Mika Tyynismaa

Lean- ja Kanban-menetelmät ohjelmistotuotannossa

Opinnäytetyö

Kevät 2014

Tekniikan yksikkö

Teknologiaosaamisen johtamisen koulutusohjelma

Seinäjoen ammattikorkeakoulu
SEINÄJOKI UNIVERSITY OF APPLIED SCIENCES

SEINÄJOEN AMMATTIKORKEAKOULU

Opinnäytetyön tiivistelmä

Koulutusyksikkö: Tekniikan yksikkö

Koulutusohjelma: Teknologiaosaamisen johtamisen koulutusohjelma

Tekijä: Mika Tyynismaa

Työn nimi: Lean- ja Kanban-menetelmät ohjelmistotuotannossa

Ohjaaja: Petteri Mäkelä

Vuosi: 2014 Sivumäärä: 96 Liitteiden lukumäärä: -

Ohjelmistotuotanto on nuori tieteenala verrattuna muuhun teolliseen tuotantoon. Erilaisia menetelmiä ja tekniikoita on kehitelty runsaasti, mutta ohjelmistotuotannon käytännöt eivät ole vakiintuneet. Kehitetyt menetelmät ovat yleensä liittyneet muihin asioihin kuin itse tuotantoprosessiin, jonka hallintaan ja tehtävien virtaukseen liittyvät ongelmat ovat jääneet vähemmälle huomiolle.

Perinteisten ohjelmistomenetelmien lisäksi, ohjelmistotuotanto on yli 10 vuotta hakenut uutta suuntaa ketterien ohjelmistomenetelmien muodossa. Ketteryys ei kuitenkaan ole toiminut eikä skaalautunut hyvin kokonaisvaltaisen liiketoiminnan ja yrityksen pitkän tähtäimen tavoitteiden kanssa.

Nykyään ohjelmistotuotannon ongelmiin on etsitty apua Lean-ajattelusta, joka on saanut alkunsa japanilaisesta autoteollisuudesta. Lean-menetelmien avulla ohjelmistoprosessia pyritään tehostamaan poistamalla siitä sellainen toiminta (hukka), joka ei asiakkaan näkökulmasta lisää arvoa. Prosessin virtauksen hallitsemiseksi ja hukan välttämiseksi työkaluna käytetään Kanban-mentelmää. Lean- ja Kanban-menetelmien on katsottu kuuluvan ketterien menetelmien toiseen sukupolveen. Tutkimusten mukaan toimialan suuntaus on kohti Lean-menetelmien ja ketterien menetelmien yhdistämistä.

Tämän työn ensisijainen tavoite on tutustua Lean- ja Kanban-menetelmiin ohjelmistotuotannon näkökulmasta. Toinen tavoite on kehittää tutkimuksen kohdeorganisaation toimintaa edellä mainittujen menetelmien perusteella.

Avainsanat: Lean, Lean-ajattelu, Lean-ohjelmistokehitys, Kanban, Kanbanin periaatteet

SEINÄJOKI UNIVERSITY OF APPLIED SCIENCES

Thesis abstract

Faculty: School of Technology

Degree programme: Master's Degree in Technology Competence Management

Author: Mika Tyynismaa

Title of thesis: Lean and Kanban methods in the software development

Supervisor: Petteri Mäkelä

Year: 2014 Number of pages: 96 Number of appendices: -

The software engineering is a young discipline compared to the other industrial manufacturing sciences. A number of different methods and techniques have been developed, but the software engineering practices are not established well. The developed methods are usually related to the other areas than actual software development process. The process management and flow-related problems have received less attention.

In addition to the traditional software development methods, the software engineering has searched to a new direction in the form of agile software methodologies (more than 10 years). But the agile software development practices have not worked well and not scaled well either to the overall business and the long term objectives of the company.

The software engineering tries to prevail the problems found in today by bringing the Lean thinking into the software development process. Lean thinking has its origins in the Japanese automotive industy. The Lean methods are designed to improve the software process by removing such activites (waste) that do not increase the value from the customer's point of view. The Kanban method is used as a tool to manage the flow and process. The Lean and Kanban metods are considered to be the second generation of the agile methods. The software industry trend is towards combining the Lean thinking and agile software development.

The primary objective of this thesis is to explore the Lean and Kanban methods from the software engineering's point of view. The second objective of this thesis is to develop the organization practices according to the above mentioned methods.

Keywords: Lean, Lean thinking, Lean software development, Kanban, principles of Kanban

SISÄLTÖ

Opinnäytetyön tiivistelmä 2
Thesis abstract 3
SISÄLTÖ 4
Kuvio- ja taulukkoluettelo 6
Käytetyt termit ja lyhenteet 7
1 JOHDANTO 9
1.1 Työn tausta 9
1.2 Työn tavoite ja rajaus 10
1.3 Työn rakenne 11
1.4 Työssä käytetyt menetelmät 11
2 LEAN 13
2.1 Leanin historia 13
2.2 Toyota 14
2.2.1 Toyotan tuotantojärjestelmä (TPS) 15
2.2.2 Toyotan tapa (The Toyota Way) 17
2.2.3 Toyotan tapa Lean-johtamiseen 19
2.3 Lean 21
2.3.1 Lean-ajattelun viisi periaatetta 22
2.3.2 Lean-tuotanto 23
2.3.3 Lean-tuotekehitys 24
2.3.4 Arvo, arvovirta ja hukka 26
2.4 Yhteenveto Leanista 33
3 LEAN-OHJELMISTOKEHITYS 35
3.1 Ketterä ohjelmistokehitys 35
3.2 Lean-ohjelmistokehitys 37
3.2.1 Lean-ohjelmistotuotannon tulkintoja 39
3.2.2 Lean-ohjelmistokehitykseen siirtyminen 41
3.2.3 Lean-menetelmät laajentavat ketteriä menetelmiä 44
3.3 Pohdintaa Lean- ja Agile-ohjelmistokehityksestä 45
4 KANBAN 48

4.1 Kanban-menetelmän edut 48
4.2 Kanban-menetelmän periaatteet ohjelmistotuotannossa 50
4.2.1 Tee työnkulku näkyväksi 51
4.2.2 Rajoita työn määrää eri vaiheissa 54
4.2.3 Mittaa ja tarkkaile työn etenemistä 56
4.2.4 Tee prosessikäytännöistä havainnollisia 59
4.2.5 Käytä valmiita malleja tunnistamaan uusia kehitysmahdollisuuksia 59
4.2.6 Prosessin jatkuva kehittäminen 59
4.3 Kanban ja ohjelmistotuotannon syklit 61
4.4 Pohdintaa Kanban-mentelmästä 63
5 KOHDEORGANISAATION NYKYINEN TOIMINTAMALLI 66
5.1 Toimintaympäristön esittely 66
5.2 Toimintatavat pääpiirteissään 67
5.2.1 Esimerkki palvelupyynnön käsittelystä 71
5.2.2 Esimerkki tuotekehityspyynnön käsittelystä 73
5.3 Tuotekehityksen mittarit 74
5.4 Kohdeorganisaation lähtökohdat Lean-menetelmille 74
6 TOIMINNALLINEN KEHITTÄMINEN 77
6.1 Lean- ja Kanban-menetelmät kohdeorganisaatiossa 77
6.2 Toiminnan kehittäminen kohti Lean-ohjelmistokehitystä 78
6.2.1 Lean-periaatteiden ymmärtäminen 79
6.2.2 Asiakkaan kokeman arvon määrittäminen 81
6.2.3 Prosessin arvontuottokyvyn mittaaminen 82
6.2.4 Arvovirran kuvaaminen 83
6.2.5 Kanban-menetelmä ja prosessin virtauksen hallinta 85
6.3 Lean kohdeorganisaatiossa 87
7 YHTEENVETO JA POHDINTAA 90
LÄHTEET 94

Kuvio- ja taulukkoluettelo

Kuvio 1. TPS-talokaavio 15
Kuvio 2. Kaizen-prosessi 16
Kuvio 3. Asiakas ensin -johtamisfilosofia 18
Kuvio 4. Lean-johtajien kehittämisen timantti 20
Kuvio 5. Lean-ajattelun periaatteet 22
Kuvio 6. Lean-hierarkiapuu 24
Kuvio 7. Yhtäläisyyksiä Lean-tuotannon ja tehokkaan tuotekehityksen välillä 25
Kuvio 8. Kano-malli ja Kano-taulukko 26
Kuvio 9. Tietämyshukan kategoriat 29
Kuvio 10. Tehtävien priorisointi CD3-arvon perusteella 31
Kuvio 11. Ominaisuuden liiketaloudellisen arvon laskeminen ajan kuluessa 32
Kuvio 12. Soveltamisalue Agile vs. Lean 36
Kuvio 13. Kanban-taulu 52
Kuvio 14. Rajoittamaton WIP (töiden määrä eri vaiheissa) 53
Kuvio 15. Läpimenoaika vs. vaiheen läpimenoaika 53
Kuvio 16. Arvovirran tehokkuus 54
Kuvio 17. Kanban-taulu ja WIP-arvot 55
Kuvio 18. Työn keskimääräinen läpimenoaika 57
Kuvio 19. Kumulatiivinen virtauskaavio 58
Kuvio 20. Ohjelmistovirheiden lukumäärä 58
Kuvio 21. Ohjelmistotuotannon syklit 62
Kuvio 22. Toiminnan kattavuus maailmalla 66
Kuvio 23. Tuotemarkkinointiosaston tehtävät 68
Kuvio 24. Kohdeorganisaation työympäristö ja työkalut 70
Kuvio 25. Palvelupyynnön tai ongelmatilanteen vastaanotto 72
Kuvio 26. Tuotekehityspyynnön käsittely 73
Kuvio 27. Kano-malli 81
Kuvio 28. Työtyypin keskimääräinen läpimenoaika 82
Kuvio 29. Kanban-taulu ja arvovirta 86

Käytetyt termit ja lyhenteet

Agile	Nimitys ohjelmistotuotannossa käytettäville kevyemmille menetelmistöille.
Andon	Tuotantoalueella sijaitseva visuaalinen merkinantojärjestelmä.
Arvo (value)	Asiakkaan silmin nähtävä lisäarvo.
ERP	Yrityksen kokonaisvaltainen toiminnanohjausjärjestelmä (Enterprise Resource Planning)
Genchi gembutsu	Genchi gembutsu tarkoittaa ongelman, tilanteen tai asian täydellistä selvittämistä ja ymmärtämistä paikan päällä.
Hukka (waste)	Kaikki sellainen toiminta, joka ei asiakkaan näkökulmasta lisää arvoa tuotteeseen tai palveluun.
Just-in-time	Juuri oikeaan aikaan
Kaizen	Kaizen on japania ja tarkoittaa jatkuvaa pienin askelin tapahtuvaa kehittymistä ja parantamista.
Kanban	Kanban on Lean-periaatteiden mukainen imuohjaukseen perustuva tuotannonajoitusjärjestelmä.
Ketterät menetelmät	Nimitys ohjelmistokehityksen lyhyisiin iteraatioihin perustuville työskentelytavoille ja käytännöille.
Lean	Johtamisfilosofia, joka keskittyy tuotannossa esiintyvän turhan työn poistamiseen ja prosessin virtaviivaistamiseen.
Muda	Muda tarkoittaa tuotannossa lisäarvoa tuottamatonta toimintaa.

Mura	Mura tarkoittaa vaihtelua ja epätasaisuutta prosessissa tai tuotannossa.
Muri	Muri tarkoittaa ihmisen tai koneen ylikuormitusta jonkin ulkopuoliset asian vuoksi.
Scrum	Yleisin ketterän ohjelmistokehityksen menetelmä.
Standardoitu työ	Standardoitu työ tarkoittaa vakaata selvästi määriteltyä tuotantoprosessia tai työvaihetta.
Vesiputosmalli	Vaiheittain etenevä perinteinen ohjelmistokehitysmalli.
WIP	Työn määrän rajoittaminen (work-in-progress). Kertoo kuinka paljon työtä prosessissa tai vaiheessa saa olla enimmillään.

1 JOHDANTO

1.1 Työn tausta

Ohjelmistotuotanto on nuori tieteenala verrattuna muuhun teolliseen tuotantoon. Erilaisia menetelmiä ja tekniikoita on kehitelty runsaasti, mutta ohjelmistotuotannon käytännöt eivät ole vakiintuneet. Kehitetyt menetelmät ovat yleensä liittyneet uusiin ohjelmointikieliin tai tapoihin määritellä ja kuvata asioita. Ohjelmistotuotannon ongelmat kuitenkin liittyvät yleensä itse tuotantoprosessiin oli siinä käytettävät sisäiset menetelmät kuinka hyviä tahansa. Perinteiseen valmistavaan tuotantoon verrattuna ohjelmistotuotannon tuotteet ovat luonnostaan monimutkaisia, tietyssä mielessä näkymättömiä, helposti muunneltavia, ohjelmistot ovat ainutkertaisia, huonosti skaalautuvia projektin koon kasvaessa ja virhetilanteissa toiminta on epävarmaa. (Haikala & Märijärvi, 2004, 23 - 31.)

Perinteisten ohjelmistomenetelmien (esim. vesiputousmalli) lisäksi, ohjelmistotuotanto on yli kymmenen vuotta hakenut uutta suuntaa ketterien ohjelmistomenetelmien muodossa. Ketterät menetelmät perustuvat vuonna 2001 luotuun Agile-manifestiin painottaen neljää seuraavaa arvoa: yksilöitä ja kanssakäymistä, toimivaa sovellusta, asiakasyhteistyötä ja vastaamista muutokseen. Työ tehdään tiimeissä ja pienissä iteraatioissa yhteistyössä asiakkaan kanssa. Ketteryys ei kuitenkaan ole toiminut kokonaisvaltaisen liiketoiminnan ja yrityksen pitkän tähtäimen tavoitteiden kanssa. Yleensä myynti ja tuotehallinto eivät toiminnassaan ketteryyttä käytä. Toimintaa ei tuotekehityksen tavoin johdeta priorisoidun tehtävälistan kautta. Ketterät menetelmät eivät myöskään helposti skaalaudu yhden tiimin projektista usean tiimin moniprojektiympäristöksi. (Agile Manifesto Org 2001: Aalto-Yliopisto 2012.)

Nykyään ohjelmistotuotannon ongelmiin on haettu apua japanilaisesta autoteollisuudesta peräisin olevalla Lean-ajattelulla. Lean-ajatteluun perustuva tuotannonohjaus valmistavassa tuotannossa ja sen tuotekehityksessä ovat todistaneet toimivuutensa. Virtaviivaisen Lean-ajattelun avulla tuote virtaa arvonlisäysprosessien läpi keskeytyksettä, ja samalla prosessista pyritään poistamaan ylimääräisiä turhia toimintoja. Teollisuustuotannosta peräisin oleva Lean-tuotekehitys on koettu

monitulkintaiseksi ohjelmistotuotannossa. Kuitenkin Lean-käsitteitä on alettu soveltaa myös ohjelmistotuotannossa. Lean-ajattelua soveltamalla ohjelmistotuotannosta on saatu järkevämpää, mikä on johtanut laadun ja prosessin paranemiseen. Lean-periaatteet voivat oikein käytettynä auttaa koko ohjelmistotuotannon hallintaan liittyvissä kysymyksissä.

Lean-ajattelun avulla prosessia pyritään tehostamaan poistamalla siitä sellainen toiminta (hukka), joka ei asiakkaan näkökulmasta lisää arvoa. Prosessin virtauksen hallitsemiseen ja hukan välttämiseen tarpeellinen työkalu on Kanban-menetelmä. Tutkimusten perusteella Kanban-menetelmä tarjoaa vähän vastustusta aiheuttavan lähestymistavan muuttaa nykyistä prosessia. Menetelmä ei sisällä erillisiä roolituksia, ja sen on todettu olevan hyvä tapa tuoda Lean-ajattelua organisaatioon, muokata työkulttuuria ja rohkaista jatkuvan parantamisen ajattelutapaan. Kanbanin on todettu sopivan ketteriä menetelmiä käyttäville tiimeille sekä myös enemmän perinteistä lähestymistapaa käyttäville tiimeille. (Liker 2006, 6: Anderson, 2010, 14 - 15.)

Lean- ja Kanban-menetelmien on katsottu kuuluvan ketterien menetelmien toiseen sukupolveen. Toimialan suuntaus on kohti Lean-menetelmien ja ketterien menetelmien yhdistämistä. (Objectives 2012a: Ikonen 2011, 34: Oivo 2014: Rodriguez 2013, 96 - 98.)

1.2 Työn tavoite ja rajaus

Opinnäytetyön aiheena on Lean- ja Kanban-menetelmät ohjelmistotuotannossa. Aihe on hyvinkin ajankohtainen, sillä Lean-ajattelun nähdään tuovan tarvittavaa tehokkuutta nykyiseen ohjelmistotuotantoon. Etenkin Kanban-tyyppisten menetelmien osuuden nousu erilaisissa tutkimuksissa herätti mielenkiinnon kyseisiä menetelmiä kohtaan.

Tämän työn ensisijainen tavoite on tutustua Lean- ja Kanban-menetelmiin ohjelmistotuotannon näkökulmasta. Tutkimuksen tarkoituksena on auttaa ymmärtämään, mitä on Lean-ajattelu, Lean-ohjelmistotuotanto ja Kanban-menetelmä. Toinen tavoite on miettiä tutkimuksen kohdeorganisaation kehittämistä teoriapohjalta

edellä mainittujen menetelmien perusteella. Tutkimuksen kohdeorganisaation toimintamalli perustuu perinteiseen vesiputousmalliin, mutta toimintaa on alettu hitaasti muuttaa kohti ketteriä menetelmiä.

Tutkimuksesta on rajattu pois ketterien menetelmien ja perinteisen vesiputousmallin tarkempi esittely. Tukimuksessa oletetaan, että lukija tietää mitä edellä mainitut ohjelmistokehitysmenetelmät ovat. Teoriaosuudessa Lean ja Kanban käsitellään ohjelmistotuotannon näkökulmasta eli perinteisen valmistavan tuotannon näkökulma on rajattu osittain pois. Empiirisessä osuudessa toiminnan kehittäminen teoriapohjalta keskittyy vain kohdeorganisaatioon.

1.3 Työn rakenne

Tutkimuksen rakenne on jäsennetty seitsemään lukuun. Johdanto-osuudessa kuvataan tutkimustyön tausta. Luvussa 2 esitellään Leanin historia, Toyotan filosofia ja periaatteet, Lean-ajattelun periaatteet, Lean-tuotanto ja Lean-tuotekehitys. Lisäksi luvussa kuvataan tärkeät käsitteet: arvo, arvovirta ja hukka. Luvussa 3 kuvataan Lean-ohjelmistokehitys ja sen eri tulkintoja. Luvussa 4 esitetään Kanban-menetelmä ja sen periaatteet. Luvussa 5 kuvataan tutkimuksen kohdeorganisaatio ja sen toimintatavat pääpiirteissään. Luvussa 6 esitetään kohdeorganisaation toiminnallinen kehittäminen teoriapohjalta. Luvussa 7 esitetään tutkimuksen yhteenveto ja pohdintaa.

1.4 Työssä käytetyt menetelmät

Tämä opinnäytetyö on tyypiltään tapaustutkimus. Tapaustutkimuksen kohteena on Lean- ja Kanban-menetelmät ohjelmistotuotannossa. Tutkimuskohteena on myös tutkimuksen tekijän oma organisaatio ja sen ohjelmistotuotannon kehittäminen edellä mainittujen menetelmien pohjalta. Työ suoritettiin kvalitatiivisena eli laadullisena tutkimuksena, jotta ymmärrettäisiin tutkimukseen liittyvää viitekehystä: Toyotan valmistamisen filosofia, Lean-ajattelu, Kanban-menetelmä, perinteinen ohjelmistotuotanto ja ketterä ohjelmistotuotanto.

Tapaustutkimuksen aineistohankinnan lähtökohtana oli tutkimustehtävä. Tutkimuksen primaariaineistona on käytetty välitöntä tietoa tutkimuskohteena olevista Lean- ja Kanban-menetelmistä. Tutkimuksen teoriaosuus aloitettiin tutustumalla Mary ja Tom Poppendieckin (2003 – 2013) kirjoittamaan Lean-ohjelmistotuotannon kirjasarjaan, missä teollisen tuotannon Lean-ajattelun periaatteet on muunnettu ohjelmistotuotantoon. Tutkimustyö eteni Toyotan Lean-periaatteisiin tutustumiseen John Stewartin (2012), Jeffrey K. Likerin (2006) ja Likerin ja Convisin (2012) teosten kautta. Lean-tuotekehityksen taustalla oleviin periaatteisiin ja teoriaan käytettiin Donald G. Reinertsenin (2009) ja Allen C. Wardin (2007) kirjoja, jotka antoivat hyvät pohjatiedot myös Kanban-menetelmälle. Kanban-osuuden primaariaineistona käytettiin David J. Andersonin (2010) kirjaa.

Tutkimuksen sekundaariaineistona käytettiin muiden tekemiä empiirisiä tutkimuksia. Tälläistä tutkimusaineistoa löytyi väitöskirjojen muodossa sekä Lean-menetelmästä että Kanban-menetelmästä. Sekundaariaineisto laajensi tutkimusta kohti Lean-menetelmien ja ketterien menetelmien yhdistämistä.

2 LEAN

2.1 Leanin historia

Lean-ajattelutapa ja filosofia ovat peräisin Japanista, autonvalmistaja Toyotan kehittämästä tavasta valmistaa ja kehittää autoja. Vuonna 1945 pieni autonvalmistaja Toyota ryhtyi kehittämään silloisen johtajansa Kiichiro Toyodan myötä teknologioita alkavalle autoteollisuudelle. Mallia haettiin Amerikan autoteollisuudesta. Kiichiro oli aikaisemmin viettänyt vuosia Detroitissa hakemassa oppia autoteollisuuden tuotantotavoista. Alusta alkaen Amerikan tulosten saavuttamisessa oli kuitenkin ongelma, resurssipulan vallitessa autonvalmistaja Toyota ei voinut käyttää amerikkalaista massatuotantomallia. Massatuotanto oli halvin tapa tehdä autoja, mutta se vaati tuhansien samanlaisten osien ja ajoneuvojen valmistusta. Toisen maailmansodan jälkeisen Japanin markkinat eivät siihen aikaan olleet riittävän suuret kyseiselle tuotantotavalle. Materiaaleista oli pulaa ja ajoneuvotilaukset olivat taloudellisesta tilanteesta johtuen vaihtelevia. Talouden elpyminen ja Japanin uudelleen rakentaminen kuitenkin edellytti kuorma-autoja, joten Toyota sai tilauksia. Ongelmana oli rahan arvottomaksi tehnyt inflaatio ja maksun saaminen asiakkailta. Massatuotantojärjestelmästä poiketen Toyotan piti valmistaa pieniä määriä erilaisia malleja ja tilaukset vastaanotettiin maksua vastaan. Toyotan omaksui valmistusprosessin, jolla oli lyhyet läpimenoajat, joustavat tuotantolinjat ja ennen kaikkea korkea laatu. (Poppendieck & Poppendieck 2003, 2: Poppendieck & Poppendieck 2007, 4: Liker 2006, 18 - 21.)

Kiichiro Toyodalla oli luonut vision tuotantolinjasta jossa kaikki tarvittavat osat saapuisivat juuri oikeaan aikaan linjalle. Kiichiro loi täten pohjantyön *juuri oikeaan aikaa* -tuotannolle (*Just-in-time, JIT)*. Visio toteutui kuitenkin vasta vuonna 1962, noin kymmenen vuotta Kiichiro Toyodan kuoleman jälkeen, kun Taiichi Ohno vastasi Toyodan visioon ja haasteeseen kehittämällä tuotantojärjestelmää. Aikaisemmin vuonna 1956 Ohno oli matkustanut opintomatkalle Yhdysvaltoihin, vieraillut autotehtaissa ja tutustunut Fordin tuotantojärjestelmään. Tärkein löytö opintomatkalla Ohnolle oli amerikkalaistyylinen supermarket. Kaupasta asiakas valitsi juuri sen mitä halusi, sellaisen määrän kuin halusi ja vielä milloin halusi. Kaupan tapa

toimittaa tuotteita oli yksinkertainen, tehokas ja oikea-aikainen. Tämä löytö loi pohjan imuohjaukselle ja Kanban-prosessityökalun synnylle. Ohnon vuosien kehitystyön pohjalta syntyi Toyotan tuotantojärjestelmän (*Toyota Production System, TPS*), jonka yhtiö otti käyttöönsä maailmanlaajuisesti. Järjestelmän periaatteena on kaiken hukan (waste) poistamien tuotannosta, eli kaikki mikä ei tuota arvoa asiakkaalle on hukkaa. (Poppendieck & Poppendieck 2007, 4 - 5: Toyota Motor Manufacturing Kentucky 2012.)

Hukan muodot on jaettu kolmeen eri pääkategoriaan, joiden japaninkieliset termit ovat: muda, muri ja mura. Muda on yleisimmin käytetty ja tarkoittaa tuotannossa lisäarvoa tuottamatonta toimintaa. Muri tarkoittaa ihmisen tai koneen ylikuormitusta jonkin ulkopuoliset asian vuoksi. Mura tarkoittaa vaihtelua ja epätasaisuutta prosessissa tai tuotannossa. (Stewart 2012, 88 - 93: Liker 2006, 114.)

Taiichi Ohnon apuna Toyotan tuotantojärjestelmää kehittämässä ollut konsultti Shigeo Shingo on listannut seitsemän tuotannossa esiintyvää, lisäarvoa tuottamatonta toimintaa ja hukan muotoa (muda). Luetellut toiminnot auttavat hukan tunnistamisessa tuotannosta. Nämä seitsemän hukan muotoa ovat: varastointi, ylikäsittely, ylituotanto, kuljetus, odottaminen, liike ja virheet. (Poppendieck & Poppendieck 2003, 4.)

Näitä valmistavan tuotannon hukan muotoja tarkastellaan vielä myöhemmin tässä tutkimuksessa ohjelmistokehityksen näkökulmasta.

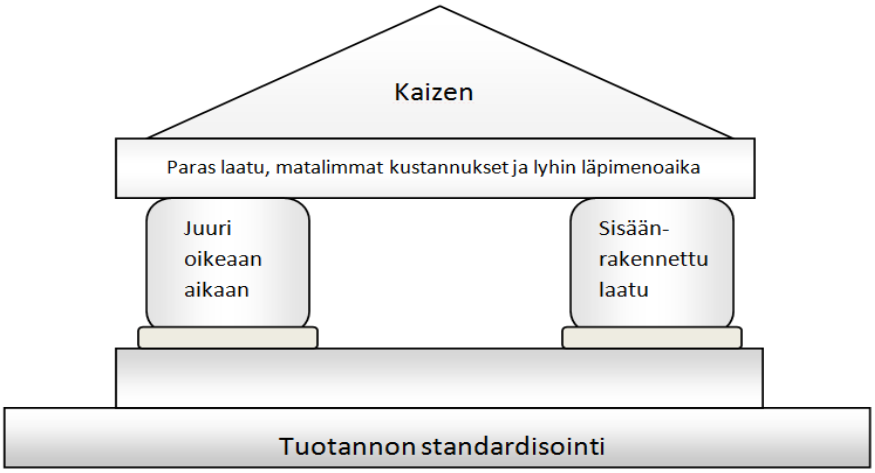
2.2 Toyota

Toyotan valmistamisen filosofia on esimerkki mitä Lean-tuotannolla ja Lean-johtamisella saadaan aikaan. Vuoteen 2008 asti yhtiö oli ollut yhtäjaksoisesti tuottoisa 70 vuotta (Helsingin Sanomat 2008) ja noussut yhdeksi maailman suurimmaksi moottoriajoneuvojen valmistajaksi. Yhtiön toimintatavasta on tullut mittapuu ja opas monille muille toimijoille.

