeilemalla löytää sekä korjata mahdollisia puutteita ennen prosessien virallista käyttöönottoa. Onnistuneessa lopputuloksessa malli on omaksuttu kaikkialla tuotekehityksessä ja -hallinnossa yhtenäisellä tavalla, jolloin organisaatio toimii tehokkaimmalla mahdollisella tavalla.

# 8 Johtopäätöksiä

Ensimmäisenä tutkimuskysymyksenä etsittiin prosessista eroja ja yhtäläisyyksiä perinteiseen ja ketterään mallin verrattuna. Toimeksiantajan prosessia, tarkemmin sanottuna *prosesseja*, tarkasteltaessa näyttää siltä, että molemmista oli saatu vaikutteita. Tuotepäätöksiä tehtäessä pyritään päätösmekanismiin, jossa päätökset tehdään huolellisen suunnittelun ja analyysin sijasta tiettyjen organisaatioiden päätöksestä, liiketoiminnan tarpeisiin perustuen. Esikuvana toimii perinteinen malli, mutta päätökset tehdään enemmänkin ketterän menetelmän mukaisesti kevyemmällä prosessilla. Samoin perinteisen mallin mukainen määrittelyvaihe toteutuu prosessin ensimmäisessä vaiheessa, jota kutsutaan *ymmärtämiseksi*. Siinä kartoitetaan vaatimuksia, jotka kirjoitetaan käyttäjätarinoiden muotoon. Vaatimusten keruun osalta noudatetaankin perinteistä mallia. Kuitenkaan perinteisen mallin mukaiset vaatimukset esimerkiksi arkkitehtuurista tai algoritmeista eivät ole käyttäjätarinoissa oleellista, tärkeämpää on kirjoittaa tarinat asiakasnäkökulmasta käsin. Edelleen, prosessi jatkuu käyttäjätarinoiden priorisoimisella. Tärkeimmistä muodostuu MVP, Minimum Viable Product, jonka kehittämispäätös perustuu liiketoimintapäätökseen. Myönteinen päätös aloittaa tuotekehityksen, jossa noudatetaan ketterää menetelmää (Scrum). Mielestäni prosessi ei ole puhtaasti perinteisen eikä ketterän mallin edustaja. Kyse on enemmänkin hybridimallista, joskin siinä on vaikutteita erityisesti ketterästä menetelmästä.

Toinen tutkimuskysymyksistä käsittelee ongelmia, joihin prosessi pyrkii vastaamaan. Suureksi ongelmaksi koettiin se, että vaatimuksia ei osattu tulkita oikein alusta alkaen. Selvää tuntui olevan se, että vaatimushallintaa tuli muuttaa asiakaslähtöisempään suuntaan. Asiakkaan ymmärtäminen asetettiin ensisijaiseksi tavoitteeksi. Käyttäjätarinoilla koettiin olevan suuri merkitys asiakkaan ymmärtämisessä. Tarinat tuli kirjoittaa BDD (Behavioural Driven Development) -tekniikkaa käyttäen. Ne kuvaavat varsin hyvin loppukäyttäjän tarvetta, mutta hyvin suppeasti. Entä jos vaatimus tulisikin ymmärtää laajemmin? Miten tarkistetaan, että käyttäjätarina todella vastaa useamman asiakkaan tarvetta ja toiveita? Entä muutokset – miten ne otetaan käyttäjätarinoissa huomioon? Miten voidaan seurata yksittäistä tehtävää käyttäjätarinan tasolle asti? Vaatimukset tuli kirjata järjestelmään, joka antaa mahdollisuuden jäljitykseen esim. projektien tai tuotemodulien kautta. Tämä kannattanee ottaa huomioon, kun prosesseja määritellään tarkemmalla tasolla.

Ongelmia kartoitettaessa olisi ehkä ollut hyödyllistä myös nähdä, mitkä syyt johtivat siihen, että esimerkiksi tuoteversioita ei saatu ajoissa julkaistua. Oliko kyse siitä, että alkuperäisiä vaatimuksia ei ymmärretty oikein vai kenties siitä, että muutos- tai versionhallintaa ei

suoritettu järjestelmällisesti? Vaiko siitä, että kehitystiimien ollessa pirstaloituneina eri maihin, työ ei edennyt niin tehokkaasti kuin se muutoin olisi voinut edetä?