2.2.1 Toyotan tuotantojärjestelmä (TPS)

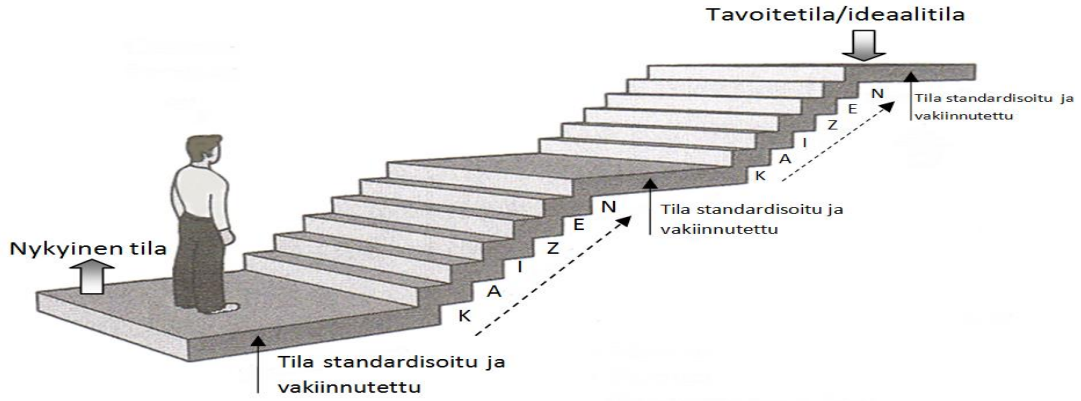
Nykyään Toyotan tuotantojärjestelmä on usein kuvattu TPS-talokaaviona, joka rakentuu Ohnon määritelmän mukaan kahden pilarin varaan: *juuri oikeaan aikaan* ja *sisäänrakennettu laatu* (jidoka). *Juuri oikeaan aikaan* -pilari sisältää periaatteita, menetelmiä ja työkaluja, joiden avulla tuotantoprosessissa on käsillä vain oikea määrä tuotteita. *Sisäänrakennettu laatu* -pilari sisältää tavat, joilla laatuongelmat tuodaan esille ja selvitetään. TPS-talokaavioita on useita erilaisia, mutta kaikki ne rakentuvat samojen periaatteiden varaan. (Stewart 2012, 26 - 29.)

John Stewart (2012) on kirjassaan *The Toyota Kaizen Continuum: A Practical Guide to Implementing Lean* kuvannut yksinkertaisen TPS-talokaavion kuvion 1 mukaisesti.



Kuvio 1. TPS-talokaavio
(Stewart 2012, 27.)

Kuviossa 1 ovat ne elementit, joihin Toyotan tuotantojärjestelmä perustuu. Kaiken perustana on standardisointi, joka sisältää prosessit, tuotteet, työkalut ja työvaiheet. Standardisoinnin päällä lepää kaksi pilaria: *Juuri oikeaan aikaan* ja *Sisäänrakennettu laatu*. Talon kattona on kaizen. Kaizen on japania ja tarkoittaa jatkuvaa pienin askelin tapahtuvaa kehittymistä ja parantamista. Tuotannon standardisointi tarvitaan, koska se takaa vakaan ja toistettavan prosessin, jossa asiat tehdään aina samalla. Tämä stabiilisuus mahdollistaa toimintotapojen jatkuvan parantamisen pienin askelin kohti tuotannon ideaalitilaa (kuvio 2). Talon kattona oleva Kaizen on siis prosessi, joka toistetaan aina kun edellinen muutos on saatu standardisoitua ja vakiinnutettua. (Stewart 2012, 26 - 29.)



Kuvio 2. Kaizen-prosessi
(Stewart 2012, 29.)

Vaikein osuus Kaizen-prosessissa ei välttämättä ole muutos, vaan muutoksen vakiinnuttaminen. Standardisointi edesauttaa TPS-talon katosta löytyvin tavoitteiden (paras laatu, matalimmat kustannukset ja lyhin läpimenoaika) saavuttamista vahvistamalla kahta talon pilaria. (Stewart 2012, 31.)

Ilman jatkuvaa parantamista järjestelmän tuottama arvo ei toteudu, ja siksi jatkuva parantaminen onkin velvollisuus. Toyotalla tätä prosessia mitataankin sanomalla ”*Älä koskaan mittaa missä sinä olit, vaan sitä missä sinun pitäisi olla*”. Tällä mitataan tuotannon tavoitetilaa ja nykyisen tilan eroa. (Stewart 2012, 52.)

Toyotan tuotantojärjestelmässä *Juuri oikeaan aikaan* ja *Sisäänrakennettu laatu* ovat kaksi pääperiaatetta kaikessa, mitä Toyota tekee. Työkalujen (JIT, Value stream map, Kanban, Andon, Genchi Gembutsu, 5-whys yms.) tarkoitus on tukea näitä kahta edellä mainittua periaatetta. Tuotteita ja niiden vaatimuksia on erilaisia, joten oikea tapa tuotantoon on yrityksen etsittävä itse. Todellisia kysymyksiä ovatkin: Mikä on paras tapa varmistaa *Juuri oikeaan aikaan* tuotanto? Mitkä ovat ne toimenpiteet joilla varmistetaan laatu? Avaintekijä Toyotan tuotantojärjestelmässä on ymmärtää ja hallita organisaation tavoitteita. Täytyy ymmärtää, että ei ole olemassa muutoksen tekevää hopealuotia, vaan muutos tulee pienin edistysaskelin, jotka auttavat yrityksen johtoa laatimaan ja asettamaan asianmukaisia tavoitteita. Toyotalla näitä pieniä parannusehdotuksia tulee vuosittain tuhansia, ja niiden laatimisesta ja toteuttamisesta ovat vastuussa itse työntekijät omilla vastuualueillaan. (Stewart 2012, 67 - 68.)

Toyotan tuotantojärjestelmässä nämä kaksi pilaria *Juuri oikeaan aikaan* ja *Sisään-rakennettu laatu* ovat pystyviä, mutta järjestelmä ja työkalut niiden ympärillä ovat joustavia. Osa työkaluista on käytössä ja osa ei ole käytössä lainkaan. Osaa työkaluista voidaan käyttää juuri kuin oppikirjoissa kerrotaan, ja muita työkaluja voidaan muuttaa organisaation käyttöön sopivaksi. (Stewart 2012, 68 - 69.)

2.2.2 Toyotan tapa (The Toyota Way)

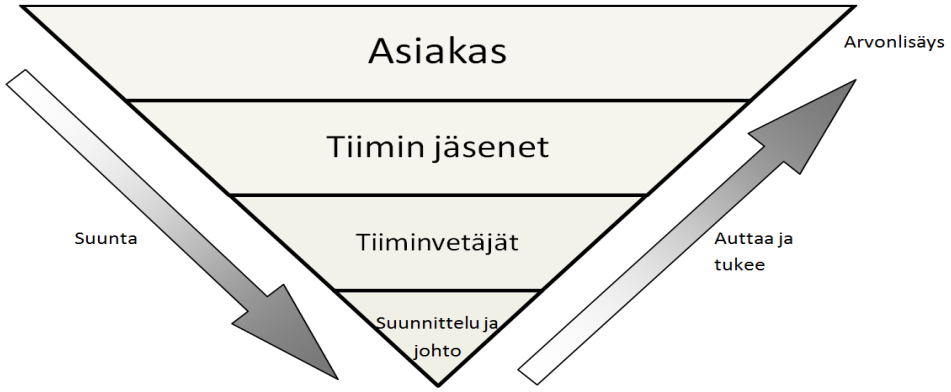
Toyotalla kaikkea toimintaa ohjaa *Toyotan tapa* -filosofia (*The Toyota Way*), joka perustuu kahteen perusarvoon: ihmisten kunnioittamiseen ja jatkuvaan parantamiseen (*Kaizen*). *Toyotan tapa* ei ole järjestelmä, prosessi tai ohjelma, se on ajattelutapa joka ohjaa kuinka olla vuorovaikutuksessa toisten kanssa ja kuinka toimintaa johdetaan. (Stewart 2012, 82.)

Toyotan tapa on vuonna 2001 toimitusjohtaja Fujio Chon käynnistämä konsepti. *Toyotan tapa* oli aluksi yhtiön sisäinen, vuonna 2001 julkaisema dokumentti koulutustarkoituksiin. Dokumentti sisälsi viisi Toyotan toimintatavan määrittävää ydinarvoa: haasteisiin tarttumisen henki, Kaizen, Genchi Genbutsu, tiimityö ja kunnioitus. Aikaisemmin näitä arvoja ei ole tarvinnut kirjata ylös, sillä ne ovat automaattisesti periytyneet japanilaisesta kulttuurista ja uskonnosta. Arvot kirjoitettiin kuitenkin ylös, koska Toyotan toiminta oli laajentunut muihin kulttuureihin ja ympäri maailmaa. Tarvittiin konsepti, joka määrittelee kuinka Toyotan tulee toimia yrityksenä maailmanlaajuisesti. (Poppendieck & Poppendieck 2010, 153: Liker & Convis 2012, 30 - 31.)

Toyotan tuotantojärjestelmä (TPS) ei ole sama kuin *Toyotan tapa*. TPS on Toyotan valmistamisen filosofia ja esimerkki siitä, mitä *Toyotan tavan* periaatteet voivat saada aikaan. *Toyotan tapa* sisältää Toyotan perusperiaatteet, joiden ansiosta TPS toimii niin tehokkaasti. Vaikka TPS ja *Toyotan tapa* ovat eri asioita, TPS:n kehitys on vaikuttanut myös *Toyotan tavan* kehitykseen. (Liker 2006, 27.)

Toyotan entinen toimitusjohtaja Kasuaki Watanabe on kertonut, että heille *Toyotan Tapa* määrittelee tavan kuinka toimia. Perusarvoista ensimmäinen, eli *Ihmisten kunnioittaminen* käsittää työntekijät, toimittajat ja asiakkaat. Myös tärkeällä *Asia-*

kas ensin -periaatteella (kuvio 3) ei tarkoiteta ainoastaan ulkoista loppuasiakasta, vaan tuotantolinjalla oleva seuraava henkilö on myös asiakas (sisäinen asiakas). Tämän periaatteen omaksuminen johtaa tiimityöhön missä omaa toimintaa analysoidaan ja tarkkaillaan, sillä omaa asiakasta ei haluta häiritä. Tällainen toiminta kasvattaa työntekijän kykyä tunnistaa ja korjata ongelmia, mikä johtaakin periaatteista toiseen eli *jatkuvaan parantamiseen*. *Toyotan tapa* -ajattelutavan perusta onkin olla tyytymätön nykyiseen olotilaan ja etsiä ikuisesti uusia kehittämiskohteita. Tämä konsepti on Toyotalla käytössä maailmanlaajuisesti. (Poppendieck & Poppendieck 2010, 195.)



Kuvio 3. Asiakas ensin -johtamisfilosofia
(Stewart 2012, 18.)

TPS-järjestelmää sovellettaessa valmistusprosessin tutkiminen aloitetaan ensin asiakkaan näkökulmasta. Tämä toimintatapa määrittää asiakkaalle tuotettavan arvon. Kun prosessia havainnoidaan asiakkaan silmin, voidaan erottaa lisäarvoa tuovat vaiheet lisäarvoa tuottamattomista. Valmistusprosessin lisäksi tätä näkökulmaa voidaan soveltaa myös muihin prosesseihin kuten palveluprosessiin. (Liker 2006, 27.)

Toyotan tapa kuvailee 14 periaatetta, jotka ovat myös Toyotan tuotantojärjestelmän perusta. Periaatteet on jaettu neljään periaateluokkaan:

1. Filosofia (pitkän tähtäimen filosofia)
2. Prosessi (hukan eliminointi)
3. Ihmiset ja yhteistyökumppanit (kunnioita, haasta ja kasvata heitä)
4. Ongelmanratkaisu (jatkuva parantaminen ja oppiminen). (Liker 2006, 13.)

Likerin (2006, 13) tutkimusten mukaan useimmat Lean-yritykset puuhastelevat kuitenkin periaateluokassa kaksi eli prosessitasolla. Ei omaksuta jatkuvan parantamisen ja ihmisten kunnioittamisen kulttuuria. Tämän myötä hukan poistaminen tuotantoprosessista jää helposti johdon vastuulle ilman työntekijöiden parannusehdotuksia. Ihmisten kunnioitus on avain jatkuvaan parantamiseen. (Liker 2006, 6,13: Denning 2010, 190 - 191.)

2.2.3 Toyotan tapa Lean-johtamiseen

Toyotalla Lean-tuotannon taustalla on aina ollut myös Lean-johtaminen. Toyotan johtoa ohjaa *Toyotan tapa* -filosofia ja yrityksen määrittelemä pitkän tähtäimen suunnitelma.

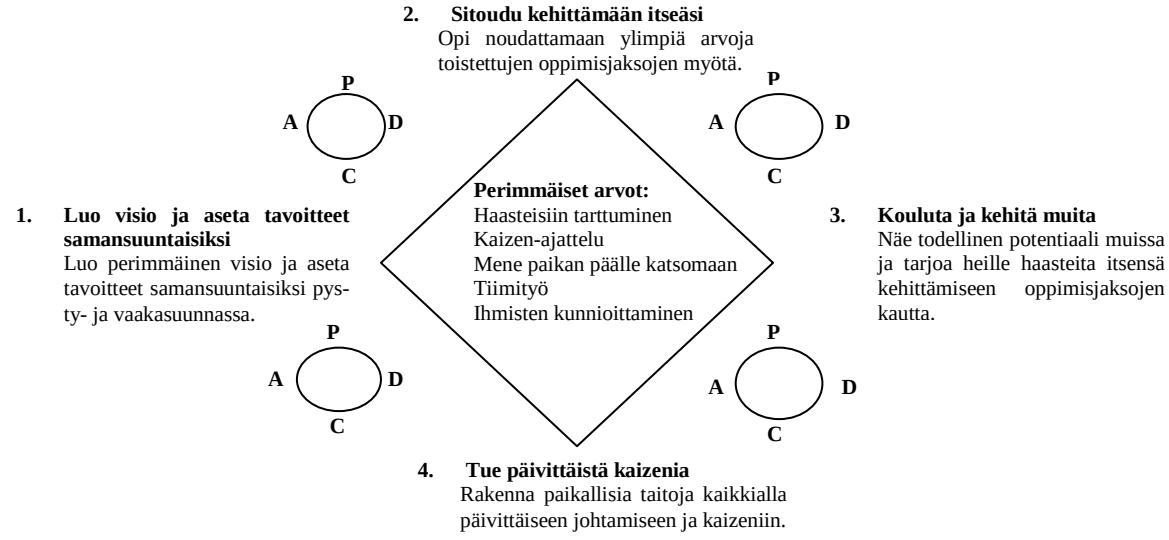
Monilla eri teollisuuden aloilla erilaiset Lean-ohjelmat tuottavat aluksi parannuksia, mutta ajan kuluessa prosessit rapistuvat ja toimita alkaa heiketä. Likerin ja Convisin (2012, 8) mukaan Leania noudatettavilta yrityksiltä puuttuu osaaminen Leania ylläpitävästä kulttuurista ja sen johtamisesta. Kulttuurin muuttaminen on huomattavasti vaikeampaa kuin eri toimintojen välisen arvovirran parantaminen. (Liker & Convis 2012, 8: Liker 2006, 7, 13.)

Tutkimusten mukaan organisaation kulttuuri heijastaa sen johtamista, ja korreloi vahvasti yrityksen tulosta ja markkina-arvoa. Kun yrityskulttuuri on kunnossa, yritys tuottaa taloudellista tulosta. Hyvä johtaminen edellyttää organisaation kulttuurin ymmärtämistä ja sen hallitsemista. Syy japanilaisten yritysten kilpailukykyyn on aasialainen kulttuuri ja filosofia (yhteiskunnan palvelemistehtävä, ihmisten kunnioittaminen, täydellisyyden tavoittelu, omien tehtävien huolellinen arviointi ja parantaminen ja heikkouksien etsintä). Toyota on kehittänyt Lean-johtajia yli 60 vuoden ajan luoden oman johtamismallin, missä johtajat pyrkivät parantamaan jatkuvasti toimintaansa jokaisella liiketoiminnan osa-alueella. Parannuksen saavuttamiseksi jokainen ylimmästä johdosta pienten työryhmien esimiehiin työskentelee yhdessä. Tämä johtamismalli edellyttää kuitenkin pitkän tähtäimen johtamista hierarkian jokaisella tasolla. Jatkuvan parantamisen periaate johtamisessa näkyy myös siinä, että Toyota on onnistunut menestyksekkäästi laajentamaan toimintaansa Yhdysvaltoihin, vaikka maiden väliset kulttuurierot ovatkin huomattavia. Toyotan johta-

mismallia ei voi suoraan kopioida toisen organisaation käyttöön käyttöön. (Huuhka 2006, 110 - 113: Liker & Convis 2012, 7, 27, 38.)

Toyotan toimintaa ohjaava jatkuvan parantamisen periaate on aiheuttanut myös sen, että vaikutteita ja ideoita on otettu runsaasti länsimaista. Länsimaista on lainattu seuraavia asioita: Henry Fordin tuotannon virtauksen ja hukan eliminoinnin käsitteet, amerikkalaisen laatupioneeri W. Edwards Demingin PDCA-ongelmanratkaisumalli, Yhdysvaltain armeijan työnohjauskoulutusmalli. Lainattuja asioita ja ideoita ei suoraan kopioitu, vaan ideoita testattiin, parannettiin, yhdisteltiin ja sovitettiin Toyotan järjestelmään ja kulttuuriin. (Liker 2006, 22 - 23: Liker & Convis 2012, 38 - 39.)

Vuonna 2012 julkaistussa kirjassa *Toyotan tapa Lean-johtamiseen*, Liker ja Convis (2012, 34) ovat kuvanneet mallin Lean-johtajien kehittämiseen (kuvio 4). Perustana ovat Toyotan tavan mukaiset ydinarvot: haasteisiin tarttumisen henki, Kaizen, Genchi Genbutsu, tiimityö ja ihmisten kunnioitus. Kehittämisen neljä eri vaihetta etenevät syklisesti toistuen useaan kertaan. Jokainen vaihe sisältää lukuisia PDCA-kehityssyklejä (plan-do-check-act, suunnittele, tee tarkista, toimi/korjaa). Kehityssyklin alussa on tärkeää ymmärtää nykytilanne ja tavoiteltavan ihannetilanteen välinen kuilu. (Liker & Convis 2012, 31 - 34.)



Kuvio 4. Lean-johtajien kehittämisen timantti
(Liker & Convis 2012, 34.)

Toyotan tapa kehittää Lean-johtajia pohjautuu perimmäiseen päämäärään, vision selkeään ymmärtämiseen ja pyrkimykseen täydellisten prosessien tekemiseen. Oikea prosessi ja hyvin määritellyt tavoitteet johtavat oikeisiin tuloksiin, kun seuraavat: prosessi ymmärretään syvällisesti työtä suorittavalla tasolla, prosesseja ja ihmisiä on valmisteltu ja koulutettu huolellisesti pitkää aikaväliä varten, ongelmien juurisyitä ymmärretään syvällisesti ennen kuin toimitaan. Ihmisiä kehitetään toimimaan tehokkaasti tiimeissä, jotka ongelmanratkaisutyökalujen hallinnan myötä osaavat ratkaista ongelmia tehokkaasti. Lisäarvoa tuottavan työn tekevät ihmiset halutaan ottaa mukaan parantamaan prosessia. Mittareita ei ensisijaisesti käytetä ihmisten valvontaan ja kontrollointiin, vaan mittarit ovat tavoitteita ja työkaluja, joilla tiimit ja työntekijät voivat arvioida omaa suorituskykyään. Tällä tavoin tiimit ja työntekijät asetetaan tavoitteiden osalta samansuuntaisiksi. Mittarit tukevat laatu-, kustannus- ja toimitustavoitteiden päivittäistä hallintaa. (Liker 2006, 22 - 23: Liker & Convis 2012, 203 - 205.)

Toyotan koko yhtiön laajuisen toiminnan koordinointiprosessia, tavoitteiden asettamista ja tavoitteiden konkreettista toteutussuunnitelmaa nimitetään *Hoshin kanri* -prosessiksi. Tämä prosessi lähtee liikkeelle pitkän tähtäimen visiosta, jonka toteutussuunnitelmat virtaavat ylhäältä alaspäin ja tarkentuen organisaation eri tasojen halki. *Hoshin kanri* etenee syklisesti PDCA-mallin mukaisesti ja näitä syklejä on ympäri vuoden. Organisaatiotasojen edistymistä mitataan ja tulokset ohjataan ylemmälle tasolle. Jatkotoimenpiteet päätetään todellisten olosuhteiden ja tavoitteiden välisen kuilun pohjalta. (Liker & Convis 2012, 130, 135 - 136.)

2.3 Lean

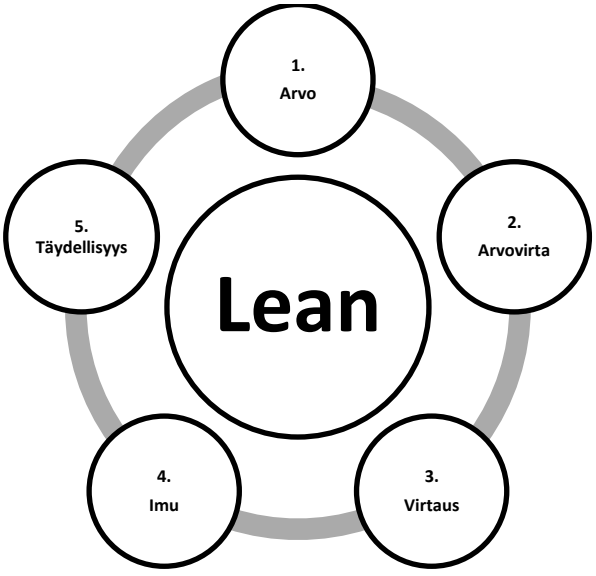
Toyotan tuotantojärjestelmän tehokkuus paljastui vuoden 1973 öljykriisin jälkeen, koska Toyota selvisi kriisistä muita autonvalmistajia nopeammin. Toyotan tuotantojärjestelmää oli opiskeltu myös muissa japanilaisissa yrityksissä, jotka saapuivat kansainvälisille markkinoille laadukkailla tuotteilla ja todella alhaisilla hinnoilla. 1980-luvun alussa Amerikka ja Eurooppa olivat saaneet Japanista todellisen kilpailijan. Kiinnostus japanilaisten ja Toyotan toimintatavoista oli herännyt. (Poppendieck & Poppendieck 2007, 7.)

2.3.1 Lean-ajattelun viisi periaatetta

Vuonna 1990 James P. Womackin kirja *The Machine That Changed the World* esitteli termin “Lean”. Vuonna 1996 James Womack ja Daniel T. Jones jalostivat Lean-ajattelun viisi periaatetta seuraavasti:

1. Arvon määrittäminen asiakkaan näkemänä arvona
2. Arvovirran määrittäminen ja arvoa tuottamattomien toimintojen poistaminen
3. Arvovirran luominen eli mahdollista prosessin jatkuva virtaus askel kerrallaan
4. Asiakasvetoisen prosessin eli imuohjauksen käyttö siellä missä jatkuva virtaus on mahdollista
5. Jatkuva parantaminen ja täydellisyyden tavoittelu (Kaizen). (Lean Enterprise Institute 2009.)

Viisivaiheinen itseään toistava ajatteluprosessi (kuvio 5) on helppo muistaa, mutta ei aina helppo saavuttaa.



Kuvio 5. Lean-ajattelun periaatteet

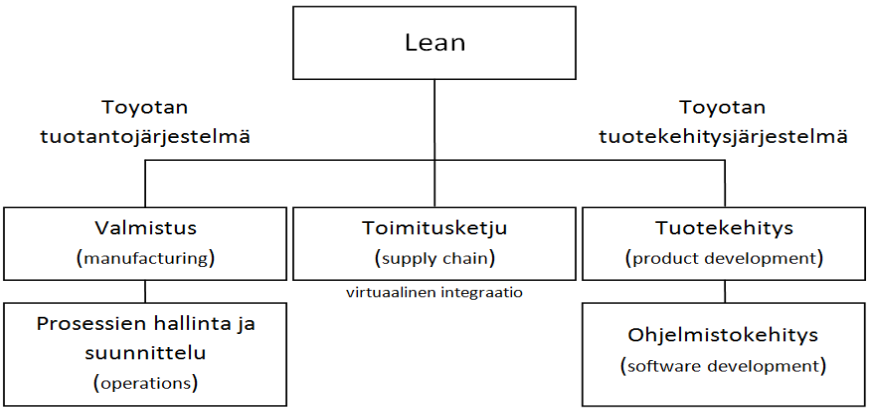
2.3.2 Lean-tuotanto

Lean-termin ilmestymisen jälkeen Toyotan valmistustapaa alettiin kutsua Lean-tuotannoksi. Ollakseen Lean valmistajalta vaaditaan ajattelutapaa, joka keskittyy siihen, että tuote virtaa arvonlisäysprosessien läpi keskeytyksettä. Lisäksi vaaditaan kulttuuria, jossa jokainen pyrkii jatkuvaan parantamiseen. Taiichii Ohno, TPS:n perustaja, tiivisti asian ytimekkäästi: "*Me katsomme ainoastaan aikajanaa siitä hetkestä, kun asiakas antaa meille tilauksen, siihen pisteeseen, kun keräämme rahat. Ja me pienennämme tuota aikajanaa poistamalla lisäarvoa tuottamattoman hukan*". (Liker 2006, 7.)

Lean-ajattelun ydinajatuksena on maksimoida asiakkaalle luotava arvo ja minimoida lisäarvoa tuottamaton toiminta eli hukka. Lean-organisaatiossa asiakasarvon ymmärtäminen ja sen parantaminen tapahtuu jatkuvalla arvonlisäysprosessin kehittämisellä, läpi koko arvoketun. Toiminnan johtamisen painopiste on siirretty koko arvoketjun optimointiin. Hukan poistaminen arvonlisäysprosessista tarkoittaa sitä, että asiakkaalle tuotettu arvo tapahtuu vähemmillä resursseilla. Prosessit tarvitsevat vähemmän ihmisten työtä, vähemmän tilaa, vähemmän pääomaa ja vähemmän aikaa valmistaa tuotteita, joissa on vähemmän vikoja ja kustannuksia. Valmistuksen nopeutuneen läpimenoajan myötä organisaatio pystyy vastaamaan asiakkaan muuttuviin toiveisiin nopeasti, hyvällä laadulla ja alhaisilla kustannuksilla. Tiedon hallinta ja toiminnan johtaminen on myös yksinkertaisempaa ja tarkempaa. Lean-ajattelu sopii monenlaiseen liiketoimintaan ja prosessiin, mutta se tarkoittaa suurta muutosta aikaisempaan tapaan johtaa liiketoimintaa. Kyseessä on pitkän aikavälin prosessi, joka vaatii pitkäjänteisyyttä. (Lean Enterprise Institute 2009.)

Monet yritykset ovat yrittäneet omaksua Lean-tuotantoa, mutta se on osoittautunut huomattavan vaikeaksi. Huonolla menestyksellä yritykset ovat aluksi muuttaneet pintapuolisesti vain tietyn osan yrityksen tuotannosta Lean-tuotannoksi. Keskitytään liiaksi vain työkaluihin ymmärtämättä, että Lean on kokonainen järjestelmä, jonka täytyy keskittyä koko organisaatioon. Usein oli myös niin, että ylempi johto ei ollut sitoutunut päivittäisiin operaatioihin ja jatkuvaan parantamiseen, joka on osa Leania. Kaikesta huolimatta Lean-tuotanto alkoi onnistua ja Leaniin perustuva liiketoiminta alkoi kukoistaa. Tämän seurauksena Lean-ajattelua alettiin siirtää tuo-

tannosta myös muihin toimintoihin kuten tilauskäsittelyyn, vähittäiskauppaan ja ilmailualalle. Lean periaatteet ovat myös laajentuneet toimitusketjuihin, tuotekehitykseen ja myös ohjelmistokehitykseen (kuvio 6). (Liker 2006, 7: Poppendieck & Poppendieck 2007, 7.)



Kuvio 6. Lean-hierarkiapuu

Toyota on laajentanut Lean-tuotannon koskemaan myös toimittajia. Massatuotanto ja Lean-tuotanto eivät sopineen hyvin yhteen, joten Toyota auttoi toimittajiaan omaksumaan TPS-tuotantojärjestelmän. Tämä Toyotan Lean-toimittajaverkko (supply chain) antoi yritykselle huomattavan kustannusedun. Virtuaalisen toimitusketjun avulla kumppaneita kohdellaan kuin he olisivat yrityksen sisällä. Tietoja vaihdetaan vapaasti, jotta toimitusketju säilyisi Leanina. (Poppendieck & Poppendieck 2007, 13.)

Kun Lean-tekniikat on toteutettu yrityksen tuotannossa hyvin ja tuotannon eräkoot ovat pieniä, tuotannossa joustavuus lisääntyy ja vaihtelevuus eri vaiheissa vähenee, läpimenoaika lyhenee, varastojen määrä pienenee ja käytettävissä oleva pääoma sekä tuottavuus lisääntyvät. Tämän myötä tuotteiden laatu paranee, oppiminen lisääntyy ja tuotantokustannukset pienenee. (Denning 2010, 118 - 121.)

2.3.3 Lean-tuotekehitys

Toyotan myötä Lean-ajattelu on laajentunut myös tuotekehitykseen. Todellinen ero Toyotan ja muiden autonvalmistajien välillä ei ole itse tuotantojärjestelmä, vaan käytössä oleva tuotekehitysjärjestelmä. Tuotekehitys on eri asia kuin valmistus,

mutta tutkimuksissa on kuitenkin todettu yhtäläisyyksiä (kuvio 7) Lean-tuotannon ja tehokkaan tuotekehityksen välillä. (Poppendieck & Poppendieck 2007, 13 - 15.)

Lean-tuotanto	**Lean-tuotekehitys**
Toistuvat koneistuksen asetusaikojen muutokset (tuotteen vaihtuessa)	Toistuvat tuotemuutokset (ohjelmistotoimitukset)
Lyhyt valmistuksen läpäisyaika	Lyhty tuotekehitysaika
Rajoitettu keskeneräisten töiden/tuotteiden määrää valmistusvaiheiden välillä	Rajoitettu keskeneräisten töiden/ominaisuuksien määrää tuotekehityksen työvaiheiden välillä
Tuotteiden pienet eräkoot valmistusvaiheiden välillä	Esitiedon siirtyminen tuotekehitysvaiheiden välillä
Varaston pienentäminen vaatii löysää kapasiteettiin ja tiedon virtauksen lisäämistä tuotantovaiheiden välille	Tuotekehitysajan pienentäminen vaatii löysää kapasiteettiin ja tiedon virtauksen lisäämistä tuotekehitysvaiheiden välille
Sopeutumiskykyä muuttaa valmistusmääriä, tuotetarjontaa ja tuotesuunnittelua.	Sopeutumiskykyä muuttaa tuotesuunnittelua, aikataulua ja kustannustavoitteita
Laajat tehtäväannot monitaitoisille tuotannon työntekijöille	Laajat tehtäväannot monitaitoisille tuotekehityksen työntekijöille
Keskitytään nopeaan ongelmanratkaisuun ja jatkuvaan prosessin parantamiseen	Keskitytään vaiheittaiseen innovointiin ja jatkuvaan tuotteen ja prosessin parantamiseen
Samanaikaisesti parannetaan laatua, toimitusaikaa ja valmistuksen tuottavuutta	Samanaikaisesti parannetaan laatua, tuotekehitysaikaa ja tuotekehitysprosessia

Kuvio 7. Yhtäläisyyksiä Lean-tuotannon ja tehokkaan tuotekehityksen välillä (Poppendieck & Poppendieck 2007, 14.)

Lean-tuotekehityksessä haastetta tuo tuotannon ja tuotekehityksen välinen ero. Tuotekehitys on prosessi, jossa luodaan uutta ja valmistuksessa pyritään prosessi toistamaan tehokkaasti. Leanin soveltaminen tuotekehitykseen on tapahtunut kahdessa eri vaiheessa. Ensimmäisessä vaiheessa, 1990-luvulla, sen katsottiin olevan liityntäpinta Lean-valmistukseen. Toisessa vaiheessa, 2000-luvulla, Lean-tuotekehitys on keskittynyt elinkaarikustannuksiin ja tuotekehitysajan lyhentämiseen erilaisilla työkaluilla. Lean-tuotekehityksestä on olemassa useita tulkintoja ja lähestymistapoja, joilla kaikilla on tavoitteena työkalujen kautta lyhentää tuotekehitysaikaa. (VTT 2008.)

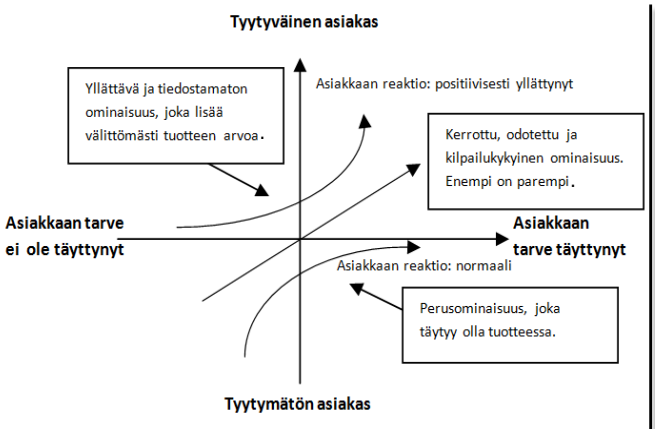
Samalla tavalla kuin tuotannossa, myös Lean-tuotekehityksessä keskitytään tehokkaaseen arvovirran rakentamiseen. Tuotekehityksessä arvovirta rakentuu tie-

don jalostamisesta, joka sisältää paljon muuttuvia elementtejä, ja on sen vuoksi haastavampi toteuttaa. Tuotannossa arvovirta koostuu ennustettavista ja tehokkaasti toistettavista tehtävistä jotka ovat kestoltaan tasalaatuisia. Tuotekehityksessä arvovirta rakentuu informaatiosta, erilaisista tehtävistä ja vaiheista, joiden kesto, toistettavuus ja resurssitarve vaihtelevat. (Ward 2007, 62 - 63: Rodriguez 2013, 49: Reinertsen 2009, 86.)

Kirjallisuuden perusteella Lean-tuotekehityksessä korostuu kaksi osa-aluetta: hukan poistaminen ja tehokkaan arvovirtauksen luominen. Tehokkaassa tuotekehitysprosessissa organisaation kokeilukierto toimii nopeasti, saa palautetta nopeasti, oppii nopeasti ja ilahduttaa asiakkaita täyttämällä arvolupauksen.

2.3.4 Arvo, arvovirta ja hukka

Asiakkaiden tarpeiden tunnistamiseen ja toiminnallisten ominaisuuksien arviointiin, sekä näiden suhdetta asiakastyytyväisyyteen ja asiakkaan kokemaan arvoon, voidaan tutkia esimerkiksi Kano-mallin avulla. Kano-mallin (kuvio 8) perusteella asiakkaan tarpeet ja toiminnalliset ominaisuudet voidaan jaotella kolmeen eri kategoriaan: vaadittu perusominaisuus (must-have), odotetusti tarpeet täyttävä (linear features) ja positiivisesti odottamaton (delight, surprise). (Cohn 2006, 112 - 117.)



		Ominaisuus puuttuu				
		Like	Expect	Neutral	Live with	Dislike
Ominaisuus sisältyy	Like	Q	E	E	E	L
	Expect	R	I	I	I	M
	Neutral	R	I	I	I	M
	Live with	R	I	I	I	M
	Dislike	R	R	R	R	Q

Kuvio 8. Kano-malli ja Kano-taulukko

Kano-malli jakaa tuotekriteerit viiteen kategoriaan: *pakollinen* (M), *yksiulotteinen* (L), *vetovoimainen* (E), *yhdentekevä* (I) ja *käänteinen* (R). Epäselviä ja tarkennusta vaativia kysymyksiä varten taulukossa on olemassa *kyseenalainen*-kategoria

(Q). Lajittelu tapahtuu kahden kysymyksen avulla: *tuote sisältää ominaisuuden* ja *tuote ei sisällä ominaisuutta*. Vastaus molempiin kysymyksiin annetaan samalla asteikolla: *tyytyväinen (Like)*, *näin asian pitää olla (Expected)*, *yhdentekevää (Neutral)*, *pystyn elämään asian kanssa (Live with)* ja *tyytymätön (Dislike)*. Lopuksi tulokset kirjataan Kano-taulukkoon analysointia varten. (Poppendieck & Poppendieck 2007, 49 - 50: Cohn 2006, 112 - 117.)

Lean-ajattelussa arvon määrittämiseksi prosessia tarkastellaan asiakkaan näkökulmasta. Tällä tavoin voidaan erottaa lisäarvoa tuovat vaiheet lisäarvoa tuottamattomista. Prosessivaiheiden arvo määritellään yleensä kolmella tasolla:

1. Lisäarvoa tuottava toiminta
2. Lisäarvoa tuottamaton toiminta
3. Lisäarvoa tuottamaton, mutta välttämätön toiminta. (Liker 2006, 28: Ward 2009, 19 - 21.)

Tietyissä tapauksissa voidaan puhua myös arvoa laskevista ja tuhoavista toiminnoista, kun asiakkaan näkökulmasta tuotteen lisäarvo onkin laskenut. Varastointi on esimerkki tällaisesta prosessivaiheesta, jos asiakkaan fyysinen tuote varastoinnin aikana pilaantuu. Toinen esimerkki on ohjelmistomuutoksen vaatimusmääritys, joka menettää arvoa vietettyään kohtuuttoman pitkän ajan tuotekehitysjonossa. (Neilimo & Uusi-Rauva 2010, 151.)

Prosessia pyritään tehostamaan poistamalla sellainen toiminta (hukka), joka ei asiakkaan näkökulmasta lisää arvoa. Prosessin tehostaminen tapahtuu yleensä prosessin arvovirran kartoituksella (Value stream mapping). Arvovirran alku- ja loppupiste määritellään, arvovirta kuvataan ja prosessi pilkotaan osiin, materiaalin ja informaation eteneminen selvitetään, hukka pyritään tunnistamaan ja poistamaan arvovirrasta. Arvovirran kuvaaminen luo pohjan kaikille muille Lean-työkaluille. (Liker 2006, 27 - 32, 275 - 276: Poppendieck & Poppendieck 2007, 83 - 85.)

Lean-ajattelussa arvovirtoja voidaan ajatella olevan kaksi; tuotekehityksen arvovirta ja valmistavan tuotannon arvovirta. Katsotaankin, että tuotekehityksen arvovirta luo valmistavan tuotannon arvovirran. Lean-tuotekehityksen tärkein asiakas on siis

valmistava tuotanto. Lean-tuotekehitys luo siis kahdenlaista arvoa; tuotantojärjestelmiä ja käyttökelpoista tietämystä. Valmistavassa tuotannossa arvovirran toiminnot muuttavat raaka-aineet asiakkaalle toimitettaviksi tuotteiksi. Lean-tuotekehityksen tehokkuus ja kannattavuus voidaankin mitata juuri valmistavan tuotannon kannattavuudesta. Tuotekehityksellä on arvoa vain silloin, kun valmistava tuotanto tuottaa parempia tuotteita omille ostaville asiakkailleen. Arvovirran kaikki vaiheet pitäisi käsitellä kokonaisuutena, arvovirran yhden vaiheen optimointi aiheuttaa yleensä kustannuksia ja vaikeuksia muille vaiheille. (Ward 2007, 9 - 11, 68.)

Wardin (2007, 18) mukaan melkein kaikki epäonnistuneet projektit ovat johtuneet siitä, että oikea tieto ei ole oikeassa paikassa oikeaan aikaan. Lean-tuotekehityksen tärkein arvo onkin oikean ja käyttökelpoisen tiedon tuottaminen ja jatkuva oppiminen. Käyttökelpoinen tietämys ehkäisee virheitä, laatuongelmia ja viivästymisiä. Käyttökelpoisen tietämyksen luominen tapahtuu kolmen erityyppisen oppimisen kautta:

- Integraatio (integration learning). Oppiminen tapahtuu siinä toimintaympäristöstä missä tuotetta käytetään. Saatu tieto auttaa ymmärtämään mikä olisi parasta asiakkaille, toimittajille ja partnereille.

- Innovaatio (innovation learning). Innovointi luo uusia mahdollisuuksia ja uutta osaamista.

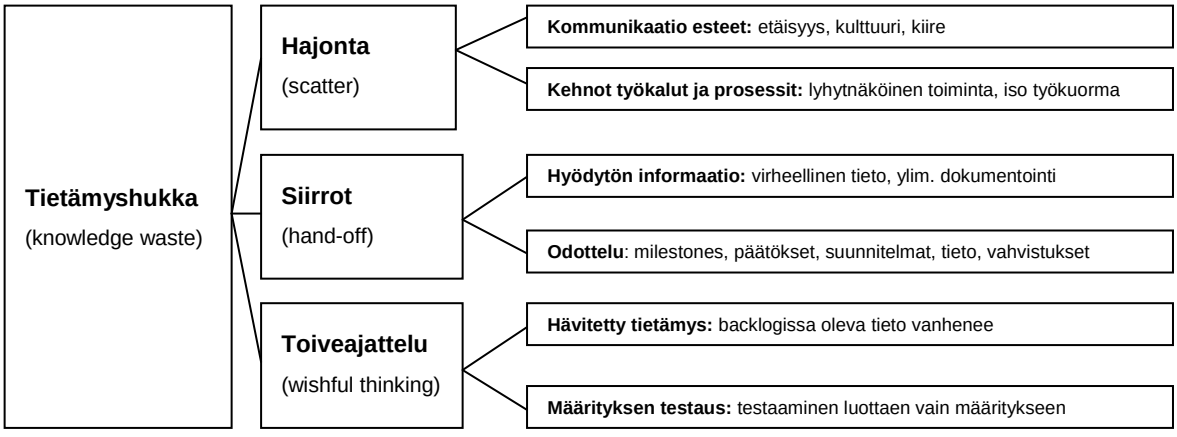
- Taustojen tutkiminen ja kelpoisuusselvitys (feasibility learning). Tämä oppiminen auttaa päätöksenteossa liittyen uusiin sovelluksiin estäen viivästymisiä, kustannus ja laatuongelmia. (Ward 2007, 18.)

Keskittyminen käyttökelpoiseen tietoon onkin Lean-tuotekehityksen ydin. Tuotekehityksen arvoa tuottamaton toiminta (hukka) on pääasiassa tietämyshukkaan, joka häiritsee prosessin arvovirran virtausta. Wardin (2007, 31) mukaan perinteisen tuotekehityksen vaiheisiin perustuva tapa ”*tehdä oikeita asioita”* aiheuttaa suuren osan hukasta ja myös vaikeuttaa tunnistamaan hukkaa. Suuri osa ajasta menee tiedon etsimiseen. (Ward 2007, 30 - 33.)

Hukan muodot on jaettu kolmeen eri pääkategoriaan, joiden japaninkieliset termit ovat: muda, muri ja mura. Muda on yleisimmin käytetty ja tarkoittaa tuotannossa lisäarvoa tuottamatonta toimintaa. Muri tarkoittaa ihmisen tai koneen ylikuormitusta jonkin ulkopuoliset asian vuoksi. Mura tarkoittaa vaihtelua ja epätasaisuutta prosessissa tai tuotannossa. Edellä mainituilla hukan muodoilla on suhde toisiinsa ja esiintyessään lopulta johtavat seuraavaksi esiteltäviin lisäarvoa tuottamattomiin toimintoihin. (Stewart 2012, 88 - 93.)

Toyotan tuotantojärjestelmää kehittämässä ollut konsultti Shigeo Shingo on listannut seitsemän tuotannossa esiintyvää, lisäarvoa tuottamatonta toimintaa ja hukan muotoa (muda). Nämä seitsemän tuotannossa tunnistettua hukan muotoa ovat: varastointi, ylikäsittely, ylituotanto, kuljetus, odottaminen, liike ja virheet. (Poppendieck & Poppendieck 2003, 4.) Liker (2006, 29) on lisännyt tähän listaan vielä kahdeksannen hukkatyypin eli työntekijän luovuuden käyttämättä jättämisen.

Ward (2007, 31) on kirjassaan kuvannut tuotekehityksessä esiintyvän hukan termillä tietämyshukka (knowledge waste). Tietämyshukka (kuvio 9) on jaettu kolmeen eri muotoon: hajonta, siirrot, toiveajattelu. Tietämyshukkaa syntyy aina kun erotetaan tieto, vastuu, toiminta ja palaute. Ratkaisuksi tähän ongelmaan esitetään monitaitoista tiimityötä, nopeaa palautteen saamista ja töiden tasaista virtausta. Tuotekehitysprosessin tekeminen näkyväksi auttaa löytämään hukkaa. (Ward 2007, 29 - 57.)



Kuvio 9. Tietämyshukan kategoriat
(Ward 2007, 31.)

Hajonta (scatter) on tuotekehitysprosessin virtausta ja tiimityötä häiritsevää toimintaa. Oikea tieto ei ole oikeassa paikassa oikeaan aikaa. Siirrot (hand-off) erottavat tiedon, vastuun, toiminnan ja palautteen siten, että prosessin vaiheet etenevät yleensä vaiheittain porttien läpi. Tästä seuraa se, että päätöksiä tekee henkilö, jolla ei ole riittävästi tietämystä tehdä sitä hyvin. Kiire ja vastuun jakaminen estävät myös tehokkaasti asian oppimista. Toiveajattelu tarkoittaa päätösten tekemistä ilman tietoa. Perinteinen tuotekehitys aloitetaan vaatimusten ja määritysten tekemisellä, minkä alussa asiakas ei tarkalleen tiedä mitä haluaa ja tuotekehitys ei tarkalleen tiedä mitä voi tehdä. Tällöin alussa lukittu määritys perustuu osittain toiveajattelulle, joka usein aiheuttaa ongelmia myöhemmin. (Ward 2007, 32 - 57.)

Reinertsen (2009, 1 - 3) on kirjassaan kuvannut Lean-tuotekehityksen periaatteita perustuen tuotekehityksen kustannuksiin, kannattavuuteen ja todelliseen tieteeseen. Tämän kirjan tärkein asia on virtaus (flow) ja virtaukseen negatiivisesti vaikuttavien asioiden käsittely. Tämän lähestymistavan ydin on tuotekehityksessä oleva fyysisesti näkymätön ja mittaamaton jonossa olevien töiden lista. Työt ovat pääasiassa informaatiota, eikä asiaa sen vuoksi ajatella aineellisena varastona, kuten valmistavassa tuotannossa. Kuitenkin tämä jono aiheuttaa suuren osan tuotekehityksen ongelmista ja vaikuttaa ennen kaikkea heikentävästi taloudelliseen suorituskykyyn. Tuotekehityksen tehtäväjonot lisäävät vaihtelevuutta yksittäisiin tehtäviin, lisäävät riskiä ja töiden läpimenoaikaa, hidastavat palautteen saamista ja muuttavat jatkuvasti priorisointia. Jonot vähentävät tehokkuutta, laatua ja motivaatiota. Kaikk-i edellä mainitut ongelmat ovat liiketaloudellista hukkaa. (Reinertsen 2009, 1 - 6, 21, 51.)

Reinerstenin (2009, 31 - 32) mukaan jonoja hallitakseen täytyy tietää jonojen koko ja jonossa olevien tehtävien arvo/kustannukset työaika mukaanlukien suhteutettuna aikaan (cost of delay, CoD). *Cost of Delay* -käsite on arvo, joka voitaisiin tuottaa tietyn ajanjakson aikana jos se olisi heti saatavilla, mutta viivästyminen tai viivyttely vähentää tuota arvoa ja lisää kuluja. *Cost of Delay* sisältää ominaisuuden taloudellisen arvon, kehitysprosessin aikana saadun informaation arvon ja arvion kuinka näiden kahden arvo häviää ajan mukana. Tarkan arvon laskenta on käytännössä haastavaa (ellei mahdotonta), mutta tällä tavoin saadaan ajalle hinta.

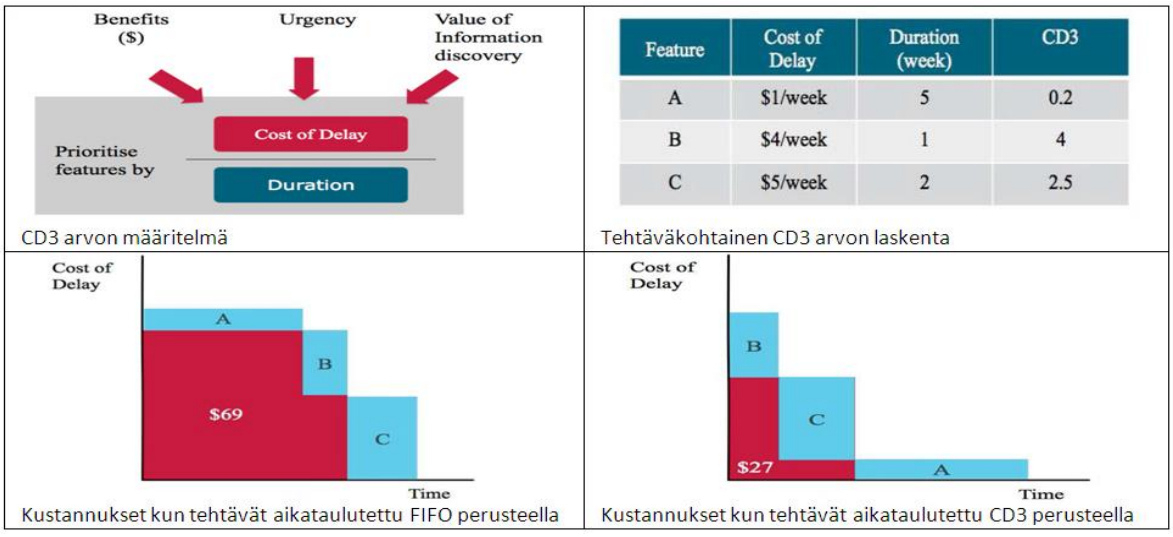
Kun tiedetään jonossa olevien tehtävien kustannukset, voidaan jonoa mitata ja hallita:

- tiedetään esimerkiksi, mitä viikon viivästyminen maksaa
- tehtäväjonossa olevia tehtäviä voidaan priorisoida kustannusten ja ajan perusteella taloudellisesti tehokkaasti. Tehtävät priorisoidaan jakamalla kustannukset tehtävän suorittamiseen arvioidulla ajalla (CD3-arvo).
- tiedon perusteella voidaan laskea erilaisia vaihtoehtoja kuten; tehdäänkö itse vai teetetäänkö alihankkijalla, viedäänkö tuote markkinoille nyt vai kehitetäänkö (innovoidaan) tuotetta vielä yksi kuukausi. (Reinertsen 2009, 30 - 48, 68 - 71.)

Ominaisuuksien liiketaloudellisen arvon ja hyödyn arviointiin voidaan käyttää neljää seuraavaa ryhmää. Ryhmät ovat:

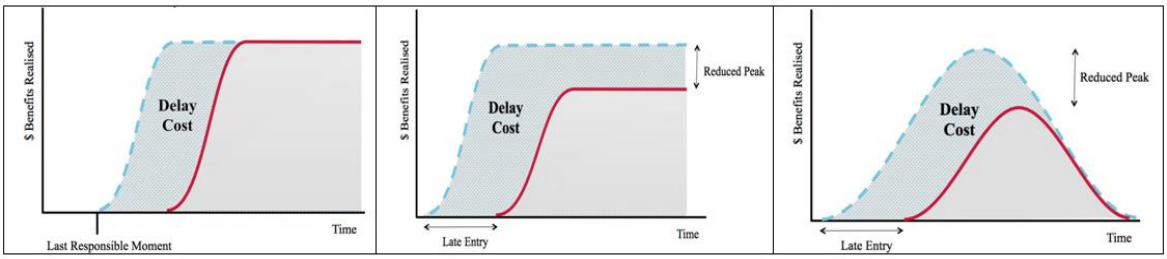
- Lisää tuloja – parantaa voittomarginaalia
- Suojaa tuloja – ylläpitää voittomarginaalia
- Vähentää kuluja – vähentää tämänhetkisten kustannusten määrää
- Välttää kuluja – vähentää mahdollisia tulevia kustannuksia. (Black Swan Farming 2013.)

Kuviossa 10 on esitelty FIFO- vs. CD3-priorisointitavan vaikutukset tehtävien kustannuksiin.



Kuvio 10. Tehtävien priorisointi CD3-arvon perusteella
(Black Swan Farming 2013.)

Kuviossa 11 on kuvattu kuinka ominaisuuden liiketaloudellinen arvo laskee ajan kuluessa (viiveprofiili).



Kuvio 11. Ominaisuuden liiketaloudellisen arvon laskeminen ajan kuluessa (Black Swan Farming 2013.)

Tuotekehityksen resurssien maksimaalinen hyödyntäminen lisää Reinertsenin (2009, 59) mukaan tuotekehityksen jonoja eksponentiaalisesti. Kun työ aikataulutetaan siten, että kapasiteetti kasvaa 60 prosentista 80 prosenttiin, jonon koko kaksinkertaistuu. Sama tapahtuu, kun kapasiteetti kasvaa 80 prosentista 90 prosenttiin ja siitä edelleen 95 prosenttiin. Tuotekehityksessä korkealla kapasiteetilla on taipumus lisätä vaihtelevuutta tehtävien aloitusajankohtaan ja itse suoritusaikaan. Tästä syystä vaihtelevuus lisää edelleen jonon kokoa, vaikka tulevien tehtävien määrä olisi ennustettavissa. Tutkimusten mukaan kapasiteetin kasvattaminen 75 prosentista 95 prosenttiin kasvattaa tehtäviin liittyvää vaihtelevuutta 25-kertaisesti. Tällöin jonon koon kasvamisen lisäksi tehtävien läpimenoaika kasvaa, palautteen saaminen viivästyy, laatu heikkenee ja prosessista tulee ylipäätään epävakaa. (Reinertsen 2009, 59, 64 - 65.)

Jonoja hallintaan Reinertsen (2009, 71) ehdottaa työvälineeksi kumulatiivista virtauskaaviota ja prosessien läpimenoajan määrittämisessä käytettävää Littlen lakia. Jonon koon tarkkailu on tärkeämpää kuin kapasiteetin ja läpimenoajan, sillä jonon koko vaikuttaa kapasiteettia nopeammin tehtävien läpimenoaikaan. Jonon koon tarkkailu kertoo tulevista ongelmista 20 kertaa nopeammin kuin läpimenoaikojen tarkkailu. Eräkokojen pienentäminen eli työn suorittaminen pieninä palasina vähentää myös läpimenoaikaa, vaihtelua, nopeuttaa palautteen saamista ja vähentää riskiä. Tuotekehityksessä eräkokojen pienentäminen on taloudellisesti tärkeämpää kuin valmistavassa tuotannossa. Töiden tasaisen virtauksen mahdollistamiseksi ja nopean läpimenoajan ylläpitämiseksi on tuotekehityksessä käytettävä

käynnissä olevien töiden rajoittamista (WIP). Tämä on lyhyellä aikajaksolla huomattavasti tehokkaampi työkalu vaihtelun pienentämiseen kuin eräkoon pienentäminen. (Reinertsen 2009, 71 - 76, 112 - 114, 141, 143.)

Yleisin ja valmistavasta tuotannosta tuttu työkalu käynnissä olevien töiden rajoittamiseen on nimeltään Kanban. Kanban on prosessityökalu missä tehtävät virtaavat imuohjatusti eri työvaiheiden läpi siten, että uusi tehtävä haetaan kun edellinen saadaan valmiiksi. Tuotekehityksessä uusi tehtävä valitaan työn alle CD3-priorisoidusta listata, ei valmistavasta tuotannosta tutulla FIFO-periaatteella. Kanban tekee työn virtauksen näkyväksi ja paljastaa prosessissa olevia ongelmia pysäyttämällä virtauksen, kun käynnissä olevien työtehtävien määrä ylittää sallitun rajan. Kun ongelmat ovat esillä, prosessin jatkuva parantaminen on mahdollista (Kaizen). Työn virtaus voidaan jakaa myös useampaan eri kategoriaan, joilla on omat WIP-rajoituksensa. WIP-rajojen avulla pyritään siihen, että töiden virtaus olisi nopeaa ja ruuhkatonta. (Reinertsen 2009, 148 - 149, 160 - 162, 166, 175, 191.)

Cost of delay -arvon (CoD) mukaan ominaisuuksia voidaan myös luokitella eri kategorioihin (Cost of delay profile). Tätä luokittelua voidaan hyödyntää Kanban taululla siten, että jokaisella profiililla on oma WIP-rajoitus. (LeanAgileChange 2014.)

2.4 Yhteenveto Leanista

Lean-filosofiassa on useita periaatteita mistä valita: Toyotan tuotantojärjestelmän kaksi pilaria, Toyotan tavan 14 periaatetta, kaikkiin prosesseihin sopiva Lean-ajattelun viisi periaatetta ja valmistuksen seitsemän lisäarvoa tuottamatonta hukan päätyyppiä. Kulttuuritaustasta johtuen lähestymistapojakin on ainakin kaksi, amerikkalainen Lean-ajattelu ja japanilainen *Toyotan tapa*. Toyotan arvoista ja toimintatavoista on kirjoitettu useita kirjoja. Tässä tutkimuksessa ei tutkita kuitenkaan Toyotan tuotantojärjestelmää eikä *Toyotan tapaa*, joten aiheet rajataan pois. Mutta edellä mainitut perusasiat on hyvä tietää. Lean-filosofian käsittely tämän tutkimuksen osalta päättyy tähän. (Stewart 2012: Liker, 2006.)

Wardin kuvaus tuotekehityksen tietämyshukasta (hajonta, siirrot ja toiveajattelu) kuvaa tämän työn tekijän mielestä hyvin ohjelmistotuotannossa esiintyviä ongel-

mia. Perinteisen toimintatavan vaiheittainen tai portti-mallin mukainen toimintatapa aiheuttaa tietämyshukkaa, koska siinä tietyllä tavalla erotetaan tieto, vastuu, toiminta ja palaute. Ward kehoittaakin siirtymään vaiheittaisesta tuotekehitysmallista enemmän rinnakkaiseen toimintatapaan (Ward 2007, 42).