On kuitenkin positiivista, että prosessi vaatii että sekä toiminnalliset että ei-toiminnalliset vaatimukset dokumentoidaan kehitysjonoon. Tätä ei yleisesti vaadita ketterissä menetelmissä.

Kolmas ja samalla viimeinen tutkimuskysymys käsitteli vaatimushallinnan toteutumista prosessissa. Pääsääntöisesti voidaan todeta, että vaatimushallinta kyllä näyttäisi toteutuvan, mutta sitä ei ole määritetty eikä kuvattu riittävällä tarkkuudella. Mielestäni pitäisi pyrkiä määrittämään prosessi loppuun siten, että vaatimushallinnan eri osa-alueisiin on kiinnitetty huomiota niiden ansaitsemalla tavalla. On selvää, että esimerkiksi vaatimusten seuranta ja jäljittäminen ovat tärkeässä asemassa, kun tuotteita jatkokehitetään.

Kuvatut prosessit kattoivat sekä vaatimusmäärittelyn että siihen sisältyvän vaatimus-hallinnan. Ensisijaisesti pyrittiin kuvaamaan sitä tapaa, millä vaatimusmäärittely ja -hallinta suoritetaan. Tärkeää olisi ollut kertoa myös, mihin prosessin eri vaiheilla tähdätään, ja miten suoriutumista, etenkin laadullista suoriutumista, mitataan. Laadun mittaamisessa hyvänä työkaluna on esimerkiksi ISO-standardin listaamat ominaisuudet vaatimukselle. Prosessien kuvaamisessa voisi lisäksi mielestäni käyttää selkeitä, yksinkertaisia malleja ja pyrkiä esimerkkien valossa kertomaan, mistä kussakin prosessissa on kysymys, ja miten kukin prosessi nivoutuu seuraavaan prosessiin. Ketterän mallin mukainen toiminta vaatii sen, että tiimit toimivat paikallisesti. Tähän varmaankin tähdätään, ja se yksin jo tuo toimintaan tehokkuutta, jota muilla keinoin on vaikea saavuttaa.

Lopuksi haluaisin sanoa, että vaatimushallinnan uudistaminen on vaativa projekti, etenkin, kun se tapahtuu eri kulttuureissa ja erilaisen toimintatavan omaavissa tuotekehitysryhmissä. Muutosta suunniteltaessa on tunnistettava ne ongelmat, joihin ratkaisua haetaan, ja ne syyt, mistä ongelmat johtuvat. Muutos ei tapahdu hetkessä, ja se vaatii jatkuvaa sitoutumista ja vuoropuhelua, ei ainoastaan tekijöiltä, vaan erityisesti johdolta ja varsinkin liiketoimintajohdolta. Ajan kuluessa nähdään, miten prosessimuutos lopulta muotoutui ja onnistui.

Koska prosessia ei ole määritelty loppuun asti, eikä sen tuloksia voida siis analysoida, olisi tarpeen suorittaa analysointi myöhemmässä vaiheessa. Miten hyvin vaatimusten ymmärtämisessä ja kuvaamisessa sekä muutoshallinnassa sekä jäljittämisessä onnistuttiin? Miten versionhallinta toteutettiin? Millaisia laadullisia mittereita prosessissa asetettiin ja miten niissä suoriutumista mitattiin?

Opinnäytetyön kirjoittamisen aikana vaatimushallinnan merkitys itselleni konkretisoitui. Opin, miten tärkeää vaatimusten ymmärtäminen on, asiakkaan ja liiketoiminnan kannalta, ja millä eri tekniikoilla vaatimuksia voidaan kartoittaa.

Lähteet

Laamswerde, A. (2009). Engineering Requirements for System Reliability and Security. Université catholique de Louvain B-1348 Louvain-la-Neuve, Belgium.

Royce, W. (1970). Managing the development of large software system: Concepts and techniques. Teoksessa *Proceedings of the 9th International Conference on Software Engineering*. (s. 1–9). IEEE Computer Society.

Sommerville, I. (2004). Software Engineering, 7th edition. Chapter 7.

Strategyzer *Business Model Canvas* Haettu 19.8.2016 osoitteesta http://businessmodelgeneration.com/canvas/bmc?_ga=1.45496100.1892707729.1469534596