Reinertseinin kuvaus Lean-tuotekehityksen virtauksesta ja taloudellisesta näkökulmasta oli mielenkiintoinen ja meni astetta syvemmälle kuin moni muu kirja. Kirjassa taloudellinen tehokkuus ja kannattavuus olivat etusijalla ja ne menivät jopa tiettyjen Lean-periaatteiden edelle. Asiakas tuotekehitystoiminnan arvokäsitteen ainoana tuomarina onkin liiketaloudellisesta näkökulmasta sekä virheellinen että vaarallinen. Myös päätöksen tekeminen etupainotteisesti tai viimeisellä mahdollisella hetkellä, jolloin tietoa on eniten (yksi Lean-ohjelmistokehityksen periaatteista), riippuu taloudellisesta näkökulmasta. Kun ominaisuuden kulut, tulos ja riski ovat aikariippuvaisia, jokaisella päätöksellä on optimaalinen ajan hetki. (Reinertsen 2009, 44.)

3 LEAN-OHJELMISTOKEHITYS

3.1 Ketterä ohjelmistokehitys

Ennen kuin käsitellään Lean-ohjelmistokehitystä, on mainittava taustalla olevan ketterän ohjelmistokehityksen arvot ja periaatteet. Vuonna 2001 luodun manifestin ydin on neljä arvoa, jotka konkretisoituva vielä 12 erilliseen. Manifesti muodosti käännekohdan Lean-ajattelun ymmärtämisessä ohjelmistokehityksessä (Rodriguez 2013, 49). Manifestin sisältö on seuraavanlainen:

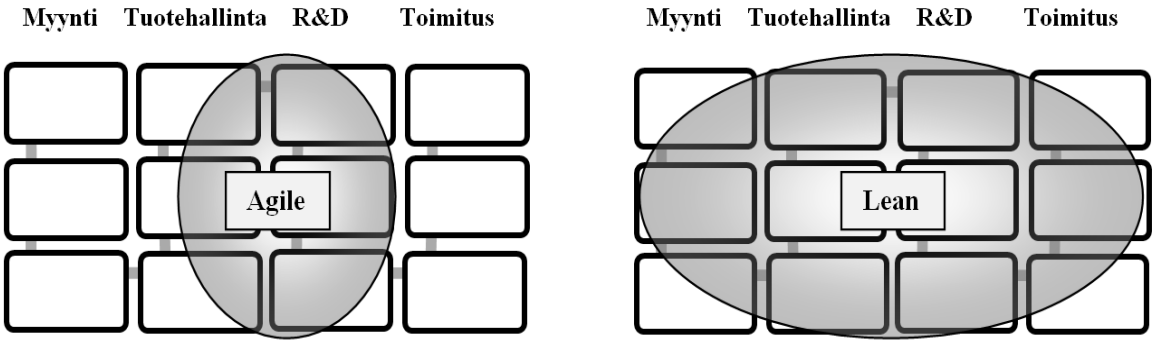
> ”*Löydämme parempia tapoja tehdä ohjelmistokehitystä, kun teemme sitä itse ja autamme muita siinä. Kokemuksemme perusteella arvostamme:*
>
> ***Yksilöitä ja kanssakäymistä*** *enemmän kuin menetelmiä ja työkaluja*
>
> ***Toimivaa ohjelmistoa*** *enemmän kuin kattavaa dokumentaatiota*
>
> ***Asiakasyhteistyötä*** *enemmän kuin sopimusneuvotteluja*
>
> ***Vastaamista muutokseen*** *enemmän kuin pitäytymistä suunnitelmassa*
>
> *Jälkimmäisillä asioilla on arvoa, mutta arvostamme ensiksi mainittuja enemmän.*” (Agile Manifesto Org 2001.)

Manifestin kannalta tärkeintä on toimiva ohjelmisto ja tyytyväinen asiakas. Prosessi, työkalujen käyttö ja dokumentaatio koetaan kuitenkin arvokkaaksi. Tämän pohjalta on luotu useita kevyempiä ohjelmistotuotannossa käytettäviä menetelmiä (XP ja Scrum) raskaiden ja suunnitelmaohjattujen kehitysprosessien tilalle (CMMI ja plan-driven). Keveiden ja ketterien menetelmien lähestymistavat ovat arvopohjaisia (value-driven), joissa keskitytään asiakkaan kannalta arvokkaimpiin ominaisuuksiin. Ominaisuudet rakennetaan inkrementaalisesti. Lisäksi keskitytään siihen, miten kehitystiimit työtään suorittavat, huomioidaan ihmisten kunnioitus ja kommunikaatio. Manifestin perusperiaatteet kuulostavat hyviltä, mutta osassa 12 perusperiaatteessa on ollut lähtökohtana pienehköt asiakasprojektit. Suurissa tuotekehityshankkeissa tiettyjä periaatteita, kuten asiakkaan jatkuva läsnäolo ja kasvokkain kommunikointi on vaikea soveltaa. Nykyisin ketteristä menetelmistä yleisin käytössä oleva on Scrum-menetelmä. Muista menetelmistä kuten XP:stä, käytössä on vain hyvät ideat, kuten jatkuva integrointi ja lyhyet iteraatiot. (Haikala & Mikkonen 2011, 43 - 46.)

Edellä mainitut kevyet menetelmät ovat ensimmäisen sukupolven ketteriä menetelmiä. Menetelmät perustuvat suoraviivaiseen toimintaan, jossa tiimit suorittavat kiinteämittaisen iteraation aikana iteraation alussa sovitut tehtävät. Yleensä nämä menetelmät on tarkoitettu pienille ja keskisuurille tiimeille projektin toteutusvaiheeseen hallintaan. Ongelmana näissä menetelmissä on koettu mukautuvuuden ja muutosten rajoittaminen iteraation aikana. Lisäksi ongelmaksi koetaan menetelmien skaalatutuvuuden hallinta tiimien määrän kasvaessa. Ketteryys ei ole myöskään toiminut kokonaisvaltaisen liiketoiminnan ja yrityksen pitkän tähtäimen tavoitteiden kanssa. Yleensä myynti ja tuotehallinto eivät toiminnassaan ketteryyttä käytä. Toimintaa ei tuotekehityksen tavoin johdeta priorisoidun tehtävälistan kautta. Ketterät menetelmät eivät myöskään helposti skaalaudu yhden tiimin projektista usean tiimin moniprojektiympäristöksi. (Net Objectives 2012b: Aalto-Yliopisto 2012.)

Kun ketterässä menetelmässä aletaan soveltaa Lean-menetelmiä, menetelmän katsotaan kuuluvan ketterien menetelmien toiseen sukupolveen. Tällöin menetelmästä tuleekin ajattelutapa, jota käytetään koko ohjelmistoprosessin parantamiseen (kuvio 12). Menetelmän ensisijainen tavoite on se, että kaikkien organisaation toimintojen ja resurssien tulisi olla keskittyneitä asiakasarvon luomiseen. Lisäksi Lean-menetelmät tarjoavat huomattavan määrän oivalluksia, jotka ovat erityisen hyödyllisiä poistamaan prosessista hukkaa eli viivettä ja ylimääräistä työtä. (Net Objectives 2012a.)

Kuviossa 12 on esitetty ketterien menetelmien soveltamisalue verrattuna Lean-menetelmiin.



Kuvio 12. Soveltamisalue Agile vs. Lean

VersionOne julkaisee vuosittain *Agile survey* -tutkimuksen. Uusimman tutkimuksen mukaan 88 % vastanneista yrityksistä käyttää ketteriä menetelmiä. Scrum-sukuisia menetelmiä käyttää 73 % vastaajista. Kanban-tyyppisten menetelmien osuus on noin 12 %. Kanbania käytettiin pääasiassa organisaation sisäisissä prosesseissa. Puhtaasti Lean-menetelmiä käyttävien osuus oli 3 %. (VersionOne 2014a.)

Vastaajista 15 % ilmoitti, että yksikään ketterän menetelmän mukainen projekti ei ollut epäonnistunut. Jos projekti oli epäonnistunut, syyksi ilmoitettiin seuraavaa: yhtiön arvot eivät olleet sopivia ketterien menetelmien kanssa (13 %), ulkoinen paine palata noudattamaan perinteistä vesiputous-menetelmää (10 %) ja ongelmia kommunikaatiossa (10 %). Ketterien menetelmien laajemmalle käytölle koettiin seuraavia esteitä: organisaation jäykkä kulttuuri (52 %), yleinen muutosvastarinta (42 %) ja ketterien menetelmien sovittaminen perinteisiin menetelmiin (35 %). (VersionOne 2014a.)

Edellisessä 2012 tutkimuksessa projektin epäonnistumisen syyksi ilmoitettiin seuraavia organisaatiotason asioita: epäonnistuttiin ihmisten yhdistämisessä (34 %), epäonnistuttiin tiimipohjaisen kulttuurin opettamisessa (28 %) ja kommunikointi epäonnistui tuoteomistajan ja kehitystiimin välillä (21 %). (VersionOne 2013.)

Uuden tutkimuksen perusteella ketteriä menetelmiä pidempään käyttäneiden mielestä menetelmän kolme parasta etua olivat: muuttuvien vaatimusten parempi hallinta, lisää tuottavuutta, parantaa projektin näkyvyyttä. Seuraavia menetelmän etuja oli: parantaa laatua, toiminta linjassa liiketoiminnan kanssa, vähentää riskiä, nopeuttaa markkinoille pääsyä ja yksinkertaistaa kehitysprosessia. (VersionOne 2014a.)

3.2 Lean-ohjelmistokehitys

VersionOnen (2014a) Agile survey -kyselyn perusteella Leanin osuus ketterien menetelmien käytössä oli 3 %. Poppendiekin (2007, 17) mukaan Lean-ohjelmistokehitys on Lean-tuotekehityksen alamuoto ja Lean-ohjelmistokehitykselle ei ole olemassa yhtä oikeaa lähestymistapaa tai prosessia. On vain olemassa Lean-ajattelu ja siitä johdettuja periaatteita, jotka on ymmärrettävä, kun

räätälöidään omaa Lean-ohjelmistoprosessia. Lean ei ole sama kuin esimerkiksi testivetoinen kehitys (TDD). Testivetoinen kehitys on käytäntö, joka tukee Lean-ohjelmistokehityksen *rakenna laatu ohjelman sisään* -periaatetta. Lean-ohjelmistoprosessia rakennettaessa on huomioitava ihmiskeskeisyys, hyväksyttävä tietotyön monimutkaisuus ja epävarmuus, työskenneltävä kohti parempaa taloudellista tulosta ja luoda käyttökelpoista tietoa. (Rodriguez 2013, 49.)

Pilar Rodriguezin (2013, 5) väitöskirjan perusteella Lean-menetelmiin perustuva ohjelmistokehitys on väline laajentaa nykyään suosittujen ketterien menetelmien käyttöä. Lean-menetelmien ja muiden ketterien menetelmien yhteensopivuutta ei vielä ymmärretä, eikä niiden parasta yhdistelmää myöskään ymmärretä riittävän hyvin. (Rodriguez 2013, 5, 49.)

Teollisuustuotannosta peräisin oleva Lean koetaan monitulkintaiseksi ohjelmistotuotannossa. Ohjelmistojen kehittäminen eroaa valmistavasta tuotannosta monilla eri tavoin. Lean-ajattelun yleinen arvokäsite ei ole suoraviivainen ohjelmistotuotannossa. Ohjelmiston helppo muunneltavuus (aina tarvittaessa) on hallitseva tekijä tuotteen kokonaisarvossa. Ohjelmistot ovat muokattavia, eikä niiden arvo siten ole sidottu yhteen tuotekehitysvaiheeseen. (Rodriguez 2013, 48 - 49.)

Myös valmistavan tuotannon hukkakäsite on ristiriitainen asia. Ohjelmistotuotannon hukka ei välttämättä esiinny alkuperäisen seitsemän hukan mukaisessa muodossa. Valmistavassa tuotannossa hukan lähde ja esiintyminen voidaan havaita tarkkailemalla fyysisen materiaalin virtausta, sekä koneen tai työntekijän toimintaa. Ohjelmistotuotannon työtehtävät ja työvaiheet ovat enemmänkin aineettomia ja siten vaikeuttaa hukan tunnistamista ja prosessin parantamista. Arvovirtakuvauksen ja töiden virtauksen periaatteet ovat haastavia, sillä ohjelmistotuotannon prosessi on pääasiassa näkymätöntä tietoa. Prosessissa ominaista on myös se, että toistettavuus puuttuu ja inhimillinen tekijä on hallitseva tekijä. Ohjelmistoprosessi tuottaa mahdollisesti vain yhden kappaleen lopullisesta tuotteesta. On tietysti ohjelmistoja, joista prosessi valmistaa muunnelmia samasta tuotteesta. Valmistavassa tuotannossa ihmisen läsnäoloa tarvitaan käyttämään automatisoituja koneita. Ohjelmistotuotannossa olennaisinta on luovuus, tieto ja kokemus. (Rodriguez 2013, 48 - 49.)

Vaikka perustavanlaatuinen ero ohjelmistotuotannon ja teollisen valmistuksen välillä tunnistetaan, ohjelmistoyritykset käyttävät räätälöityjä tulkintoja Lean-periaatteista. Lean-käsitteitä soveltamalla ohjelmistotuotannosta saadaan järkevämpää ja voi johtaa laadun ja prosessin paranemiseen. Lean-periaatteet voivat oikein käytettynä auttaa ohjelmistotuotannon hallintaan liittyvissä kysymyksissä, kuten: lyhyempi toimituskierto, nopeuttaa markkinoille pääsyä käyttämällä prosessissa virtausta, määritellään keskeiset suorituskykyindikaattorit perustumaan arvoon, parannetaan yhteistyötä asiakkaisiin käyttämällä imuohjausta ja vältetään lyhyen aikavälin ajattelua käyttämällä jatkuvaa parantamista pyrkimällä täydellisyyteen. Lean-periaatteita sovellettaessa työkalut ja käytännön periaatteet pitää sovittaa sopiviksi. (Rodriguez 2013, 49.)

3.2.1 Lean-ohjelmistotuotannon tulkintoja

Lähtökohtana voidaan käyttää seuraavia tulkintoja usealta eri henkilöltä ja teoksesta. Ohjelmistotuotannon näkökulmasta Mary ja Tom Poppendieckin sekä Womackin ja Jonesin Lean-tulkinnat ovat tutkimuksen perusteella yleisimmin viitatut tulkinnat.

Mary ja Tom Poppendieck. Poppendieckit ovat luoneet seitsemän Lean-ohjelmistokehitykseen soveltuvaa periaatetta ja kuvanneen seitsemän ohjelmistotuotannossa esiintyvää hukkaa. Nämä periaatteet ovat peräisin valmistavan tuotannon Lean-ajattelusta.

1. Poista hukka (eliminate waste)
    - Ymmärtämällä ensin mitä on arvo ja poistamalla kaikki turha työ
2. Rakenna laatu ohjelman sisään (build quality in)
    - Virheiden välttäminen kehitysprosessin aikana. Testaamalla nopeasti, automatisoimalla toimintoja ja refaktoroimalla.
3. Tietämyksen luominen (create knowledge)

- Nopea palautteen saanti ja pyrkimys jatkuvaan parantamiseen. Ohjelmiston asteittainen kehittäminen tarkentamalla.

4. Lykkää sitoutumista (defef commitment)
    - Ylläpidä vaihtoehtoja ja tekemällä päätöksiä viimeisellä mahdollisella hetkellä, jolloin on saatavilla eniten tietoa.
5. Toimita nopeasti (deliver fast)
    - Pienissä erissä ja rajoittamalla käynnissä olevien töiden määrää. Toimittamalla usein ja nopeasti. Tällöin asiakas ei ehdi muuttaa mieltään.
6. Kunnioita ihmisiä (respect people)
    - Ihmiset tekevät työtä autonomisissa tiimeissä, joilla on pätevät johtajat.
7. Optimoi kokonaisuus (optimize the whole)
    - Toteuta Lean koko arvovirtaan, ei yksittäisiin vaiheisiin.

Ohjelmistotuotannon seitsemän erityyppistä hukkaa ovat: osittain tehty työ, ylimääräiset ominaisuudet, uudelleen oppiminen, tuotteen luovutukset, tehtävien vaihdot, viivästykset ja virheet. (Poppendieck & Poppendieck 2007, 23 - 41.)

Womack ja Jones. Womackin ja Jonesin määrittelemät Lean-ajattelun viisi periaatetta: arvo, arvovirta, virtaus, imu ja täydellisyys. (Lean Enterprice Institute 2009.)

Reinertsen. Reinertsenin kuvaamat periaatteet tuotekehityksen virtauksesta, jonojen hallinnasta, eräkoon pienentämisestä, käynnissä olevien työtehtävien määrän rajoittaminen ja nopea palutteen saanti. (Reinertsen 2009, 15.)

Ward. Wardin esittämät Lean-tuotekehityksen periaatteet asiakasarvon luomiseen ja tietämyshukan estämiseen. (Ward 2007.)

Anderson. Andersonin mukaan Kanban-menetelmä on hyvä tapa tuoda Lean-ajattelua organisaatioon. Kanban-mentelmän viisi keskeistä periaatetta: visualisoi työn kulku, rajoita käynnissä olevien työtehtävien määrää, tee prosessikäytännöistä selkeitä, mittaa ja hallinnoi työnvirtaa, käytä valmiita malleja kehitysmahdollisuuksien tunnistamiseen. (Anderson 2010, 15.)

Larman ja Vodde. Larmanin ja Vodden kaksi pilaria: *Ihmisten kunnioitus* ja *jatkuva parantaminen*. Sekä 14 periaatetta: johdon päätökset perustuvat pitkän aikavälin filosofiaan, virtaus, imuohjaus, vähemmän vaihtelua ja ylirasitusta, pysäytä ja korjaa, master normit, käytä visuaalista kontrollia, käytä testattua teknologiaa, kasvata johtajia, kehitä työntekijöitä, kunnioita yhteistyökumppaneita, mene ja ymmärrä tilanne käytännössä, tee päätökset hitaasti konsensuksella, kehity oppivaksi jatkuvalla parantamisella. (Larman & Vodde 2009.)

Mary ja Tom Poppendieck. Poppendieckin (2013) uusimmassa kirjassa Lean kuvataan ajattelutapana. Ohjelmistotuotanto keskittyy suunnittelemaan ja toimittamaan asiakkaille oikeita ominaisuuksia, jotka aidosti ilahduttavat heitä. Tuotekehitysprosessi on oikeiden asioiden tekemistä. Uusi tieto virtaa tasaisesti läpi suunnittelu- ja toimitusprosessin tarjoten nopean palautteen asiakkaalta. Väärien asioiden rakentaminen ymmärretään ohjelmistotuotannon suurimpana hukkana. Toiminta on keskittynyt kokonaisuuden optimointiin, ihmisiin ja asiakkaisiin. (Poppendieck & Poppendieck 2013, 3, 6 - 9, 75, 92, 102.)

Denning. *Radical Management* kirjan seitsemän periaatetta. Työn johtamisessa keskitytään asiakkaan ilahduttamiseen (client delight) ja todellisen asiakasarvon tuottamiseen jatkuvan innovoinnin kautta. Työ tehdään itseorganisoiduissa tiimeissä, jotka tuottavat asiakkaille arvoa jokaisessa iteraatiossa (ketterät menetelmät). Jatkuvaa asiakasarvoa tuottava työn virtaus vaatii toimivaa kommunikointia kaikilta osapuolilta. (Denning 2010, 40 - 44.)

3.2.2 Lean-ohjelmistokehitykseen siirtyminen

Lean-menetelmiin siirtyminen tarkoittaa organisaatiolle suurta muutosta. Lean-ohjelmistokehityksen periaate *Optimoi kokonaisuus* tarkoittaa, että koko organi-

saatio siirtyy käyttämään Lean-periaatteita. Tällainen muutos ei voi toteutua ilman ylimmän johdon tukea, strategiaan ja visiota. Muutos tarvitsee pitkän tähtäimen tavoitteet ja ohjeet ylimmältä johdolta, niitä tarkennetaan organisaation alemmilla tasoilla. Koska Lean on kokonainen järjestelmä, vaatii se huomattavaa keskittymistä johdon ja henkilöstön koulutukseen, kommunikointiin ja ajattelumallin muuttamiseen. (Cloud Software Finland 2012: Liker 2006, 12 - 13: Rodriguez 2013, 107.)

Käytännössä tämä tarkoittaa sitä, että ohjelmistoyritykset toteuttava Lean-periaatteita osittain, sillä tuotekehityksen ulkopuolella olevat osastot eivät yleensä käytä Lean-ajattelua. Lean-ajattelun laajentaminen koko organisaatioon koetaankin haastavaksi. Lisäksi tutkimuksissa on todettu, että vain osa Lean-työkaluista on todellisuudessa soveltuvia ohjelmistokehitykseen. Osa hyödyllisistä työkaluista, kuten käynnissä olevien töiden määrän rajoittaminen (WIP) ja toiminnan pysäyttäminen ongelmanratkaisun ajaksi (Andon, stop-the-line), ovat muutosvaiheen alussa voitu kokea turhauttavaksi. Lean-ajattelu on kuitenkin helpottanut ongelmien juurisyiden löytämistä. (Rodriguez 2013, 53, 117 - 118.)

Tutkimuksen myötä Lean-ohjelmistokehitykseen siirtymisen haastavuus onkin nähtävissä VersionOnen *Agile survey* -kyselyn tuloksessa, jonka perusteella yksistään Leania käyttävien osuus ketterien menetelmien käytössä oli 3 %. Huomioitava on kuitenkin Rodriguezin (2013, 98) tutkimuksissa saatu tulos, jossa yhdistetty Lean-Agile-menetelmän käyttäjien osuus oli 21,6 %. Lean-ohjelmistokehitykseen siirtyminen tapahtuu asteittain yhdessä ketterien menetelmien kanssa. Lean-ajattelun *Optimoi kokonaisuus*-periaatteen myötä Lean pitäisi ottaa käyttöön koko organisaatiossa, mikä lisää siirtymisen haastavuutta. Siirtyminen Kanban-työkalun käyttöön on tutkimuksissa kuitenkin todettu soveltuvan hyvin ohjelmistotuotantoon. (Rodriguez 2013, 53 - 56, 96 - 98, 106 - 107.)

Hyvän kokonaiskuvan Lean-ohjelmistokehityksen laajasta viitekehyksestä saa tutustumalla Mary ja Tom Poppendieckin kirjasarjaan (2003, 2007, 2010, 2013). Siirtyminen eli transformaatio Lean-ohjelmistokehitykseen voidaan toteuttaa mukaillen heidän seitsemää periaatettaan:

- Kokonaisuuden optimointi. Huomioidaan Lean-periaatteet koko arvoketjussa ja arvoketjun sujuvuutta myös mitataan.

- Tiimityö ja ihmisten kunnioitaminen. Koulutetaan johtajia, korostetaan ammattiylpeyttä ja laatua, siirretään vastuuta ja valtaa alas työtä suorittaville tiimeille.

- Toimitaan nopeasti. Työ tehdään lyhyissä iteraatioissa ja pienissä erissä. Keskitytään resurssien maksimaalisen käytön sijasta virtaukseen ja nopeaan tuotantoon. Työ tehdään valmiiksi ennen kuin otetaan uusi työ. Rajoitetaan työn määrää prosessissa läpimenon nopeuttamiseksi.

- Lykätään päätöksentekoa. Vaatimusmäärittelyt tehdään rinnakkain kehitystyön kanssa, ei pyritä täydelliseen vaatimusmäärittelyyn etupainotteisesti. Poistetaan arkkitehtuurisia riippuvuuksia siten, että ominaisuuksia voidaan lisätä milloin vain.

- Tietämyksen luominen. Tiimeille annetaan mahdollisuus kehittää omaa toimintaansa. Koulutetaan ja kehitetään ongelmanratkaisukykyä. Luodaan prosessiin läpinäkyvyyttä.

- Keskitytään laatuun ja jatkuvaan laadunvarmistukseen. Integroidaan koodia jatkuvasti, automatisoidaan rutiinitehtävät, siivotaan ja refaktoroidaan ohjelmakoodia säännollisesti.

- Poistetaan hukka ja jäte. Keskitytään hukan poistamiseen prosessista, opetellaan tunnistamaan asiakasarvo ja turha työ. Keskitytään oikeiden ominaisuuksien kehittämiseen. (Poppendieck & Poppendieck 2007, 23 - 41: Helsingin Yliopisto 2014)

Lean-menetelmiin siirtymisen helpottamiseksi on luotu myös erilaisia transformaatiomalleja. Näiden mallien osalta yrityksen on valittava sopivin malli ja sovellettava sitä itselleen parhaalla tavalla. Lisää tietoa näistä malleista löytyy Cloud Software Finlandin (2012) verkkojulkaisusta.

3.2.3 Lean-menetelmät laajentavat ketteriä menetelmiä

Ketterät menetelmät tukevat monia Lean-ajattelun ja Lean-ohjelmistokehityksen periaatteita. Useassa eri tutkimuksessa on todettu, että Lean-menetelmillä voidaan parantaa ketteriin menetelmiin perustuvaa ohjelmistokehitystä. Voidaankin nähdä, että Lean parantaa koko ohjelmistoprosessia tunnistamalla ja poistamalla siitä hukkaa ja lisäämällä arvoa. Scrum ketteränä menetelmänä tekee saman tiimin sisäiselle tavalle rakentaa ja ohjata tuotteen kehitystyötä. Ikosen (2011) tutkimuksen mukaan Lean myös laajentaa ketteriä menetelmiä antaen opastusta uusiin tilanteisiin, pitää keskittyä enemmän tuotekehitysaikaan (läpimenoaika) kuin resurssien kapasiteettiin, ja optimoidaan mieluimmin kokonaisuutta kuin yksittäisiä vaiheita (Ikonen 2011, 34).

Eräässä kokeilussa Lean-menetelmien myötä ketterän projektin ajallinen kesto laski 40 % verrattuna perinteisen vesiputousmalliin, ja kustannukset alitettiin 10 %:lla (Cloud Software Finland 2012). Toisessa tutkimuksissa tuotekehityksen tuottavuus parani yli 30 %, sekä asiakastyytyväisyys että työympäristö kokivat huomattavaa parannusta (Rodriguez 2013, 96).

Yhteenvetona voidaan todeta Lean-ohjelmistokehityksen myötä tulevan seuraavia uusia elementtejä:

- Keskitytään todelliseen asiakasarvoon, joka vaatii joustavuutta asiakkaan muuttuviin vaatimuksiin.
- Hukkatyön poistamiseen ja vähentämiseen perustuva työkulttuuri
- Keskitytään prosessin läpinäkyvyyteen, yhteistyöhön ja taloudelliseen tuottavuuteen (ROI). Lisäksi kehitetään päätöksentekoa.
- Kanban eli kysyntään perustuva “pull”-menetelmä ja keskeneräisten töiden rajoittaminen (WIP).
- Keskitytään jatkuvaan parantamiseen (Kaizen) ja koko prosessin arvovirtaan eli optimoidaan ja mitataan kokonaisuutta (end-to-end flow).
- Keskitytään käyttökelpoisen tietämyksen luontiin ja sen jakamiseen.

Kaiken taustalla on asiakasarvo, ihmiskeskeisyys ja hyvät työkalut, jotka mahdollistavat läpinäkyvän toiminnan ja yhteistyön. Lean-menetelmissä ohjelmistotuotannon työ on jaettu arvovirtoihin ja juuri näiden arvovirtojen avulla Lean laajentaa nykyisiä ketteriä menetelmiä yhdistämällä organisaation eri osa-alueita. Arvovirtoja voi olla rinnakkaisia ja peräkkäisiä, työ ja informaatio siirtyvät arvovirrasta toiseen Kanban-menetelmän avulla kysynnän perusteella (imuohjatusti).

3.3 Pohdintaa Lean- ja Agile-ohjelmistokehityksestä

Lean-käsite on ohjelmistotuotannossa tällä hetkellä kohtalaisen uusi asia ja kokemusperäistä tietoa on aika vähän. Nykyinen tietämys aiheesta annetaan kirjoissa, joissa yleensä kirjoittaja tekee omia tulkintojaan Lean-ohjelmistokehitysestä ja sen käytöstä. Todellista tietoa siitä, mitä Lean-käytäntöjä ja työkaluja on käytännössä sovellettu, on edelleen aika niukasti olemassa (Rodriguez 2013, 58.). Hyvän kuvan Lean-ohjelmistotuotannosta saa tutustumalla aiheeseen liittyviin tutkimuksiin, joihin tieto on valmiiksi tiivistetty useasta eri lähteestä. Näitä tutkimuksia alkaa olla olemassa yhä enemmän ja enemmän. Eri oppilaitosten opintomateriaalit antavat myös tiivistetyn yleiskuvan Lean-ohjelmistotuotannosta. Taustalla on filosofia ja periaatteita, asiakkaan hyvä ymmärtäminen ja keskittyminen asiakasarvoon, hukan poistaminen, toisten ihmisten kunnioittaminen, asioiden virtaus organisaation läpi oikeaan aikaan ja pyrkimys jatkuvaan parantamiseen.

Oulun yliopiston tutkimusten (2014) perusteella Leanin ja ketterien mentelmien käytön yhdistämisen ansiosta ohjelmistokehitys on saavuttanut hyötyjä:

- Liiketoiminnan merkittävä kasvu
    - 50 – 70 % nopeampi toimintasykli
    - 30 % kannattavuuden parantumien
    - Asiat tehdään oikein ensimmäisellä kerralla (ei hukkaa)
    - Sovellukset saadaan nopeammin asiakkaille
- Uuden liiketoiminnan kasvattamien ja kehittäminen

    - Menetelmien yhdistäminen mahdollistaneet 50 % kasvun

- Toiminnan yleinen nopeutuminen
    - Uusien versioiden tuottaminen lyhentynyt 4 kuukaudesta 3 viikkoon
    - Läpimenoaikojen parantuminen 30 – 50 %
    - Uusia tuotteita saadaan nopeammin markkinoille
- Tuotteiden laadun parantuminen
    - Asiakkaan kokeman laadun parantuminen 50 %
    - Lähdekoodin teknisen velan vähentyminen 25 %
    - Asiakastyytyväisyyden huomattava parantuminen
    - Testauksen ja integroinnin automatisointi
    - Ongelmatilanteissa tuotanto pysäytetään ja virheet korjataan välittömästi
- Toimintatapojen kehittäminen
    - Ketterät menetelmät kaikkiin tuotantolinjoihin tuottaa nopeammat vasteajat, kehitysajat sekä parantuneen laadun.
    - Parantunut tuotantoprosessi johtanut tehokkuuteen, laatuun ja henkilökunnan sitoutumiseen.
- Yleinen työtyytyväisyyden lisääntyminen. (Oivo 2014.)

Ketterissä menetelmissä ja erityisesti Lean-menetelmien käytössä on aina kuitenkin huomioitava asiakkaan osaaminen eli onko projekti mahdollista toteuttaa ketterää mallia tai Lean-mallia noudattaen. Asiakasta pitää tarvitaessa kouluttaa, sillä yksin on vaikea olla ketterä. Lean-ohjelmistokehitys vaatii asiakkaalta vahvaa sitoutumista, sillä projektia ei välttämättä aloiteta kaiken kattavalla määrittelyvaiheella, vaan kehitystyö aloitetaan kevyesti määritellyillä tuotekehitysjonoon kirjatuilla ominaisuuksilla. Kirjattujen ominaisuuksien täytyy kuitenkin olla asiakkaalle oikeita

ominaisuuksia, sillä väärien asioiden rakentaminen nähdään ohjelmistotuotannon suurimpana hukkana (Poppendieck & Poppendieck 2013, 7). Ominaisuuksia priorisoidaan ja niiden määrityksiä tarkennetaan toteutuksen edetessä eli uusia päätöksiä tehdään aina iteraation tai version valmistuttua. Jatkuvasti tiedetään, mitä ollaan tekemässä ja tarvittaessa aikataulu tai ominaisuudet joustavat. Ketteryydessä voi kuitenkin olla vaarana, että jatkuvan läsnäolon myötä asiakas väsyy ja uupuu. Tekniset taitojen lisäksi projektin onnistumiseen tarvitaan myös sosiaalisia taitoja molemmilta osapuolilta.

Jos asiakas ei ole valmis toimimaan edellä mainitulla aikaveloitteisella tavalla, on toimittava perinteisemmällä tavalla. Kiinteähintaisessa projektissa, jossa on tiukka aikataulu ja toimitettavat asiat tarkasti kuvattuna, ei välttämättä tuota asiakkaan toivomaa lopputulosta. Tämän työn tekijän kokemuksen perusteella voidaan todeta, että jos toimittajalla ja asiakkaalla on pitkä yhteinen historia räätälöitävän ohjelmistotuotteen kanssa, perinteinen kiinteähintainen projekti on hyvä ratkaisu. Lean-mallin mukainen projektisopimus voisi olla yhdistelmä aikavelotteisesta ja kiinteähintaisesta sopimuksesta. Tällöin toimittaja pystyy arvioimaan aikaveloitteisen osuuden aikana toteutuvat kustannukset ja asiakas saavuttaa Lean-mallin mukaisesti oikean lopputuloksen. Sopimuksen loppuosuus voi tarvittaessa olla kiinteähintainen.

4 KANBAN

Japaninkielinen sana Kanban tarkoittaa suomeksi näkyvää taulua, ja menetelmänä se liittyy Leanin *juuri oikeaan aikaan* -periaatteeseen, hukan poistamiseen ja tehtävien virtaukseen. Kanban-menetelmä on prosessityökalu tuotannon ohjaukseen, joka auttaa löytämään nykyisestä prosessista ja toimintatavasta kehittämiskohteita. Valmistavassa teollisuudessa Kanban on ollut käytössä pitkään. Toyotalla menetelmä on ollut käytössä jo yli 60 vuotta. Kanbanissa kysyntä ohjaa tuotantoa, prosessissa työ tai tehtävä haetaan imuohjatusti siten, että vapautuva resurssi hakee tarvitsemansa uuden työn kun edellinen saadaan valmiiksi. Kanban-menetelmä on suunniteltu toimimaan siten, että se rajoittaa prosessissa olevien töiden määrää. Mitä enemmän töitä prosessissa on, sitä hitaampi on töiden virtaus. Menetelmän lähtökohtana on nykyisen toimintatavan asteittainen parantaminen ja virtaviivaistaminen evoluutiomaisesti. Kanban-menetelmää on kuitenkin vasta viime aikoina alettu soveltaa ohjelmistotuotannon prosessimallina. Tutkimusten perusteella menetelmä sopii hyvin ohjelmistotuotantoon, ja on samalla hyvä tapa siirtää Lean-ajattelua organisaatioon. Ikosen tutkimuksen mukaan menetelmä on kuitenkin yksin riittämätön ohjelmistoprojektin kokonaisvaltaiseen hallitsemiseen. (Ikonen 2011, 79 ja 84 - 86: Poppendieck & Poppendieck 2010, 122 - 123.)

Tässä tutkimuksessa lähtökohtana käytetään seuraavien henkilöiden näkemyksiä Kanban-prosessityökalun soveltamisessa ohjelmistotuotantoon: David J. Anderson (2010), Donald G. Reinertsen (2009), Tom ja Mary Poppendieck (2003 – 2013), Henrik Knieberg ja Mattias Skarin (2010), Corey Ladas (2008), Jesper Boeg (2011) ja Marko Ikonen (2011).

4.1 Kanban-menetelmän edut

Kirjallisuuden perusteella Kanban-menetelmän on todettu johtavan seuraavanlaisiin parannuksiin tuotantoprosessissa. Menetelmä parantaa yleisesti työn laatua, nopeuttaa vaatimusten ja töiden läpimenoaikaa, auttaa tunnistamaan ja eliminoimaan prosessin pullonkauloja, vähentää tehtäväjonossa vietettyä aikaa, parantaa tiimityötä, vähentää turhaa puuhastelua ja mahdollistaa prosessin asteittaisen pa-

rantamisen (Kaizen). Lyhyesti sanottuna toiminta on sujuvampaa, läpinäkyvää, ennustettavaa ja keskittyy kokonaisuuteen tunnistamalla ja poistamalla siitä hukkaa. Ketterien menetelmien kanssa Kanban-menetelmän tuoma hyöty on vieläkin suurempi. (Ikonen 2011, 77 - 78, 83.)

Leanin yhteydessä Kanban tarkoittaa korttia tai signaalia, joko edustaa yksittäistä työtä prosessissa. Prosessissa kiertävien korttien lukumäärä vastaa järjestelmän kapasiteettia. Näin Kanban-menetelmä on suunniteltu vetämään imuohjatusti uusi tehtävä työn alle, kun prosessilla on kyky käsitellä se. Kanban menetelmän etuna on se, että oikein määritellyn korttilukumäärän myötä prosessia ei voi ylikuormittaa, verrattuna perinteiseen työntöohjattuun menetelmään. (VersionOne, 2014b)

Ohjelmistotuotannon kannalta Kanban onkin nähty olevan oikea työkalu ohjelmistoprosessin arvovirran virtauksen hallintaan, joka on jäänyt tuotesuunnittelun, riskien hallinnan ja taloudellisten asioiden varjoon. Sillä suuri osa ohjelmistotuotannon ongelmista aiheutuu juuri prosessin arvovirran hallinnasta, joka on jäänyt vähemmälle huomiolle. (Ladas 2008, 9 - 10, 17.)

Ladasin (2008, 52) mielestä ketterien menetelmien ensimmäisen sukupolvi ja ajallisesti rajatut iteraatiot ovat olleet tärkeä välivaihe tunnistamaan se asia, että jatkuva työn virtaus ja imuohjaus (kanban) ovat olleet se asia, jota on tavoiteltu. Hyvin hallitussa imuohjatussa menetelmässä iteraatiot eivät tuota lisäarvoa lainkaan. Virtauksen myötä ohjelmiston jatkuva integraatio (continuous integration) täytyy sovittaa jatkuvaan suunnitteluun (continuous planning) ja jatkuvaan toimittamiseen (continuous deployment). (Ladas 2008, 52 - 53: VersionOne, 2014b.)

Andersonin tutkimusten perusteella Kanban-menetelmä tarjoaa vähän vastustusta aiheuttavan lähestymistavan muuttaa nykyistä prosessia. Menetelmä ei sisällä erillisiä roolituksia, ja se on todettu olevan hyvä tapa tuoda Lean-ajattelua organisaatioon, muokata työkulttuuria ja rohkaista jatkuvan parantamisen ajattelutapaan. Kanbanin on todettu sopivan ketteriä menetelmiä käyttäville tiimeille, sekä myös enemmän perinteistä lähestymistapaa käyttäville tiimeille. (Anderson 2010, 14 - 15.)

Andersonin ensimmäinen Kanban-menetelmän käyttöönotto tapahtui Microsoftilla vuonna 2004. Intiassa ollut ylläpitotiimi suoritti ohjelmistojen virheenkorjausta use-

an eri tuoteomistajan ylläpitämän tehtävälistan kautta. Yksittäisten tehtävien läpimenoaika oli tuolloin noin viisi kuukautta, ja tehtävien määrä jonossa kasvoi kaiken aikaa. Kun tiimin prosessi muutettiin Kanban-menetelmän mukaiseksi, yksittäisen tehtävä keskimääräinen läpimenoaika oli 14 päivää, luvattu toimituspäivä toteutui 98 % varmuudella ja uusien vaatimusten käsittelynopeus kolminkertaistui. Tuottavuus parani yli 200 % aikaisempaan verrattuna. Huomioitavaa on, että kaikki tapahtui ilman muutoksia ohjelmointitapaan ja testausprosessiin. Parannus tapahtui prosessin pullonkaulojen ja hukan eliminoinnilla, sekä tehtäviin liittyvän vaihtelun vähentämisellä, jotka vaikuttivat asiakkaan odotuksiin ja tyytyväisyyteen. Ajallisesti tämä ensimmäinen Kanban-menetelmän toteutus vei 15 kuukautta. (Anderson 2010, 35 - 47.)

Kanban-menetelmää voidaan käyttää myös Scrum-menetelmän kanssa, jolloin puhutaan Scrumban-menetelmästä. Tässä tapauksessa Kanban-taululla käsitellään yksittäiseen iteraatioon valittujen tehtävien virtaus. Kanban-taulu tyhjätään aina iteraatioiden välissä. (Kniberg ja Skarin 2010, 50.)

4.2 Kanban-menetelmän periaatteet ohjelmistotuotannossa

Ohjelmistotuotantoon sovellettuna Kanban-menetelmän periaatteet vaihtelevat hieman esittäjästä riippuen. Alla yhteenveto Kanban-menetelmän periaatteista osittain toteutusjärjestyksen mukaisesti.

1. Tee työnkulku ja eri vaiheet näkyviksi
2. Rajoita työn määrää eri vaiheissa (WIP)
3. Mittaa ja tarkkaile työn etenemistä
4. Tee prosessikäytännöistä havainnollisia (eksplisiittisiä)
5. Käytä valmiita malleja tunnistamaan uusia kehitysmahdollisuuksia
6. Prosessin jatkuva kehittäminen (Kaizen). (Kniberg ja Skarin 2010, 4 - 5 ja 35: Anderson 2010, 15: Boeg 2011, 21 - 35.)

Kolme ensimmäistä periaatetta ovat yleisesti käytössä toteutettaessa Kanban-menetelmää ohjelmistotuotannossa. Periaatteet neljä ja viisi ovat Andersonin tavasta toteuttaa Kanban-menetelmää. Lean-ajattelu ilmenee prosessin jatkuvana kehittämisenä ja parantamisena periaatteen kuusi mukaisesti. (Kniberg ja Skarin 2010, 4 - 5 ja 35: Anderson 2010, 15: Boeg 2011, 21 - 35.)

Seuraavissa kappaleissa Kanban-menetelmä on kuvattu tuotekehityksen näkökulmasta, mutta Kanban-mentelmä skaalautuu myös organisaation ylemmille tasoille. Kanban skaalautuu yksittäisestä tiimistä monen tiimin yhteiseen projektiin, ja edelleen yrityksen projektikokonaisuuksiin (portfolio). (VersionOne 2014b.)

4.2.1 Tee työnkulku näkyväksi

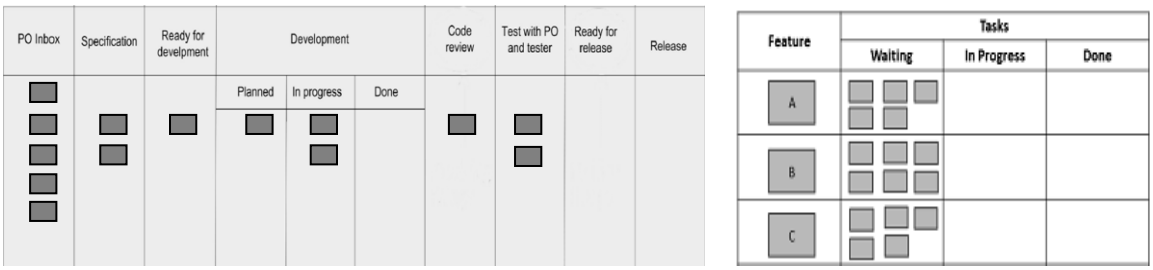
Tee työnkulku näkyväksi -periaatteen tarkoitus on visualisoida työnkulku ja ymmärtää miten nykyinen prosessi toimii. Tässä vaiheessa on tärkeää olla muuttamatta nykyisestä prosessista mitään ja yrittää vain kartoittaa ohjelmistoprosessin työnkulku ja arvovirta. Suosituin tapa prosessin virtauksen esittämiseen on käyttää Lean-työkalua nimeltä arvovirtakuvaus. Menetelmä on tosin saanut arvostelua sopiiko työkalu ohjelmistoprosessin kuvaukseen, joka ei välttämättä ole niin lineaarinen kuin valmistavassa tuotannossa. Tärkeä asia on päättää arvovirtakuvauksen alku ja loppu pisteet ja määritellä näille rajapinta. Rajapinnoissa sovitaan kuinka oma arvovirta vastaanottaa ja luovuttaa työ tai informaation. (Boeg 2011, 21: Kniberg ja Skarin 2010, 67.)

Toinen asia on tunnistaa mitä erityyppisiä töitä arvovirta sisältää (vaatimus, muutospyyntö, virhe, kehityspyyntö). Keskeinen asia prosessin arvovirtauskuvauksessa on miettiä missä muodossa työ tai tieto saapuu sisäisesti eri vaiheisiin. Yleensä arvoa lisäävän toiminnallisen vaiheen nimi, kuten *Toteutus* tai *Testaus*, sisältää paljon muitakin toimenpiteitä kuin nimen mukainen toiminta. Vaiheiden välillä olevaa arvoa lisäämätön toimita on määritelty odotusajaksi (hukka). Vaiheiden määrää kannattaa aluksi selvyyden vuoksi rajoittaa ja lisätä tarvittaessa myöhemmin. Kun vaiheet ovat selvillä, sijoitetaan ne Kanban-taululle omina sarakkeina. Tietyissä tapauksissa työvaiheen sarake voidaan jakaa sisäisesti kahteen tai kolmeen osaan (esimerkiksi Development: Planned, In-Progress, Done). Puskureiden tar-

koitus on tasata työn vaihtelevuutta ja parantaa läpimenoajan ennustettavuutta. Kanban-taulu voi olla fyysinen taulu tai ohjelmallisesti toteutettu työkalu. Kuvatulla arvovirralla pitäisi olla omistaja, joka ratkaisee ilmaantuneet ongelmat. Ongelmia esiintyy yleensä kahden eri arvovirran rajapinnoissa kun molemmat osapuolet optimoivat omaa toimintaansa paikallisesti. Täytyy muistaa, että kokonaisuuden optimointi on tärkeää. (Anderson 2010, 64 - 65: Boeg 2011, 22: Poppendieck & Poppendieck 2007, 84.)

Kanban-taululla olevat kortit edustavat prosessissa olevaa työtehtävää. Korttien tietosisällön suunnittelu kannattaa tehdä hyvin, etenkin jos käytössä on fyysinen Kanban-taulu. Korttien on mahdollistettava imuohjattu toiminta eli tiimin jäsenet voivat itse valita työtehtävän edellisestä vaiheesta. Työtehtävät ovat yleensä kirjattuna sähköisessä tuotekehitysjonossa, joten työtehtävän tunnistetiedot on oltava Kanban-kortilla. Kortin väri tai jokin muu tapa voi ilmaista työn tyypin tai palvelutason. Yleisesti ottaen Kanban-kortti on lupa tehdä jotakin. (Anderson 2010, 70 - 71: Ladas 2008, 23.)

Kuviossa 13 on esitetty ohjelmistotuotantoon sopivia Kanban-taulun malleja. Rakenne voi kuvata yleistä ”vesiputousmallin” mukaista toimintatapaa tai rakenne voi olla myös yksinkertaisempi. Kanban-tauluja voi olla monia erilaisia riippuen prosessista.



Kuvio 13. Kanban-taulu

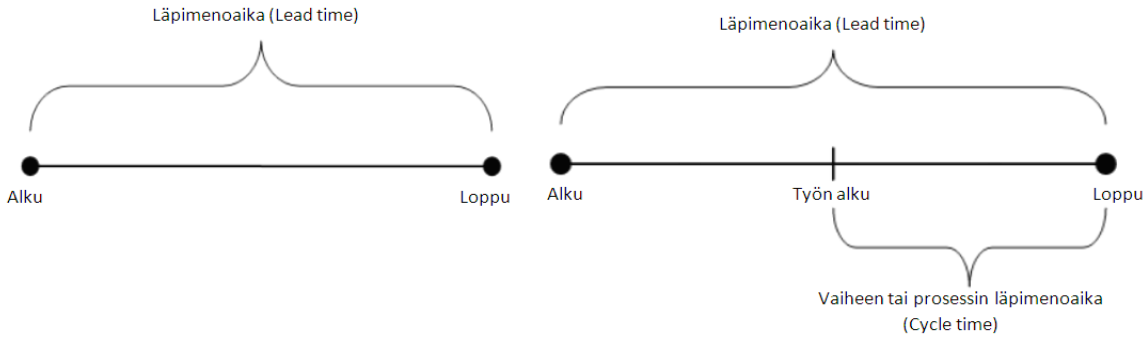
Arvovirtakuvauksen tekeminen yleensä paljastaa prosessista paljon kysymyksiä, epäselvyyksiä ja parannusehdotuksia. Ne liittyvät yleensä töiden läpimenoaikoihin, ajankäyttöön ja prosessissa työn alla olevien työvaiheiden määrään. Nämä ovat asioita, joita Reinertsen (2009, 3) on kirjassaan käsitellyt ja kuvannut. Ongelmien ydin on tuotekehityksessä oleva fyysisesti näkymätön ja mittaamaton jonossa olevien töiden lista, joka aiheuttaa suuren osan tuotekehityksen ongelmista ja vaikut-

taa ennen kaikkea heikentävästi taloudelliseen suorituskykyyn. Jos töiden määrää ei ole rajoitettu, analyysin tulos voi olla kuvan 14 mukainen, missä vaatimusten analysointi ja työmäärien arviointivaiheet ovat määrällisesti suuria.

State			Number of Requirements
Idea!			729
New			897
Being Drafted			416
Ready for Review			422
Ready for Guesstimation			181
Ready for Prioritization	**Analysis**	**2980**	335
Ready for Estimation			68
Ready for Authorization			219
Authorized	**Estimation**	**502**	215
Development Initiated			242
Development Complete	**Development**	**326**	84
Testing (total of all states)	**Testing**	**395**	395
On Hold	**Waiting**	**471**	471

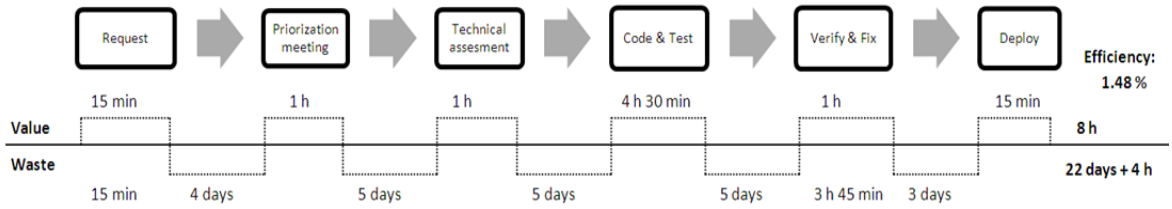
Kuvio 14. Rajoittamaton WIP (töiden määrä eri vaiheissa)
(Black Swan Farming 2013.)

Prosessin tietyn vaiheen läpimenoaika (cycle time) on hidasta johtuen pitkistä odotusajoista. Littlen lain avulla laskettu jonotusaika (kuvio 14) priorisointia odottaville tehtäville (Ready for Prioritization) voi olla useita viikkoja. Jos tehtäviä käsitellään esimerkiksi 15 kpl viikossa, odotusaika on 22 viikkoa (335 / 15 = 22,3 viikkoa). Tehtävän koko prosessin läpimenoaika (lead time) tarkoittaa aikaa, jonka tehtävä tarvitsee asiakastilauksen vastaanottamisesta tuotteen toimittamiseen. (Poppendieck 2007, 100: Reinertsen 2009, 73.)



Kuvio 15. Läpimenoaika vs. vaiheen läpimenoaika

Arvovirtakuvauksesta saadaan selville myös sen tehokkuus. Tämä tapahtuu laskemalla arvovirran kokonaisläpimenoajasta asiakasarvoa tuottava aika. Arvovirran hukka löytyy arvoa tuottamattomista vaiheista, joista asiakas ei maksa mitään. Kuviossa 16 on esitelty eräs ohjelmistotuotannon arvovirtakuvaus ja sen tehokkuus. Kokonaisläpimenoajasta 1,50 % on asiakasarvoa lisäävää työtä ja 98,5 % arvoa lisäämätöntä työtä. Syynä tähän yleensä on tuotekehityksen resurssien maksimaalinen hyödyntäminen ja tuotekehityksen pitkät tehtäväjonot. (Poppendieck 2007, 86 - 87.)



Asiakasarvoa tuottava aika yht. (value):	8 h
Asiakasarvoa tuottamaton aika yht. (waste):	22 d + 4 h (= 532 h)
Prosessin kokonaisläpimenoaika:	540 h
Arvovirran tehokkuus:	8 h / 540 h = 1.48 %
Prosessin WIP (Littlen kaavasta):	540 / 8 = 67 tehtävää

Kuvio 16. Arvovirran tehokkuus

4.2.2 Rajoita työn määrää eri vaiheissa

Rajoita työn määrää eri vaiheissa -periaate on seuraava vaihe työnkulun visualisoinnin jälkeen. Tähän vaiheeseen ei kuitenkaan kannata siirtyä heti, vaan on tärkeää aluksi seurata ja tutustua työnkulkuun, joka on yleensä kovalla työllä saatu näkyväksi. Työn määrän rajoittaminen (jatkossa WIP) kuvaa kuinka paljon työtä koko prosessissa on enimmillään. Kanban-taulussa WIP tarkoittaa kuinka monta tehtävää tai työtä tietyssä vaiheessa voi enimmillään olla yhdellä kertaa. WIP-arvot esitetään yleensä Kanban-taululla olevan vaiheen sarakeotsikossa (kuvio 17). Taulussa olevien WIP-arvojen miettimiseen ja kokeilemiseen kannattaa käyttää aikaa, sillä niihin vaikuttaa tiimin henkilölukumäärä, työskentelytapa ja tehtävien tavoiteltu läpimenoaika. Andersonin (2010, 114) tutkimusten perusteella kaksi teh-

tävää henkilöä kohden olisi optimaalinen arvo, mitä voidaan tietenkin säätää kokemuksen karttuessa.

- Liian pieni WIP arvo → ihmisiä valmiustilassa → huono tuottavuus
- Liian suuri WIP arvo → tehtäviä valmiustilassa → pitkä läpimenoaika. (Kniberg ja Skarin 2010, 47.)

Liian tiukat WIP-arvot rasittavat organisaatiota ja niiden kokonaan puuttuminen olisi prosessin kannalta virhe (Anderson 2010, 119).

Kuvio 17. Kanban-taulu ja WIP-arvot

WIP-arvojen tarkoitus on nopeuttaa tehtävien läpimenoaikaa prosessissa. WIP-käsitteen takana on Littlen laki, joka kuvaa riippuvuussuhteen jonossa olevien asioiden, jonotusajan ja läpimenoajan suhteen. Tuotekehityksen termeille muutettuna kaava olisi: *Läpimenoaika (cycle time) = WIP / keskimääräinen valmistumisnopeus*. Läpimenoaika (keskimääräinen) kertoo ajan mikä työvaiheelta kuluu esimerkiksi Kanban-taulun taululla olevan toteutusvaiheen ja toimitusvaiheen välillä. Kaavan WIP-arvo kuvaa prosessissa olevien töiden lukumäärän. Kaavan avulla voidaan ennustaa prosessin läpimenoajan suoraan jonossa olevien töiden määrän ja töiden käsittelynopeus aikayksikössä perusteella. (Reinertsen 2009, 73 - 74.)

Reinertsen (2009, 111) on kirjassaan painottanut tehtäväjonon koon tarkkailua. Töiden läpimenoaikojen tarkkailun ja kontrolloinnin sijasta tuotekehityksen pitäisi siirtyä tarkkailemaan prosessissa työn alla olevien töiden määrää (WIP). Jonon koon kontrolloinnilla on huomattava merkitys läpimenoaikojen kontrollointiin. Kun töiden määrää rajoitetaan WIP-arvolla, positiivinen vaikutus on keskimääräisen läpimenoajan lyhentyminen. Tästä seuraa kuitenkin kaksi negatiivista vaikutusta,

joudumme hylkäämään mahdollisesti hyviäkin työtehtäviä ja laskemaan kapasiteettia. WIP-arvojen käyttö on kuitenkin tehokasta, sillä ne ovat ilmaisia, voidaan käyttää vaiheittain ja ovat palautettavissa tarvittaessa takaisin. (Reinertsen 2009, 144 - 145 ja 166.)

Esimerkki kuinka nopeuttaa läpimenoaikaa, kun tehtäväjonossa on 100 tehtävää työn alla (WIP) ja nykyinen käsittelykyky on 2 työtä viikossa. Laskettu keskimääräinen työn läpimenoaika on 100 / 2 = 50 viikkoa. Läpimenoajan lyhentäminen 25 viikkoon voi tapahtua kahdella eri tavalla: kaksinkertaistamalla käsittelykyky 4:ään tai puolittamalla tehtäväjonossa olevien työn alla olevien töiden WIP-arvo 50:een. Monissa tapauksissa WIP-arvon laskeminen on tehokkaampaa kuin käsittelykyvyn nostaminen. (Boeg 2011, 27.)

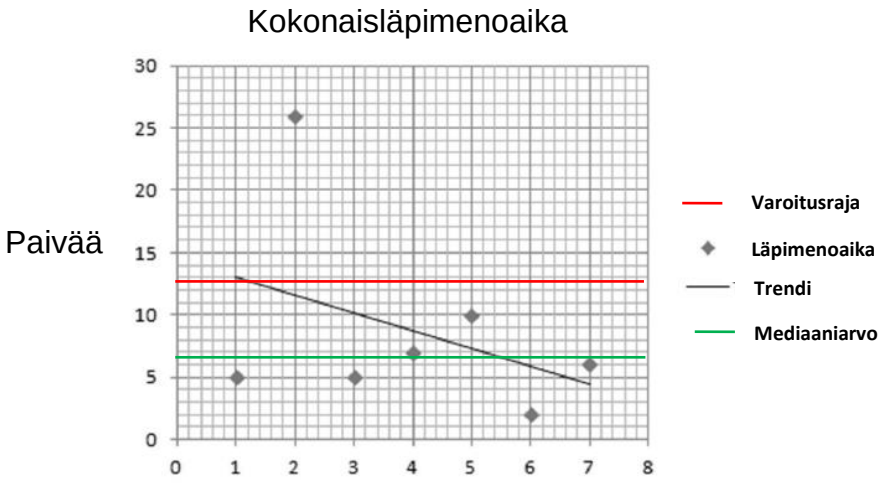
4.2.3 Mittaa ja tarkkaile työn etenemistä

Ohjelmistoprojektin mittaaminen on haastava tehtävä. Perinteisen tavan ohjelmistokehitystä mitataan seuraamalla valmisastetta, toteutuneita tunteja ja toteutuneita kuluja. Painopiste on siis menneisyydessä. Valmiusaste esitetään usein muodossa 60 % määritelty tai 85 % toteutettu, joka ei oikeastaan tarkoita mitään. Kirjallisuuden perusteella ketterien menetelmien ja Lean-menetelmien tarjoaman läpinäkyvyyden myötä ohjelmistokehityksen mittaaminen onkin siirtynyt enempi tulevaisuuden ennustamiseen. Lean-menetelmien myötä kulujen sijasta mitataan ja seurataan tuotettua arvoa (valmiita tehtäviä), sillä tasaisen työvirtauksen ja kiinteän tiimin myötä resurssikulut ovat vakiot. Mittaamisen pitäisi perustua siihen, että tiedämme mikä on ohjelmistoprosessimme kyky tuottaa asiakasarvoa. Tämän mittaustiedon perusteella osaamme ennustaa töiden keskimääräiset läpimenoajat. Yritykset painaa läpi ohjelmistoprosessista enemmän kuin sen kapasiteetti on, johtaa laatuongelmiin, suurempiin ylläpitokuluihin ja ylitöiden lisääntymiseen. (Poppendieck & Poppendieck 2010, 128 - 140: Anderson 2010, 139 - 147.)

Poppendieckin (2010, 14) mukaan oman organisaation suorituskyvyn mittaaminen pitää perustua tietynmittaiseen ajanjaksoon, ei muutamiin sattumanvaraisiin pisteisiin, sillä jokaisessa systeemissä ilmenee vaihtelevuutta. Hyvä tapa visualisoida ohjelmistoprosessin kapasiteetti on luoda aikajaksokaavio. Kaavioon merkitään

kuinka monta päivää työn tekeminen kesti, kun se saapui järjestelmään ja toimitettiin asiakkaalle. Kun näin on tehty pidemmän aikaa, saadaan selville työ keskimääräinen läpimenoaika. (Poppendieck & Poppendieck 2010, 13 - 15.)

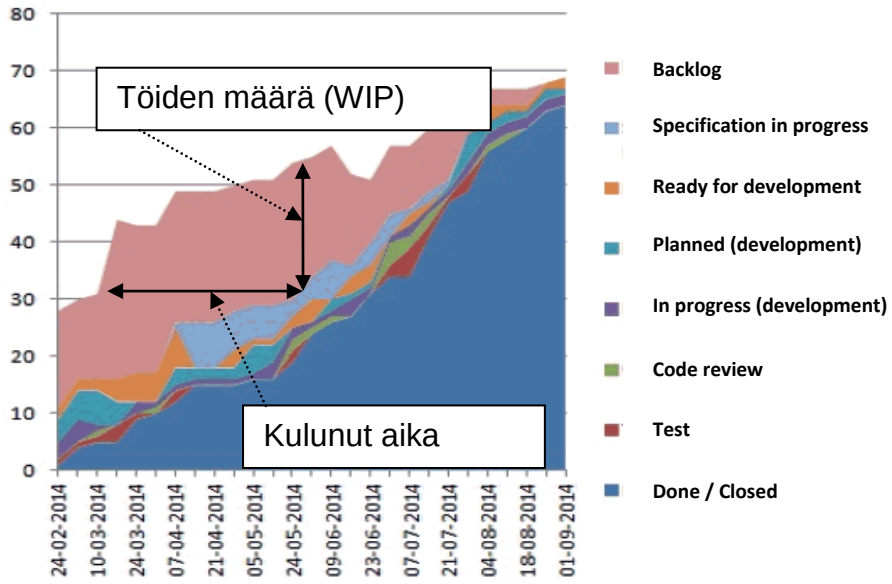
Työn kokonaisläpimenoaika (kuvio 18) on usein huomattavasti pidempi kuin se aika, mitä työn tekemiseen käytettiin. Läpimenoaikojen pitäisi olla mediaaniarvon molemmilla puolilla. Kun tämä tieto on visualisoitu, läpimenoaikojen lyhentäminen tulee huomioitua paremmin. Kokonaisläpimenoaikojen tarkkailu kertoo myös prosessin tilasta, jos keskimääräiset läpimenoajat alkavat lähestyä määriteltyä varoitusrajaa, on johdon puututtava asiaan. Mediaaniarvo ja varoitusraja ovat tutkimuksen tekijän omia lisäyksiä. Kuten aikaisemmin tässä tutkimuksessa mainittiin, Lean-johtamisessa mittareita ei ensisijaisesti käytetä ihmisten valvontaan ja kontrollointiin, vaan mittarit ovat tavoitteita ja työkaluja joilla tiimit ja työntekijät voivat arvioida omaa suorituskykyään. Tutkimuksen kannalta tämä on hyvä mittari, jolla arvioidaan tiimin tai organisaation suorituskykyä. (Poppendieck & Poppendieck 2007, 13 - 15: Liker & Convis 2012, 203 - 205.)



Kuvio 18. Työn keskimääräinen läpimenoaika

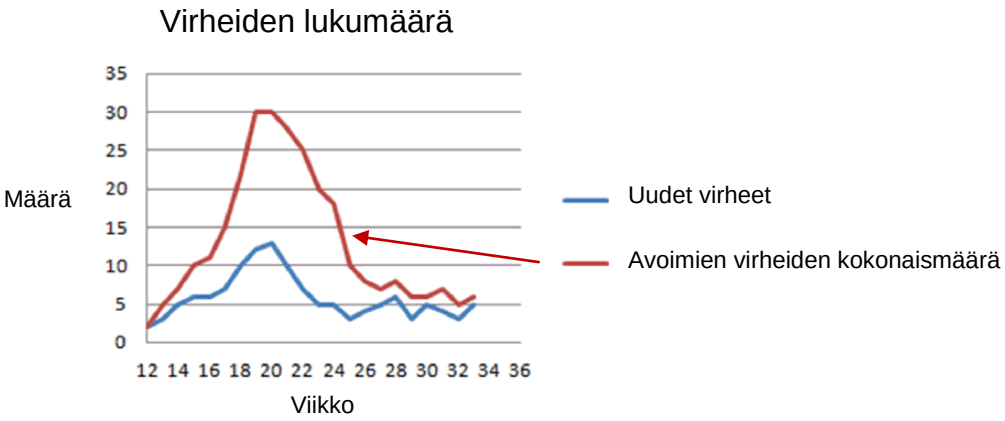
Kuvion 19 mukainen kumulatiivinen virtauskaavio (cumulative flow diagram, CFD) kuvaa koko prosessin tilaa esittämällä jokaisen vaiheen työn määrän (WIP) tiettynä ajan hetkenä. Anderson (2010, 140) ja Reinertsen (2009, 71) ovat tutkimuksissaan todenneet, että kumulatiivinen virtauskaavio on hyvä tapa seurata Kanban-menetelmän toimintaa. Teoriassa virtauskaavio on hyvä työkalu esittämään virta-

uksen pullonkaulat, ja myös sen kuinka WIP-arvo vaikuttaa läpimenoaikaan. Käytännössä virtauskaavion tulkinta voi olla hieman vaikeaa (Kniberg, 2011, 63).



Kuvio 19. Kumulatiivinen virtauskaavio

Laatukysymykset ovat kalliita, joten ohjelmistossa olevien virheiden seuraaminen on tärkeää. Tässäkin tapauksessa ohjelmistovirheiden (kuvio 20) lukumäärän näkyväksi tekeminen auttaa arvioimaan omaa suorituskykyä.



Kuvio 20. Ohjelmistovirheiden lukumäärä

4.2.4 Tee prosessikäytännöistä havainnollisia

Andersonin (2010, 137) mukaan prosessikäytäntöjen tekeminen näkyväksi tarkoittaa erilaisia laatua ja prosessia edistävien asioiden kuvaamista Kanban-taululla. Työtyyppien ja palveluluokkien havainnollistaminen tapahtuu yleensä väreillä tai käyttämällä Kanban-taulussa ”uimaratoja”. Taululle ja prosessin eri vaiheisiin voidaan myös kirjoittaa tarkentavia tietoja. Vakaville ohjelmistovirheille Kanban-taululle voidaan kuvata kokonaan oma kiireellinen kaista. Prosessikäytäntöjen näkyvillä oleminen auttaa Kanban-taulun tulkitsemisessa ja parannusehdotusten löytämisessä. (Anderson 2010, 123 - 137.)

4.2.5 Käytä valmiita malleja tunnistamaan uusia kehitysmahdollisuuksia

Andersonin (2010, 188) mukaan prosessin ja toiminnan kehittämiseen kannattaa käyttää valmiita malleja, kuten kapeikkoteoriaa (Theory of Constraints), jonoteoriaa ja Littlen lakia. Edellä mainitut mallit ovat vain esimerkkejä, joiden avulla keskitytään prosessin pullonkaulojen, hukan ja vaihtelevuuden hallintaan. Erilaisten mallien avulla Kanban-menetelmä voidaan sovittaa yritykselle parhaiten sopivaksi. Kirjassaan Anderson (2010) on kuvannut kuusi askelta menestykseen, jotka Kanban-menetelmä mahdollistaa teoriamallien avustuksella: keskity laatuun, vähennä töiden määrää prosessissa (WIP), toimita usein, pidä töiden kysyntä ja läpimeno tasapainossa, priorisoi, poista prosessista vaihtelevuutta ja paranna prosessin ennustettavuutta (Anderson 2010, 22).

4.2.6 Prosessin jatkuva kehittäminen

Empiirisenä menetelmänä Kanban tukee Lean-periaatteiden mukaisesti prosessin jatkuvaa kehittämistä (Kaizen-kulttuuri). Kun prosessi on Kanban-menetelmän myötä tehty visuaaliseksi ja läpinäkyväksi, nähdään prosessin ongelmat yhdellä silmäyksellä ja niihin voidaan puuttua aikaisempaa nopeammin. Prosessista ja arvovirrasta mitattavien asioiden näkyväksi tekeminen auttaa arvioimaan omaa suorituskykyä. Edellä mainituilla asioilla prosessi vaikuttaa positiivisesti tuotteen laatuun. (Anderson 2010, 49 - 51, 60, 139).

Andersonin (2010, 188 - 189) mukaan prosessia pitäisi kehittää löytämällä siitä vaiheita, jotka hidastavat ja kuormittavat prosessia. WIP-rajoja säätämällä Kanban-menetelmä ilmaisee nopeasti kun tietty prosessin vaihe kuormittuu, tällöin Kanban-taulun myöhäisempiin sarakkeisiin syntyy tyhjiö. Lean-ajattelun mukaan prosessin pullonkaulana oleva ongelma pyritään ratkaisemaan nopeasti. Ratkaisun taustalla voi olla jokin teoriamalli, kuten kapeikkoteoria (Theory of Constraints). *Eliyah Glodratt:n* kirjassa *The Goal* on esittänyt viisivaiheisen mallin prosessin kehittämiseen:

1. Tunnista haaste tai rajoittava tekijä
2. Suunnittele ratkaisu haasteeseen
3. Suunnittele ratkaisu huomioiden muu organisaatio
4. Suorita tarvittavat toimenpiteet
5. Stabiloi ratkaistun haasteen tila ja aloita tarvittaessa uusi kierros.

Prosessia pyritään kehittämää inkrementaalisesti pienin askelin organisaatiolle sopivaksi. (Anderson 2010, 187 - 193).

Ongelman selvittämiseen voidaan käyttää myös W. Edwards Demingin PDCA-ongelmanratkaisumallia eli PDCA-kehityssyklejä (plan-do-check-act, suunnittele, tee tarkista, toimi/korjaa). (Poppendieck & Poppendieck 2007, 154 - 155).

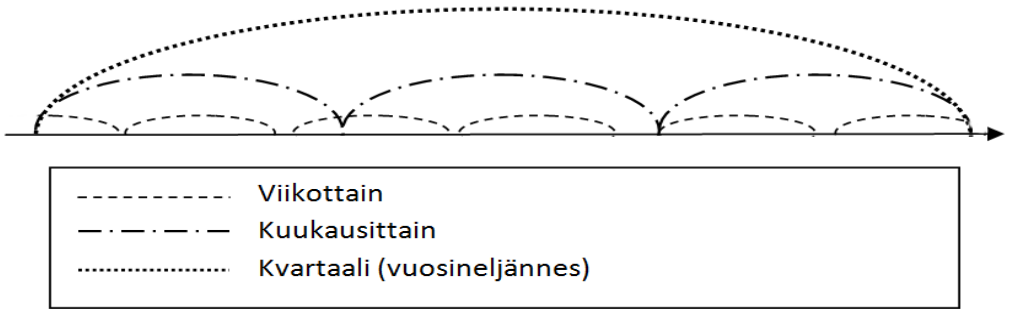
Mikään ei estä myöskään käyttämästä lean-työkalua 5-whys. Ongelmatilanteessa juurisyyn löytämiseksi kysytään viisi kertaa ’miksi?’. (Liker 2004, 252).

Perinteisessä toimintatavassa prosessissa esiintyneen ”pullonkaulan” sattuessa helposti kysytään ’kuka’ eikä ’miksi’, ja työntekijälle mahdollisesti tarjotaan toinen työ tai siirretään muihin tehtäviin. Prosessissa olevaa ongelmaa ei korjata, vaan vapautuva resurssi ja kapasiteetti täytetään uudella työllä.

4.3 Kanban ja ohjelmistotuotannon syklit

Ohjelmistotuotannon syklit eli iteraatiot rytmittävät toimintaa. Syklit ovat yrityksen liiketoiminta-arkkitehtuuriin perustuvia, ja ne lähtevät liikkeelle yritysjohdon määrittelemistä liiketoiminnallisista tavoitteista, joita ohjelmistotuotanto tukee mahdollisuuksien mukaan (Haikala ja Mikkonen 2011, 32). Esimerkkejä liiketoiminnallisten tavoitteiden luomista ja asiakkaille näkyvistä ulkoisista sykleistä ovat: uusi versio tuotteesta vuoden välein, uusi tuotejulkaisu 3 kuukauden välein ja tuotteen huoltopäivitys kuukauden välein. Tärkeintä kalenterirytmityksessä on se, että syklien aikataulun täytyy pitää luoden ennustettavuutta ja luottamusta asiakkaiden näkökulmasta. Tasainen rytmitys vähentää myös tuotteen toimitukseen liittyvää hallintaa ja kustannuksia, sillä lukuisa joukko ihmisiä tietää milloin toimitus tapahtuu. Iteraatioiden aikana tuotteeseen tehtävien asioiden laajuus ja sisältö joustavat tilanteen mukaan. (Anderson 2010, 101.)

Ohjelmistotuotannossa on useita prosessin sisäisiä syklejä. Lean-menetelmien ja etenkin Kanban-menetelmän myötä töiden tasaisen virtauksen kannalta on tärkeää rytmittää prosessiin tulevien ja siitä lähtevien töiden syklit. Kanban-menetelmä ei ole este ohjelmistotuotannon erilaisille, yhtäaikaa käynnissä oleville kalenterirytmityksille. Uusia ominaisuuksia voidaan toimittaa asiakkaille kuukauden välein, kuten ennenkin. Tehtäväjonossa olevia sisääntulevia töitä voidaan priorisoida ja arvioida kerran viikossa, kuten ennenkin. Kanban-menetelmän myötä sisääntulevien töiden ottaminen ja priorisointi voidaan laukaista Kanban-taululla ilmenneen kysynnän mukaan (taulun ”Inbox”-sarakkeessa on 3 tyhjää paikkaa). Samalla periaatteella myös uusien tehtävien työmäärän arviointipalaveri voidaan käynnistää, jos Kanban-taulun ”Inbox”-sarakkeeseen on ilmestynyt kolme uutta Kanban-korttia eli tehtävää. Tällaiset Kanban-taululla ilmenneen kysynnän mukaan laukaistavat tapahtumat vaativat tiimiltä kuitenkin vahvaa kokemusta ja taitoa. (Poppendieck & Poppendieck 2010, 126 - 129: Boeg 2011, 38.)



Kuvio 21. Ohjelmistotuotannon syklit

Prosessin sisäiset erityyppiset syklit voivat olla hyvinkin nopeita. Päivittäin pidetään töiden tilaan ja etenemiseen liittyvä aamupalaveri. Versionhallintatyökaluun yhdistetty jatkuvan integraation työkalu rakentaa toimitettavissa olevia sovelluksia jopa muutaman tunnin välein. Lean-menetelmien kannalta ulkoisten ja sisäisten syklien suuret ajalliset erot voivat aiheuttaa ongelmia. Ominaisuuteen liittyvän tärkeän palautteen saaminen asiakkailta voi tietyissä tapauksissa viedä jopa kuukauden.

Kun sylklien pituudet ovat kunnossa, työt virtaavat tasaisesti. Tasainen rytmitys rajoittaa prosessissa esiintyvää vaihtelevuutta. Toimitussyklin pituus on yleensä liian pitkä, jos syklin lopussa esiintyy huomattava määrä ylimääräistä toimintaa (suuri eräkoko). Lyhyempi toimitussykli auttaa tasaamaan työkuormaa, koska toimitettavien ominaisuuksien eräkoko on automaattisesti pienempi, ja samalla vaihtelevuus pienenee. Kapasiteettiin on kuitenkin jätettävä varaa, jotta vaihtelevuus tehtävien suoritusajoissa ei aiheuta myöhästymistä syklin päättymispäivän osalta. (Reinertsen 2009, 176 - 181).

Lean-ajattelussa ominaisuuksien priorisointi perustuu asiakasarvoon tuottaen asiakkaille mahdollisimman suuren hyödyn. Syklit täyttyvät tuoteomistajan luoman vision ja etenemissuunnitelman mukaisista ominaisuuksista. Lean-menetelmissä ja ketterissä menetelmissä tuotteen kehitysjono on priorisoitu lista tehtävistä, jonka kärkipäässä ovat tuotteen asiakasarvon ja teknisen arvon kannalta tärkeimmät ominaisuudet (Pichler 2010, 49). Kehitysjonon tehtävälistalla kannattaa olla työmäärältään erikokoisia tehtäviä, sillä tämä tasaa tehtävien virtausta. Priorisointivaiheessa määritellään myös missä suhteessa arvoltaan erityyppisiä tehtäviä kehitysjonoon laitetaan. Kapasiteetista esimerkiksi 20 % varataan ylläpitoon tai vir-

heenkorjaukseen, 30 % muutospyyntöihin ja 50 % uusia ominaisuuksia varten. (Boeg 2011, 57: Anderson 2010, 134 - 135.)

Tehtävät kehitysjonossa kuitenkin muuttuvat ja kehittyvät. Tuoteomistajan rooli on Lean-menetelmien myötä haastava. Ketterien menetelmien kanssa asiakkaan vaatimusten ja ominaisuuksien prioisointi ja kuvaaminen tapahtuu yleensä iteraatioiden välissä. Kanban-menetelmän ja töiden jatkuvan virtauksen myötä, myös tehtävälistan täyttäminen ja priorisointi on sykliltään nopeampaa. Kanban-taulu ei tyhjene iteraatioiden välissä (Kniberg ja Skarin 2010, 50). Rytmitetty priorisointitapahtuma voidaan myös poistaa, jolloin tapahtuman laukaisu on synkronoitu Kanban-taululla ilmenneeseen kysyntään. Tehtävälistan täyttämiseen ja priorisointiin kuitenkin suositellaan omaa kalenterirytmiä (viikoittainen tapahtuma). Virtauksen kannalta erilaisten asioiden synkronointi ja hallinta tiimien välillä on tärkeää. Reinertsenin (2009, 178) mukaan synkronointi tarvitsee myös ”löysää” kapasiteettiin.

4.4 Pohdintaa Kanban-mentelmästä

Lean-ajatteluun perustuvaa Kanban-työkalua on alettu viime vuosina soveltaa ohjelmistotuotannon prosessimallina. VersionOnen (2014a) tutkimuksessa Kanban tyyppisten menetelmien osuus on jo noin 12 %, ja niitä käytetään pääasiassa organisaation sisäisissä prosesseissa. Tutkimusten perusteella menetelmä sopii hyvin ohjelmistotuotantoon, ja on samalla hyvä tapa siirtää Lean-ajattelua organisaatioon. Menetelmänä Kanban ei tarvitse mitään erillistä roolitusta (kuten Scrum-menetelmä). Kokeilemaan pääsee nopeasti ja edullisesti rakentamalla valkotaululle oma Kanban-taulu käyttämällä erivärisiä post-it lappuja. Ohjeistukseksi ja periaatteiksi tarvitaan minimissään vain kolme asiaa:

- tee työnkulku ja eri vaiheet näkyviksi
- rajoita työn määrää eri vaiheissa
- mittaa ja tarkkaile työn etenemistä.

Näillä periaatteilla Kanban visualisoi työprosessin, tukee tilannekohtaista analysointia ja edistää jatkuvaa yhteistyötä. Prosessin tila ongelmineen ja pullon-

kauloineen nähdään yhdellä silmäyksellä. Visuaalisuus kannustaa jatkuvaan oppimiseen ja parantamiseen pyrkimällä löytää paras mahdollinen tapa töiden virtaukselle.

Kevyestä ohjeistuksesta huolimatta ihmisten asenteiden muuttaminen saattaa olla haastavaa. VersionOnen (2014a) tutkimuksissa ketterien menetelmien laajemmalle käytölle koettiin seuraavia esteitä: organisaation jäykkä kulttuuri, yleinen muutosvastarinta ja ketterien menetelmien sovittaminen perinteisiin menetelmiin. Voidaan kuvitella kuinka vesiputousmalli tekee paluuta, kun Kanban-mentelmän kanssa kohdataan ensimmäinen ongelmallinen tilanne. Lean-menetelmät ja ketterät mentelmät vaativat onnistukseen hyvää johtamista, muuten lopputuloksena on kaaos. Hyvällä johtamisella ei tarkoiteta perinteistä käskyttämiseen perustuvaa johtamistapaa, vaan enemmänkin valmentavaa, ohjaavaa ja prosessissa ilmenneiden esteiden poistamista (Poppendieck & Poppendieck 2010, 183).

Tutkimusten mukaan Kanban-menetelmällä ohjelmistokehitys on nopeampaa ja työn tuottavuus on suurempaa verrattuna Scrum-menetelmään (Helsingin Yliopisto 2014). Tämä siksi, että Kanban-menetelmän taustalla on paljon toimivaa teoriaa. Jonoteoria, kapeikkoteoria ja Littlen laki ovat lakeja, joita systeemit ja prosessit noudattavat. Näiden teorioiden ymmärtäminen auttaa prosessin ongelmakohtien (pullonkaulojen) ratkaisemisessa ja jonojen purkamisessa huomattavasti paremmin kuin tiimin ”tsemppaaminen”.

Kanban-menetelmä ei siis tarjoa suoraa apua ongelmien ratkaisemiseen, vaan tuo ne näkyvästi esille. Lisäksi Kanban tarjoaa asteittaisen kehityspolun ketteryyteen sellaisille organisaatioille, jotka eivät ole ketteriä menetelmiä aikaisemmin kokeilleet. Kanban-menetelmä on hyödyllinen myös silloin, kun aikarajatuista iteraatioista ja pyrähdyksistä ei ole hyötyä. Näitä ovat esimerkiksi tukipalvelut, joissa tehtäviä tulee työn alle satunnaisesti. Menetelmällä on taipumus myös levitä muille osastoille, mikä lisää toiminnan läpinäkyvyyttä organisaatiossa. (Crisp 2014.)

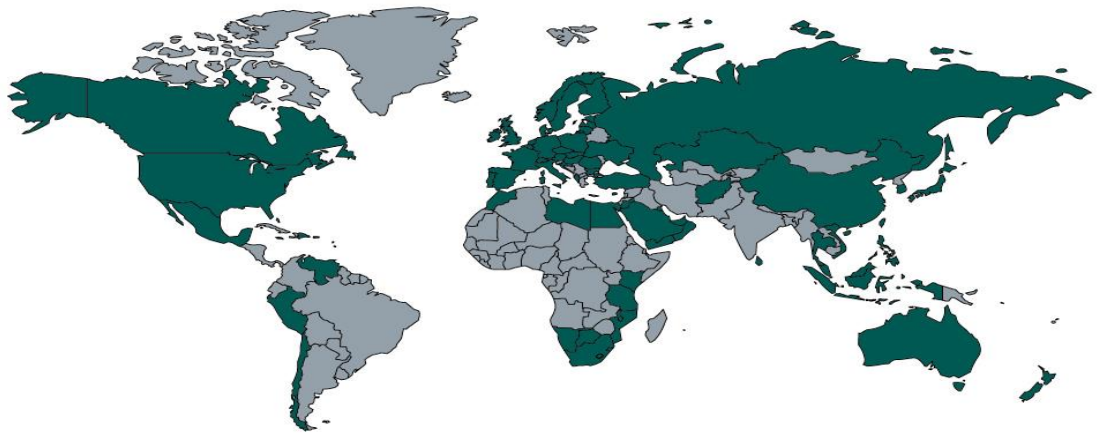
Tutkimuksen kannalta Reinertseinin (2009) kuvaus Lean-tuotekehityksestä kirjassa *The Principles of Product Development Flow: Second generation Lead Product Development* esitti Kanbanin ja töiden virtauksen taustalla olevan teoria hyvin.

Kaikesta huolimatta kanban on yksin riittämätön ohjelmistoprojektin kokonaisvaltaiseen hallitsemiseen.

5 KOHDEORGANISAATION NYKYINEN TOIMINTAMALLI

5.1 Toimintaympäristön esittely

Tutkimuksen oma kohdeorganisaatio on alun perin perustettu 1983. Yritysostojen kautta se on osa maailman suurinta toimijaa oma kapean alansa (niche) ohjelmistotuottajana. Emoyhtiö, jonka osana kohdeorganisaatio on, perustettiin vuonna 1964. Asiakkaina on maailmanlaajuisesti 25000 erillistä jälleenmyyjä- ja maahantuojaorganisaatiota. Emoyhtiö on maailman 300 suurimman yrityksen listalla. Yrityksen päätuotteita toiminnanohjausjärjestelmät jälleenmyyjille ja maahantuojille (ERP). Kohdeorganisaation oma ohjelmistotuote sisältää sekä jälleenmyyjän että maahantuojan osat. Tuotteet ovat osa emoyrityksen monista toiminnanohjausjärjestelmistä.



Kuvio 22. Toiminnan kattavuus maailmalla

Yrityksen missiossa ja visiossa painotetaan asiakkaiden menestystä, ja pyrkimystä auttaa asiakkaita hyvällä paikallistuntemuksella, vaikka kyse on suuresta globaalista toimijasta. Tarkoituksena on auttaa asiakkaita vähentämään manuaalisia prosesseja, virtaviivaistamaan heidän liiketoimintaa ja saavuttamaan mittavia tuloksia liiketoiminnan kaikilla osaalueilla. Tutkimuksen kannalta tämä kuulostaa hyvinkin Leanilta.

Toimiala on yleensä suhdanneherkkää ja talouden alamäki vaikuttaa nopeasti asiakkaiden myyntiin, ja täten myös rauhoittaa uusien tietojärjestelmien hankkimista

ja räätälöintiä. Toisaalta tietojärjestelmän uusiminen ja räätälöinti voi olla edullista talouden alamäessä, ja tuoda näin jälleenmyyjälle tulevaisuudessa kustannusetua uusien ominaisuuksien myötä. Tiettyjen maahantuojien päämiehet pyrkivät konsolidoimaan järjestelmiään, joten erilaisten tehdasliittymien määrä tulee kasvamaan, joka on hyvä trendi tietojärjestelmätoimittajalle.

Tekniseltä kannalta kohdeorganisaation tuotteet ovat olleet aikansa johtavia tuotteita omalla alallaan. Yksi ensimmäisistä Pc-pohjaisista järjestelmistä ilmestyi vuonna 1983, ja yksi ensimmäisistä Windows-pohjaisista järjestelmistä vuonna 1994. Windows-pohjaisen järjestelmän lähdekoodi on toistaiseksi pystytty migraatioiden kautta siirtämään uusille kehitysalustoille.

Nykyinen internet perusteinen trendi on haastava, sillä pitkään kehitetyt tuotteet eivät muutu nopeasti uusimpien muotivirtausten mukana. Teknologia kuitenkin normalisoituu, ja jos asiakas odottaa 1 – 3 vuotta, niin hänellä on tarvittava uudet ominaisuudet nykyisessä järjestelmässään.

Koska emoyhtiö on globaali toimija, yhtiön organisaatiorakenne on hyvin hierarkinen. Tämä vaikuttaa siihen, että muutokset toimintatavoissa tapahtuvat kohtalaisen hitaasti.

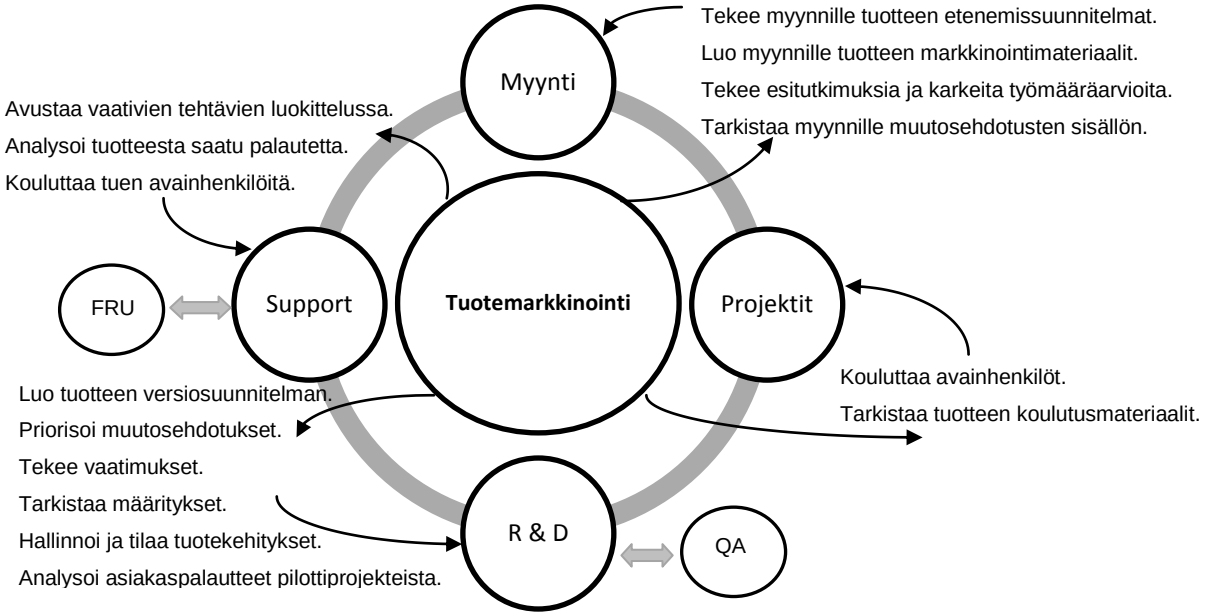
5.2 Toimintatavat pääpiirteissään

Kohdeorganisaation liiketoiminnan strategia tehdään johdon toimesta aina loppusyksystä (marraskuussa) viideksi vuodeksi eteenpäin. Kyseessä on rullaava prosessi eli yksi vuosi aina eteenpäin. Seuraavan keväänä strategiasta otetaan ensimmäinen vuosi käsittelyyn, mistä hiotaan seuraavan vuoden budjetti ja tarkat suunnitelmat. Strategian syvällisin mietintä tapahtuu emoyhtiön pääpaikoilla USA:ssa ja Englannissa.

Yritysjohdon tekemä budjetointi tukee ja kohdentaa liiketoiminnan strategiaa operatiivisesti kohdentamalla tuotekehityksen tuotantokapasiteetin ja resurssit valituille painopistealueille. Kehittämistavoitteet asetetaan tuotemarkkinointi-osaston toimesta lyhyelle ja pitkälle aikavälille. Lyhyen aikavälin suunnitelmaa kutsutaan tuotteen versiosuunnitelmaksi ja pitkän aikavälin suunnitelma on tuotteen tiekartta

(roadmap). Budjetti sisältää asiakasrahoitteisia ominaisuuksia, tuotemarkkinointiosaston itse rahoittamia kehittämiskohteita ja lakimuutosten vaatimia muutoksia. Koska kyseessä on jatkuvasti kehitettävä pitkän elinkaaren tuote, budjetista 25 % on varattu tuotteen ylläpitoon (maintenance). Kehittämistavoitteita tarkistetaan markkinatilanteen perusteella aina neljännesvuosittain.

Tuotteeseen liittyvää toimintaa ohjaa erillinen tuotemarkkinoinnin johtajasta ja tuoteomistajista (product owner) koostuva tuotemarkkinointiosasto, jonka tehtävänä on tukea ja ohjata kohdeorganisaation muita osastoja kuviossa 23 esitetyllä tavalla.



Kuvio 23. Tuotemarkkinointiosaston tehtävät

Tuotemarkkinointiosasto tekee käytännössä kaikki päätökset liittyen tuotteeseen tehtävien hankkeiden elinkaareen, ideasta tuotantokäyttöön ja tuotantokäytöstä poistamiseen. Tuotemarkkinointi tilaa yksin tai yhdessä myynnin kanssa tuotteeseen toteutettavat ominaisuudet tuotekehitysosastolta. Tuotekehitysosaston tehtävä on ennen tilausta laatia karkea työmääräarvio tuotemarkkinoinnin pyytämistä ominaisuuksista. Saadun työmääräarvion pohjalta ominaisuus tarjotaan asiakkaalle. Jos tarjous hyväksytään, tilaus siirretään tuotekehitysjonoon. Hyvä ominaisuus voidaan rahoittaa myös tuotemarkkinointiosaton omasta toimesta. Ominaisuuden tiedot kirjataan (dokumentit linkitetään) erilliseen tehtävänhallintajärjestelmään, jonka portti-mallin mukainen toimintatapa tilakoodeineen (stage-gate) määrittelee

samalla organisaation työprosessin. Kaikki organisaation dokumentit on tallennettu SharePoint-sisällönhallintajärjestelmään. Tuotemarkkinointiosaston tehtäväkartta on varsin laaja, haastava ja tarkkaan ohjaukseen perustuva.

Tuotekehitysosasto on jaettu muutamaan erilliseen tiimiin ja tiiminvetäjään, joita johtaa tuotannon esimies. Tiimien toimintatapa muistuttaa kuitenkin henkilöistä koostuvaa ryhmää, sillä tiimin jäsenillä on yleensä vastuullaan oma kehitettävä ominaisuus. Tiimiin kuuluu myös oma testaaja. Tiiminvetäjä osallistuu myös tuotekehitykseen. Tuotekehitysosaston tapa tehdä töitä perustuu perinteiseen vesiputousmalliin, vaikkakin orastavaa ketteryyttä tiimeihin on saatu aikaiseksi ”daily scrum”-aamupalaverin muodossa. Tiimeillä ei toistaiseksi ole käytössä muuta Scrum-menetelmään liittyvää toimintaa, kuten roolitusta, pyrähdyksiä, itseohjaustuvuutta ja retrospektiivipalavereja (jälkipuintipalaveri). Ketteryyshanke on käynnistetty emoyhtiön toimesta.

Koska organisaation ja tiimien toimintatapa rakentuu vesiputousmalliin, kaikki pyritään suunnittelmaan etupainotteisesti tarkasti ja hyvin. Tiiminvetäjä kommunikoi tuotemarkkinointiosaston kanssa ja jakaa saamansa tehtävät tiimin jäsenille. Ominaisuudet, joiden koko ja laajuus vaihtelee hyvinkin suuresti, on yleensä alustavasti merkattu tietylle tiimin jäsenelle (luo kriittisen polun). Asiakkaan kanssa rakennetut esitutkimukset ja vaatimukset saadaan siis tuotemarkkinointiosastolta, joiden pohjalta tuotekehitys tekee määritykset ja tarkentaa toteutuksen työmääräarviota. Määrityksen hyväksymisen jälkeen tilattu ominaisuus toteutetaan, ja muutokset viedään versionhallintaan. Toteutuksen koodimuutokset katselmoidaan yleensä toisella tiimiin kuuluvan jäsenen toimesta.

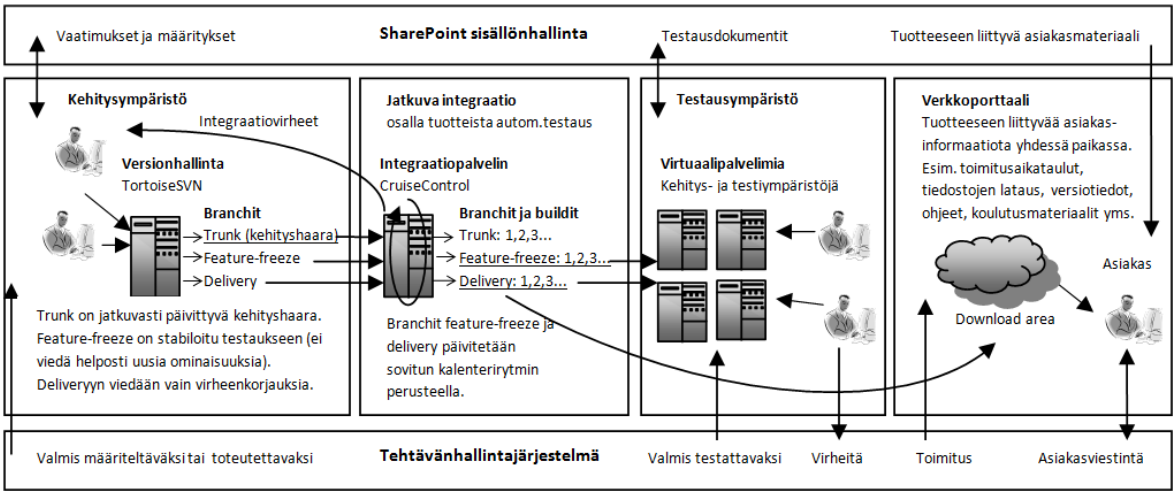
Testausosasto (QA) testaa toteutuksen perustuen määrityksiin ja osittain vaatimuksiin. Testausympäristönä (myös kehitysympäristönä) toimii yleensä virtuaalipalvelin, joka vastaa todellista asiakasympäristöä. Testaukseen liittyvät toimenpiteet suoritetaan yleensä versionhallinnassa olevaa stabiilia ”feature-freeze”-branchia vasten, joka on suljettu uusilta ominaisuuksilta. Testausosaston toiminta on rytmitetty tuotteen regressiotestiin (testataan aiemmin toteutetut ominaisuudet), ja uusien ohjelmistomuutosten testaamiseen. Sisäisen testauksen jälkeen ominaisuus menee asiakkaalle testaukseen. Lopullinen päätös toteutuksen hyväksymi-

selle on asiakkaan kanssa yhdessä suoritettava testaus eli UAT (user acceptance test). Ominaisuus toimitetaan asiakkaalle sovitun buildin yhteydessä.

Vesiputousmallissa hyvin suunniteltu edellinen vaihe takaa onnistumisen. Mutta jos lopputulos ei ollut onnistunut, syy on aina suunnittelussa tai sen toteutuksessa. Muutos alkuperäiseen vaatimukseen käsitellään erillisen muutoskäsittelyn kautta.

Vesiputousmalli on kuitenkin toiminut organisaatiossa kohtuullisen hyvin, sillä tuotetta on tehty pitkään eri asiakkaiden kanssa, jopa yli 10 vuotta. Toiminnalla on eräänlainen vakiintunut standardi. Osapuolet tietää mitä tehdään ja miten asia pitää toteuttaa, joten asioita voidaan määritellä etupainotteisesti. Kooltaan suurien tai kokonaan asiakkaan tuotantoprosessin läpi vaikuttavien ominaisuuksien toteuttamiseen etupainotteinen toimintatapa tuottaa haasteita. Organisaation sisäinen prosessi on ollut pitkään käytössä ja se on kaikille tuttu. Tämä kertoo sen, että kyseessä on kokenut ja positiivisessä mielessä kypsä organisaatio. Työntekijöiden keskimääräinen aika organisaation palveluksessa on nykyään 12 vuotta.

Kohdeorganisaation työympäristö ja työkalut on esitelty alla olevassa kuviossa.



Kuvio 24. Kohdeorganisaation työympäristö ja työkalut

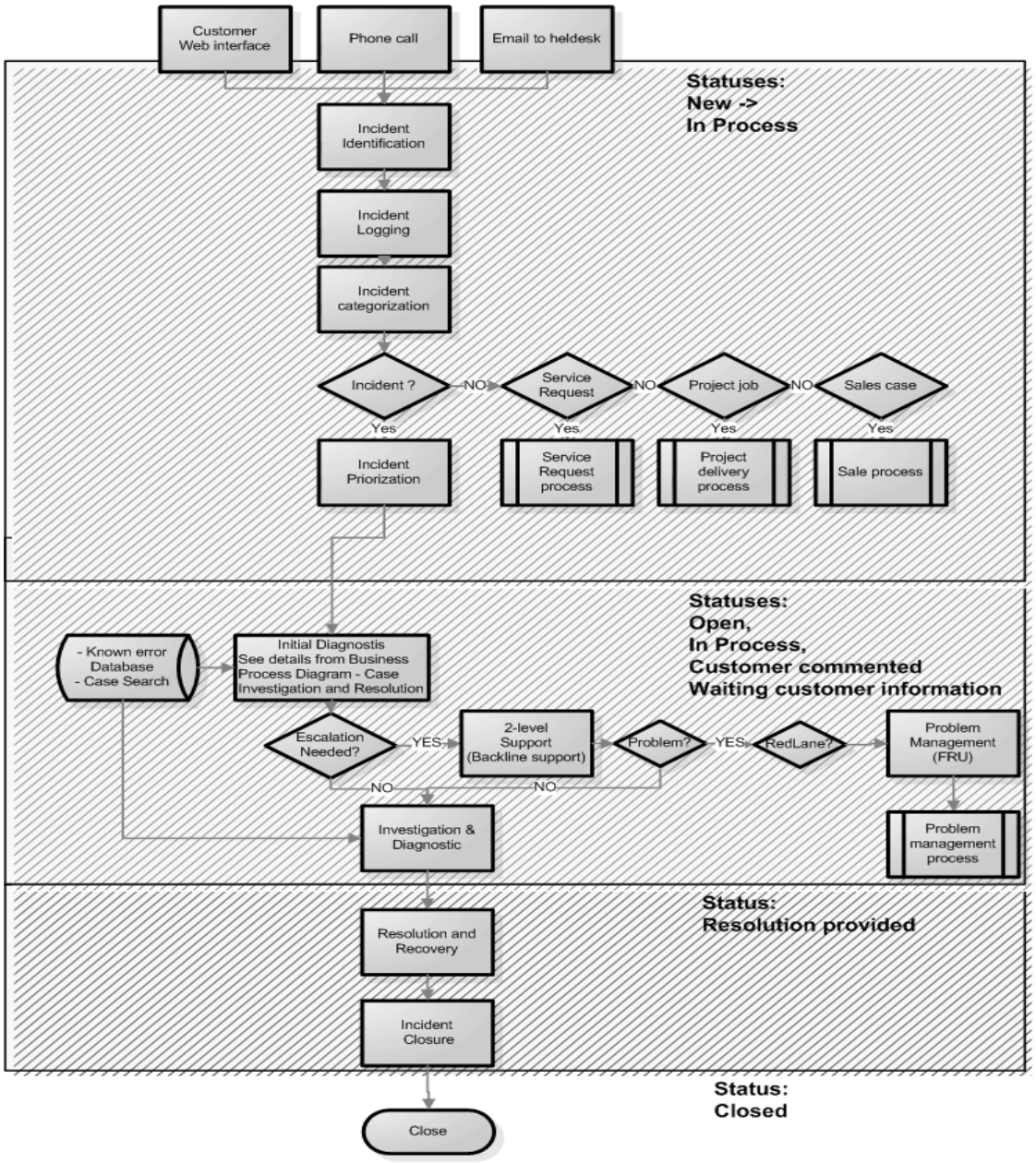
Kaikilla asiakkailla on samaan lähdekoodin perustuva versio tuotteesta, joka on parametrien avulla konfiguroitu eri asiakkaille sopivaksi. Tuotteen lähdekoodi on versionhallinnassa (TortoiseSVN) yhteisomistuksessa eli kaikki kehittäjät tekevät muutoksia ja lisäyksiä samaan lähdekoodiin. Yhteisomistuksen ja jatkuvan integ-

raation (CruiseControl) myötä lisäykset lähdekoodiin tehdään pieninä hallittavina kokonaisuuksina. Tuotteen tuotekehitykseen ja ylläpitoon liittyvät tapahtumat toimivat saman prosessin läpi.

Tuotteen jakelu asiakkaille tapahtuu verkkoporttaalin kautta. Versiot ja sovellukset päivittyvät verkkoporttaaliin asiakkaiden tiedossa olevan kalenterirytmin perusteella: versiot vuositasolla, uusia ominaisuuksia sisältävät buildit kuukausittain ja yksittäiset sovellukset tarvittaessa nopeammin (virheenkorjaus). Järjestelmän koulutuksessa, käyttöönotossa ja asennuksessa asiakkaita auttaa organisaation Projekti-osasto.

5.2.1 Esimerkki palvelupyynnön käsittelystä

Kaikki tuotteeseen liittyvät palvelupyynnöt ja ongelmatilanteet vastaanottaa Support-osasto alla olevan vuokaavion mukaisesti. Tapahtumat kirjataan aina tehtävänhallintajärjestelmään (tilakoodit kerrottu kuvassa).

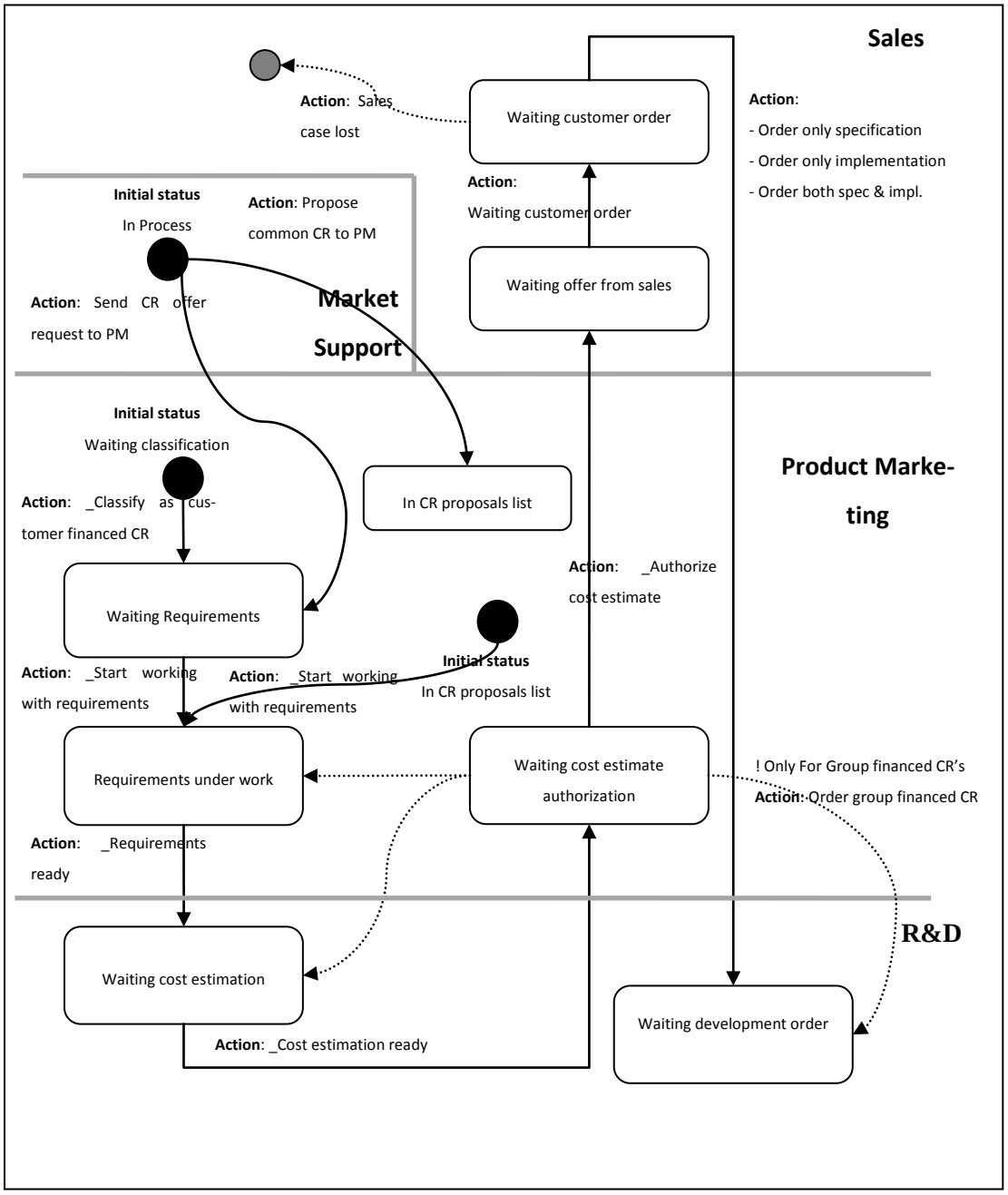


Kuvio 25. Palvelupyynnön tai ongelmatilanteen vastaanotto

Vaativissa ongelmatilanteissa Support-osastoa avustaa erillinen FRU-osasto (fast response unit), jotta asiakkaille luvattu palvelutaso vakavissa tuotannossa ilmenneissä virhetilanteissa saavutetaan.

5.2.2 Esimerkki tuotekehityspyynnön käsittelystä

Jos asiakkaan yhteydenotto on tuottanut tuotekehityspyynnön, käsitellään se tehtävänhallintajärjestelmässä alla olevan vuokaavion mukaisesti. *Action* tarkoitta sovelluksessa suoritettua toimintoa ja laatikossa on kerrottu tehtävän tila.



Kuvio 26. Tuotekehityspyynnön käsittely

5.3 Tuotekehityksen mittarit

Kohdeorganisaatiossa mitattavat asiat perustuvat ylemmän johdon määrittelmiin asioihin, joilla perinteisesti seurataan nykyistä tilannetta annetuun tavoitteeseen. Kohdeorganisaation tuotekehitysosastolla on käytössä seuraavanlaisia tavoitteita ja mitattavia asioita:

- Palvelustasoon eli virheiden korjaus sovitussa ajassa liittyvä mittari
- Ylläpitokustannusten eli ei-laskutettavan työn määrän vähentäminen
- Tehokkuuden parantaminen. Mitataan kuinka hyvin toteutuksen osalta pysytään sovituissa työmääräarvioissa.
- Mitataan kuinka usein toteutettu ominaisuus tulee takaisin testauksesta
- Laskutettavan työn lisääminen
- Henkilökohtaisten tavoitteiden täyttyminen.

Lean-ajatteluun ja asiakaskeskeisyyteen liittyviä ominaisuuksien kokonaisläpimenoaikoja (tilaus ─ toimitus) ei nykyisessä prosessissa mitata. Tiimitasolla ominaisuuksien ja tehtävien toteutuksen työmääräarvioissa pysymistä seurataan päivittäin.

5.4 Kohdeorganisaation lähtökohdat Lean-menetelmille

Vesiputousmallista huolimatta, organisaatiossa on tutkimuksen kannalta monia Lean-ajattelua ja etenkin ketteriä menetelmiä tukevia asioita. Organisaatiossa on tällä hetkellä vakiintunut prosessi ja standardoitu tapa tehdä töitä. Organisaation kaikki tehtävät ja tuotteeseen liittyvät ominaisuudet kirjataan portti-mallin mukaiseen tehtävänhallintajärjestelmään. Organisaation tuottama dokumentaatio on tallennettu SharePoint-sisällönhallintajärjestelmään. On olemassa osittain laaja-alaisia tiimejä (kehittäjiä + testaaja), jotka tuottavat kokonaisia ominaisuuksia tuotteeseen. Osa tuotteista toteutetaan testivetoisen kehityksen perusteella, ja samalla lähdekoodia myös refaktoroidaan. Versionhallinnassa oleva lähdekoodi on yh-

teisomistuksessa eli kaikki kehittäjät tekevät muutoksia samaan lähdekoodiin. Versionhallinnan yhteydessä on integraatiopalvelin, joka rakentaa ja kääntää lähdekoodista sovelluksia muutaman tunnin välein, ja ilmoittaa mahdollisista integraatiovirheistä. Virtuaalipalvelimilla olevat kehitys- ja testausympäristöt vastaavat asiakasympäristöjä. Kaikki tuotteeseen liittyvä informaatio on saatavilla ja ladattavissa yhdestä paikasta (verkkoporttaali). Asiakkaalle näkyvä toiminta on kalenterirytmitettyä ja ennustettavaa. Versiot ja sovellukset päivittyvä verkkoporttaaliin asiakkaiden tiedossa olevan kalenterirytmin perusteella.

Edellä mainitun perusteella kohdeorganisaatiossa on seuraavia Lean-ajattelua tukevia periaatteita: *rakenna laatu ohjelman sisään* (integraatiopalvelin), *luo tietämystä* (sisällönhallintajärjestelmä) ja *Kaizen-prosessin* mahdollistava standardoitu työprosessi. Integraatiopalvelin voidaan nähdä myös Andon-työkaluna (stop-the-line), koska palvelin ilmoittaa rikki menneestä buildista, joka on korjattava välittömästi.

Organisaatio kehittää ja ylläpitää pitkän elinkaaren omaavaa ohjelmistotuoteperhettä. Ohjelmistotuotetteita kehitetään tuotemarkkinointiosaston määrittelemän tiekartan ja versiosuunnitelman pohjalta, jotka ovat tuotteen arvoa lisäävää kehitystyötä (value demand). Vesiputousmallista johtuen kokonaisuudet ovat kohtalaisen suuria, mikä aiheuttaa haasteita dokumenttipohjaisten vaatimusten ja määrityksen tuottamisessa asiakkaan ja tuotemarkkinoinnin välillä. Asiakkaan kannalta vaatimusten hyväksyminen vie prosessia eteenpäin. Jos jotain oleellista jäi vaatimuksista pois, niin edessä on mahdollisesti muutoskäsittely tai lisäominaisuus. Useasti toteutusvaiheessa tarvittavat tarkennukset vievät myös paljon aikaa kehittäjä – tuotemarkkinointi – asiakas akselilla.

Vesiputousmallin mukainen vaihe vaiheelta eteneminen aiheuttaa nykyiseen prosessiin Lean-menetelmien kannalta tietämyshukkaa, koska tieto, vastuu, toiminta ja palaute ovat tietyssä määrin erotettu toisistaan. Tämän myötä ongelmat ilmenevät hyvinkin myöhäisessä vaiheessa. Lisäksi tuotekehityksen resurssien maksimaalinen hyödyntäminen aiheuttaa usein viiveitä näiden myöhäisessä vaiheessa ilmenneiden ja suunnitteluun liittyvien ongelmien selvittelyssä. Ohjelmistotuotteen suuri koko ja vuosien aikana kertynyt tekninen velka, sekä tuotteen monimutkaisuus aiheuttavat myös nykyään haasteita. Nykyään asiakkaiden yhä monipuoli-

semmat vaatimukset, jotka vaikuttavat läpi heidän oman prosessin, luovat haasteita tuotekehitykselle hallita järjestelmää, joka on täynnä erilaisia asiakaskohtaisia ja parametriohjattuja ominaisuuksia.

Ohjelmiston ylläpito, joka on käytännössä virheiden korjaamista, ongelmatilanteiden selvittelyä ja neuvomista (failure demand), tuottaa myös oman osansa tuotteeseen. Asiakkaiden tuotantoympäristöistä saapuvat palvelupyynnöt ja ongelmatilanteet käsitellään organisaation Support-osaston toimesta. Ylläpitoon liittyvät asiat tuodaan tätä kautta normaaliin tuotekehitysprosessiin aiheuttaen muutoksia ja vaihtelevuutta prosessiin. Ylläpidon kautta tulevien uusien kehitettävien ominaisuuksien läpimenoaika voi olla ajallisesti pitkä.

6 TOIMINNALLINEN KEHITTÄMINEN

Tutkimuksen tavoitteena oli tutustua Lean- ja Kanban-menetelmiin ohjelmistotuotannossa, sekä kehittää teoriapohjalta oman organisaation toimintaa näiden menetelmien perusteella. Tutkimuksen aikana tutkittava organisaatio on mielenkiintoisessa tilassa. Organisaation ohjelmistotuotanto perustuu edelleen perinteiseen vesiputousmalliin, mutta emoyhtiön toimesta toimitatapoja haluttaisiin muuttaa enemmän ketterän ohjelmistokehityksen menetelmien mukaiseksi. Emoyhtiön tavoitteena on Scrum-menetelmän käyttöönotto.

6.1 Lean- ja Kanban-menetelmät kohdeorganisaatiossa

Tutkimuksen kannalta organisaation nykyinen tai tuleva ohjelmistokehityksen menetelmä ei ole ongelma. Lean-ajattelu ja siihen perustuvat periaatteet ovat enemmänkin ajattelutapa, jota käytetään koko ohjelmistoprosessin parantamiseen. Monille on pettymys, että Lean-ohjelmistokehitykselle ei ole olemassa yhtä oikeaa lähestymistapaa tai prosessia. On vain olemassa Lean-ajattelu ja siitä johdettuja periaatteita, jotka on ymmärrettävä, kun räätälöidään omaa Lean-ohjelmistoprosessia. Lean-menetelmien ja muiden ketterien menetelmien yhteensopivuutta ei täysin ymmärretä, eikä niiden parasta yhdistelmää vielä ymmärretä riittävän hyvin. (Rodriguez 2013, 49.)

Ketterät menetelmät kuitenkin tukevat valmiiksi monia Lean-ajattelun ja Lean-ohjelmistokehityksen periaatteita. Tutkimuksissa on todettu, että Lean-menetelmillä voidaan parantaa ketteriin menetelmiin perustuvaa ohjelmistokehitystä. Voidaankin nähdä, että Lean parantaa koko ohjelmistoprosessia tunnistamalla ja poistamalla siitä hukkaa ja lisäämällä arvoa. Scrum ketteränä menetelmänä tekee saman tiimin sisäiselle tavalle rakentaa ja ohjata tuotteen kehitystyötä. Lean-menetelmät ja ketterät mentelmät vaativat onnistukseen hyvää johtamista, muuten lopputuloksena on kaaos. Hyvällä johtamisella ei tarkoita perinteistä käskyttämiseen perustuvaa johtamistapaa, vaan enemmänkin valmentavaa, ohjaavaa ja prosessissa ilmenneiden esteiden poistamista (Poppendieck & Poppendieck 2010, 183).

Koska tutkimuksen kohdeorganisaation toiminta perustuu vesiputousmalliin, lähtökohta toiminnan siirtymiselle kohti Lean-ohjelmistokehitystä on eri kuin ketteriä mentelmiä jo käyttävillä organisaatioilla. Tutkimuksen myötä selvisi, että perinteisiä menetelmiä käyttävien organisaatioiden siirtyminen kohti Lean-ajattelua kannattaa aloittaa Kanban-menetelmällä. Kanban-menetelmä voidaan aloittaa visualisoimalla nykyinen prosessi sellaisenaan kuin se on. Mitään erillistä roolitusta ei tarvita, toimintaa ohjaavia sääntöjä ja ohjeistusta on vähän. Visualisoinnin jälkeen rajoitetaan töiden määrää eri vaiheissa. Tämän jälkeen prosessin virtausta tarkkaillaan, mitataan ja kehitetään. Prosessin pullonkaulat ja ongelmat tulevat näkyviksi reaaliaikaisesti, ne ratkaistaan tiimin yhteistyöllä. Kanban-menetelmä tarjoaa vähän vastustusta aiheuttavan lähestymistavan muuttaa nykyistä prosessia kohti Lean-ajattelun mukaista toimintaa. Kanban-mentelmä vaatii Lean-menetelmien tavoin myös hyvää johtamista. (Anderson 2010, 14 - 15: Crisp 2014.)

Kohdeorganisaatiossa Support-osaston kautta nykyiseen prosessiin tulevat palvelupyynnöt ja ongelmatilanteet aiheuttavat vaihtelevuutta, priorisointitarvetta ja yllättäviä ohjelmistomuutoksia tuotteeseen. Tukimuksen perusteella, jatkuvaan virtaukseen perustuva Kanban-menetelmä sopii kohdeorganisaation nykyiseen toimintatapaan paremmin kuin Scrum-menetelmän mukainen kiinteä iteraatio, jossa tuotekehityksen tilanne on rauhoitettu iteraation keston ajaksi.

6.2 Toiminnan kehittäminen kohti Lean-ohjelmistokehitystä

Toiminnallinen kehittäminen on kerrottu seuraavissa luvuissa, joka koetaan tarpeelliseksi kohdeorganisaation kehittämiseksi kohti Lean-ohjelmistokehitystä. Esiteltävät kohdat perustuvat Lean-ajattelun periaatteisiin ja keskittyvät koko ohjelmistoprosessin parantamiseen. Kehittämisen lähtökohtana ovat Lean-ajattelun periaatteet, asiakkaan kokema arvo, prosessin kyky tuottaa arvoa ja tämän mittaaminen, arvovirran kuvaaminen ja prosessin virtauksen hallinta Kanban-menetelmän avulla. Teorian pohjalta luodut kohdat etenevät toteuttamisjärjestyksessä.

6.2.1 Lean-periaatteiden ymmärtäminen

Leanin taustalla on filosofia ja periaatteita, joiden ymmärtäminen on Lean-mentelmien perusta. Nämä periaatteet luovat pohjan koko ajattelutavalle, joka vaatii myös organisaation kulttuurin muuttamista.

Leanin taustalla on Toyotan filosofia, *Toyota Way* ja sen kaksi pilaria: *ihmisten kunnioittaminen* ja *jatkuva parantaminen*. Tämä ajattelutapa ohjaa kuinka olla vuorovaikutuksessa toisten kanssa ja kuinka toimintaa johdetaan. Organisaation toimintaa pyritään jatkuvasti parantamaan kaikkien ihmisten avulla, ei yksistään johtajien toimesta. Ihmisten kunnioittaminen on tavallaan voimanlähde jatkuvaan parantamiseen. Toiminnan taustalla on aina pitkän tähtäimen suunnitelma, jonka pohjalta päätökset tehdään. On olemassa nykytila ja tavoitetila. Tavoitetilaan pyritään pienin askelin tapahtuvan jatkuvan parantamisen (Kaizen) myötä. (Stewart 2012, 29).

Lean-ajattelun taustalla ovat Womackin ja Jonesin viisi toimialasta riippumatonta periaatetta: arvo, arvovirta, virtaus, imu ja täydellisyys. Näiden periaatteiden perusteella Lean-ajattelun ydinajatuksena on maksimoida asiakkaalle luotava arvo ja minimoida lisäarvoa tuottamaton toiminta eli hukka. Lean-organisaatiossa asiakasarvon ymmärtäminen ja sen parantaminen tapahtuu jatkuvalla arvonlisäysprosessin kehittämisellä, läpi koko arvoketun. Toiminnan johtamisen painopiste on siirretty koko arvoketjun optimointiin. Hukan poistaminen arvonlisäysprosessista tarkoittaa sitä, että asiakkaalle tuotettu arvo tapahtuu vähemmillä resursseilla. Prosessista on pyritty poistamaan viiveet ja ylimääräiset työvaiheet. Seitsemän tuotannossa tunnistettua hukan muotoa ovat: varastointi, ylikäsittely, ylituotanto, kuljetus, odottaminen, liike ja virheet. Liker (2006) on lisännyt tähän listaa vielä kahdeksannen hukkatyypin: työntekijän luovuuden käyttämättä jättäminen. (Lean Enterprise Institute 2009: Liker 2006)

Lean-ohjelmistokehityksen periaatteet ovat peräisin teollisen tuotannon Lean-ajattelusta. Taustalla on kokonaisuuden optimointi, hukan poisto ja asiakaskeskeisyys. Mary ja Tom Poppendieck (2007, 23) ovat luoneet seitsemän Lean-ohjelmistokehitykseen soveltuvaa periaatetta:

1. Poista hukka (eliminate waste)
2. Rakenna laatu ohjelman sisään (build quality in)
3. Tietämyksen luominen (create knowledge)
4. Lykkää sitoutumista (defef commitment)
5. Toimita nopeasti (deliver fast)
6. Kunnioita ihmisiä (respect people)
7. Optimoi kokonaisuus (optimize the whole). (Poppendieck & Poppendieck 2007, 23 - 41.)

Poppendieckit ovat myös kuvanneen seitsemän ohjelmistotuotannossa esiintyvää hukan tyyppiä: Osittain tehty työ, ylimääräiset ominaisuudet, uudelleen oppiminen, tuotteen luovutukset, tehtävien vaihdot, viivästykset ja virheet. (Poppendieck & Poppendieck 2007, 73 - 82.)

Huomattavaa on, että Womackin ja Jonesin viisi toimialasta riippumatonta periaatetta (arvo, arvovirta, virtaus, imu ja täydellisyys) käyvät myös hyvin Lean-ohjelmistokehityksen periaatteiksi.

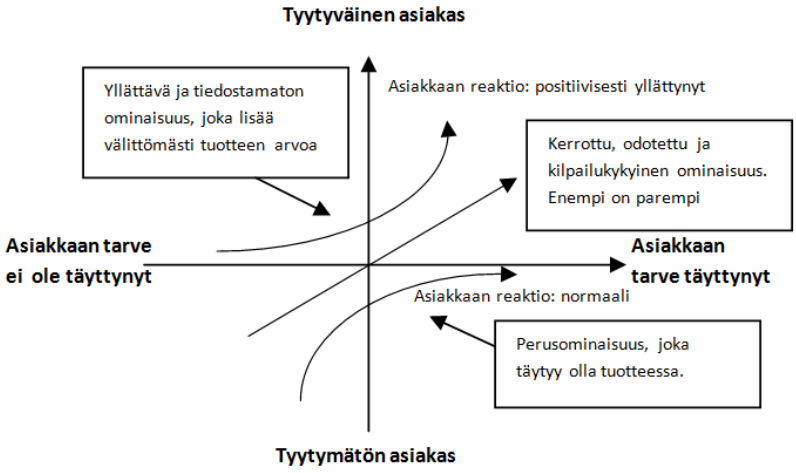
Toiminnan jatkuvan parantamisen mahdollistamiseksi kohdeorganisaatiossa on tällä hetkellä vakiintunut prosessi ja standardoitu tapa tehdä töitä. Prosessi ja työtavat ovat olleet samanlaiset jo melko pitkään. Tutkimuksen perusteella organisaation tulisi ottaa käyttöön ”jatkuvan parantamisen” -palaveri (Kaizen event), jonka voisi nähdä vastaavan Scrum-menetelmästä tuttua retrospektiivipalaveria (jälkipuintipalaveri). Ilman toiminnan jatkuvaan parantamiseen liittyvää ajattelutapaa ja sen opettelua, organisaation siirtyminen Lean-menetelmiin tai yleensä ketteriin menetelmiin on haastavaa.

Kaizen on siis prosessi, joka toistetaan aina kun edellinen muutos on saatu standardisoitua ja vakiinnutettua. Jatkuva parantaminen on asia, joka täytyy sisäistää. Tärkeä asia, mikä yleensä unohtuu ja tavallaan liittyy jatkuvaan parantamiseen, on ihmisten ongelmanratkaisukyvyn jatkuva kehittäminen. (Poppendieck & Poppendieck 2010, 200.)

6.2.2 Asiakkaan kokeman arvon määrittäminen

Oman prosessin tunteminen ja mitä asiakkaat siitä haluavat, on lähtökohta miten prosessia voidaan parantaa. Ohjelmistotuotannossa asiakkaat haluavat heille arvoa tuottavavia ominaisuuksia ja palveluja (value demand). Jos tuote, ominaisuus tai palvelu ei täytä odotuksia, asiakas haluaa tähän korjauksen (failure demand). Kun tunnetaan asiakas ja mitä arvon tuottaminen heille tarkoittaa, on tunnistettava ja ymmärrettävä oman prosessin kyky tuottaa arvoa. Täytyy ymmärtää myös miten nykyinen prosessi arvo tuottaa. Asiakkaan näkemän arvon kannalta ohjelmistotuotanto ei tuota ohjelmistoja, vaan ominaisuuksia joilla ostava asiakas ohjaa omaa tuotantoaan ja tuottaa samalla arvoa omille maksaville asiakkailleen.

Asiakkaiden tarpeiden tunnistamiseen ja toiminnallisten ominaisuuksien arviointiin, sekä näiden suhdetta asiakastyytyväisyyteen ja asiakkaan kokemaan arvoon, voidaan tutkia esimerkiksi Kano-mallin avulla (kuvio 27).



Kuvio 27. Kano-malli

Kano-mallin perusteella asiakkaan tarpeet ja toiminnalliset ominaisuudet jaotellaan kolmeen eri kategoriaan: vaadittu perusominaisuus, odotetusti tarpeet täyttävä ja positiivisesti odottamaton ominaisuus (Cohn 2006, 113). Tämän tyyppistä arviointia liittyen ominaisuuksien asiakkaan kokemaan arvoon, ei kohdeorganisaatiossa ole käytössä.

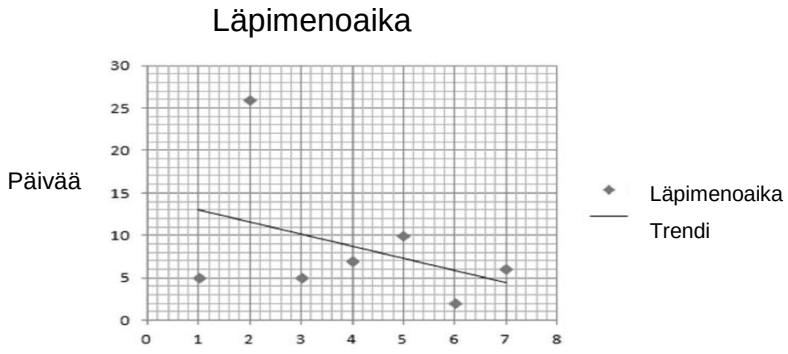
Organisaation ohjelmistotuotteita on kehitetty jo pitkään, ja ne sisältävät paljon erilaisia ominaisuuksia. Ominaisuuksien runsauden vuoksi Kano-mallissa ja sen

analysoinnissa pitäisi keskittyä enemmän asiakkaita positiivisesti yllättäviin ominaisuuksiin. Näidenkin ominaisuuksien arvo asiakkaan kokemana laskee ajan kuluessa.

6.2.3 Prosessin arvontuottokyvyn mittaaminen

Hyvä tapa visualisoida organisaation ohjelmistoprosessin kyky tuottaa arvoa on luoda aikajaksokaavio (kuvio 28). Tarkkailuun valitaan tietyntyyppinen asiakkaalle arvoa tuottava tehtävä tai työ (virheenkorjaus). Kaavioon merkitään kuinka kauan valitun työn tekeminen kesti, kun se saapui järjestelmään ja toimitettiin asiakkaalle. Kun näin on tehty pidemmän aikaa, saadaan selville valitun työtyypin keskimääräinen läpimenoaika. Tämä on hyvä tapa mitata prosessin kyvykkyys, sillä se sisältää kaikki, mitä prosessin sisällä tehtävälle tapahtuu (arvoa lisäävä työ, viiveet, turhat työvaiheet, odottaminen). (Poppendieck & Poppendieck 2010, 13, 198.)

Tämän tyyppinen visuaalinen arvoketjun mittari kohdeorganisaatiosta puuttuu. Lean-menetelmien kannalta tehtävien läpimenaikojen mittaaminen ja parantaminen on oleellinen asia. Lean-menetelmiä hyödyntämällä asiakkaalle näkyvää tehtävien läpimenoaikaa pyritään lyhentämään, jotta asiakas saa sijoituksen nopeasti ja tehokkaasti käyttöön. Läpimenoajan lyhentäminen alentaa oman organisaation tehtävien hallintaan liittyviä kustannuksia. Lyhyempi läpimenoaika nopeuttaa myös palautteen saantia.



Kuvio 28. Työtyypin keskimääräinen läpimenoaika

Oman prosessin suorituskyky on hyvä tunnistaa, sillä ongelmia seuraa, jos tavoitteet asetataan kovemmiksi kuin prosessin kyky tuottaa arvoa. Jos tulos eli tehtävi-

en keskimääräinen läpimenoaika ei ole odotusten mukainen, arvoa tuottavalle prosessille on tehtävä jotain. Muutos voidaan tehdä kolmella eri tavalla: suunnitella tapa tehdä töitä uudestaa, vääristää järjestelmää tai prosessia ja huijata järjestelmää tai prosessia (Poppendieck & Poppendieck 2010, 16).

Kun organisaatio muuttaa ja kehittää prosessia, tämä mittari osoittaa trendiviivalla mihin suuntaa prosessin arvontuottokyky on kokonaisuudessaan menossa. Informatiivisena tietona läpimenoaikojen pituuden mittaamiseen voidaan lisätä myös läpimenoarvojen mediaaniarvo ja läpimenoajoille voidaan määritellä varoitusraja, jonka ylitykseen on puututtava.

Lean-ajattelussa mittareita ei ensisijaisesti käytetä ihmisten valvontaan ja kontrollointiin, vaan mittarit ovat tavoitteita ja työkaluja, joilla tiimit ja työntekijät voivat arvioida omaa suorituskykyään.

6.2.4 Arvovirran kuvaaminen

Prosessin arvovirran kuvaaminen on tärkeä vaihe, sillä se luo pohjan kaikille muille Lean-työkaluille. Prosessin tehostaminen tapahtuu prosessin arvovirran kartoituksella (value stream mapping). Arvovirta kuvataan ja prosessi pilkotaan osiin, materiaalin ja informaation eteneminen selvitetään, hukka pyritään tunnistamaan ja poistamaan arvovirrasta. Lean-ajattelussa arvon määrittämiseksi prosessia tarkastellaan asiakkaan näkökulmasta. Tällä tavoin voidaan erottaa lisäarvoa tuovat vaiheet lisäarvoa tuottamattomista. Prosessivaiheiden arvo määritellään yleensä kolmella tasolla:

1. Lisäarvoa tuottava toiminta
2. Lisäarvoa tuottamaton toiminta
3. Lisäarvoa tuottamaton, mutta välttämätön toiminta.

Lean-ohjelmistokehityksessä arvovirta rakentuu tiedon jalostamisesta sisältäen paljon muuttuvia elementtejä, ja on sen vuoksi haastavampi toteuttaa kuin valmistavan tuotannon arvovirta. Lean-ohjelmistokehityksessä arvovirta rakentuu informaatiosta, erilaisista tehtävistä ja vaiheista joiden kesto, toistettavuus ja resurssi-

tarve vaihtelevat. Prosessia pyritään tehostamaan poistamalla sellainen toiminta (hukka), joka ei asiakkaan näkökulmasta lisää arvoa. Arvovirran kaikki vaiheet pitäisi käsitellä kokonaisuutena, sillä arvovirran yhden vaiheen optimointi aiheuttaa yleensä kustannuksia ja vaikeuksia muille vaiheille.

Poppendieckin (2007, 83) mukaan arvovirtakuvausten pitäisi alkaa asiakkaasta ja loppua asiakkaaseen. Arvovirtakuvauksia voidaan tehdä useita, tyypiltään erilaisille asiakasvaatimuksille. Kuvauksien tekeminen kannattaa aloittaa pienistä ketjuista ja laajentaa ketjujen kokoa, kun ymmärrys oman organisaation prosessista kasvaa. Arvovirtakartan avulla asiat nähdään asiakkaan näkökulmasta ja se toimii työkaluna hukan tunnistamiseen. Arvovirtakarttojen pitäisi olla yksinkertaisia ja aluksi niitä pitäisi tulkita ”alku-loppu”-periaatteella, kuin porautua yhden vaiheen yksityiskohtiin. Valmiin arvovirtakartan perusteella vastataan kysymyksiin:

- Kuinka kauan asiakkaan vaatimuksen tai ominaisuuden valmistaminen kestää?

- Kuinka monta prosenttia käytetystä ajasta oli arvoa tuottavaa aikaa?

Myöhemmin yksityiskohtaisessa tarkastelussa huomio kannattaa kohdistaa viiveisiin ja kahden eri vaiheen väliseen ”pallotteluun”. Vaatimuksiin liittyvä ”pallottelu” viittaa siihen, että vaatimukset ovat menneet hyvinkin yksityiskohtaisiksi liian aikaisin. Testaukseen liittyvä ”pallottelu” ilmaisee sen, että testaus tehdään perinteisesti liian myöhäisessä vaiheessa. Viiveiden syynä on yleensä liian pitkät jonot tehtävälistassa. Viivettä aiheutta myös se, että organisaation prosessissa on samanaikaisesti liian paljon työtä. Tietämyksen siirto (hand-off) organisaatiossa olevien osastojen välillä aiheutta viivettä, jos toinen osapuoli ei ole vielä toimenpiteeseen valmis. Yleensä toinen osapuoli ei ole valmis korkean kapasiteetin eli resurssien maksimaalisen hyödyntämisen vuoksi. (Poppendieck & Poppendieck 2007, 85, 91.)

Arvovirtakartta ylittää organisaatiossa eri osastojen rajat (myös organisaatiorajoja). Organisaatiossa yleensä kukaan ei vastaa arvovirrasta, siten arvovirtaan liittyvät kehitystoimenpiteet jäävät tekemättä. Arvovirralla pirtäisi omistaja, joka raportoi ja hoitaa arvovirtaan liittyvät ongelmakohdat. Ongelmia esiintyy yleensä kahden eri arvovirran rajapinnoissa kun molemmat osapuolet optimoivat omaa toimintaansa

paikallisesti. Täytyy muistaa, että Lean-ajattelussa kokonaisuuden optimointi on tärkeää. (Poppendieck & Poppendieck 2007, 84.)

Kohdeorganisaatiossa arvovirtakuvauksia ei ole olemassa, vaan arvovirta on kuvattu lähinnä vuokaaviona. Vuokaavio kuvaa vain prosessin, eikä esitä mitään ajallista tai resursseihin liittyvää informaatiota. Tutkimuksen perusteella kohdeorganisaatiota suositellaan tekemään osastokohtaisia arvovirtakarttoja prosessissa olevan hukan tunnistamiseksi ja turhien toimintojen poistamiseksi. Osastokohtaiset arvovirtakartat voidaan tietyllä aikavälillä yhdistää yhdeksi kokonaisuudeksi. Arvoa tuottavan ajan osuus eli tehokkuus kokonaisläpimenoajasta saattaa olla yllättävä tieto. Perinteisen vesiputosmallin mukaisessa prosessissa tehokkuus prosenttiluvulla ilmaistuna ei välttämättä ole kovinkaan suuri.

Ward (2007) suosittelee kuvaamaan arvovirtaan tai vuokaavioon myös resurssitarpeen määrän. Visuaalisessa muodossa tämä tieto auttaa prosessissa olevan hukan löytymisessä. (Ward 2007, 40 - 42.)

Prosessin tehokkuutta pyritään parantamaan luomalla tulevaisuuden arvovirtakarttoja. Tulevaisuuden arvovirtakartta on suunnitelma, josta suurimmat viiveet ja hukan aiheuttajat on poistettu. Kyseinen kartta tulee myös olla toteutettavissa kohtuullisen lyhyellä aikavälillä. Muutosten kohteena on vain 1 – 3 kehitettävää kohdetta. (Poppendieck & Poppendieck 2007, 92.)

6.2.5 Kanban-menetelmä ja prosessin virtauksen hallinta

Tutkimusten perusteella Kanban-menetelmä tarjoaa vähän vastustusta aiheuttavan lähestymistavan muuttaa nykyistä prosessia. Menetelmä ei sisällä erillisiä roolituksia ja sen on todettu olevan hyvä tapa tuoda Lean-ajattelua organisaatioon, muokata työkulttuuria ja rohkaista jatkuvan parantamisen ajattelutapaan. Kanbanin on todettu sopivan ketteriä menetelmiä käyttäville tiimeille, mutta myös enemmän perinteistä lähestymistapaa käyttäville tiimeille.

Tutkimuksen Kanban osuudessa läpikäytiin kootusti kaikki menetelmän periaatteet, osittain toteutusjärjestyksen mukaisesti. Kolme ensimmäistä periaatetta ovat

yleisesti käytössä Kanban-menetelmän totetuksessa ja käyttäänotossa ohjelmisto-tuotannossa.

1. Tee työnkulku ja eri vaiheet näkyviksi
2. Rajoita työn määrää eri vaiheissa (WIP)
3. Mittaa ja tarkkaile työn etenemistä.
4. Tee prosessikäytännöistä havainnollisia (eksplisiittisiä)
5. Käytä valmiita malleja tunnistamaan uusia kehitysmahdollisuuksia
6. Prosessin jatkuva kehittäminen (Kaizen). (Anderson 2010, 15, 50.)

Kanban-taulun rakenne vastaa prosessin arvovirtaa. Kuviossa 29 olevat sarakkeet esittävät arvovirran eri työvaiheita. Kanban-menetelmä on suunniteltu toimimaan siten, että se rajoittaa prosessissa olevaa työn määrää (WIP), siksi vaiheen WIP-arvo kerrotaan taululla. Mitä enemmän työtä prosessissa on, sitä hitaampi on töiden virtaus.

Kuvio 29. Kanban-taulu ja arvovirta

Kanban on prosessityökalu, jossa tehtävät virtaavat imuohjatusti eri työvaiheiden läpi siten, että uusi tehtävä haetaan kun edellinen saadaan valmiiksi. Toimintaa ohjataan hallinnoimalla tuotekehityslistan jonoja. Tuotekehityksessä uusi tehtävä työn alle valitaan listasta, joka on priorisoitu tuoteomistajan toimesta. Priorisointivaihe on haastava, sillä siinä on huomioitava asiakkaan näkökulma (arvo) ja myös taloudelliset näkökohdat (cost of delay ja CD3). Kanban tekee työn virtauksen näkyväksi ja paljastaa prosessissa olevia ongelmia pysäyttämällä virtauksen, kun

käynnissä olevien työtehtävien määrä ylittää sallitun rajan. Kun ongelmat ovat esillä, prosessin jatkuva parantaminen on mahdollista (Kaizen). Työn virtaus voidaan jakaa myös useampaan eri kategoriaan, joilla on omat WIP-rajoituksensa. WIP-rajojen avulla pyritään siihen, että töiden virtaus olisi nopeaa ja ruuhkatonta.

Empiirisenä menetelmänä Kanban tukee Lean-periaatteiden mukaisesti prosessin jatkuvaa kehittämistä, siksi WIP-arvoja säätämällä virtaukselle haetaan parasta mahdollista tapaa. Kanban-menetelmän taustalla on paljon toimivaa teoriaa. Jonoteoria, kapeikkoteoria ja Littlen laki ovat lakeja, joita systeemit ja prosessit noudattavat. Näiden teorioiden ymmärtäminen auttaa prosessin virtauksen pullonkaulojen ratkaisemisessa. Teorian perusteella töiden tasaisen virtauksen mahdollistamiseksi ja nopean läpimenoajan ylläpitämiseksi on tuotekehityksessä käytettävä käynnissä olevien töiden rajoittamista (WIP).

Tutkimuksen Lean-osiossa läpikäytiin Reinertsenin kuvaus tuotekehitysjonossa olevien töiden lukumäärän, resurssien maksimaalisen hyödyntämisen ja suurien eräkokojen aiheuttamien ongelmien vaikutukset töiden virtaukseen. Tuotekehityksen rajoittamattomat tehtäväjonot lisäävät vaihtelevuutta yksittäisiin tehtäviin, lisäävät riskiä ja töiden läpimenoaikaa, hidastavat palautteen saamista ja muuttavat jatkuvasti priorisointia. Jonot vähentävät tehokkuutta, laatua ja motivaatiota. Kaikki edellä mainitut ongelmat ovat liiketaloudellista hukkaa.

Suuri osa nykyisistä ohjelmistotuotannon ongelmista aiheutuu juuri prosessin arvovirran hallinnasta, joka on jäänyt vähemmälle huomiolle. Tutkimuksen perusteella ratkaisu näiden ongelmien hallintaan on Kanban-menetelmä.

6.3 Lean kohdeorganisaatiossa

Lean-ajatteluun liitty paljon asioita ja työkaluja, joita ei tässä tutkimuksessa käsitellä. Lean-ajattelu ja siihen perustuvat periaatteet ovat enemmänkin ajattelutapa, jota käytetään koko ohjelmistoprosessin parantamiseen. Lean-ohjelmistokehitykselle ei ole olemassa yhtä oikeaa lähestymistapaa tai prosessia, jonka voisi oppikirjasta kopioida. On vain olemassa Lean-ajattelu ja siitä johdettuja

periaatteita, jotka on ymmärrettävä, kun räätälöidään omaa Lean-ohjelmistoprosessia.

Periaatteet luovat kuitenkin pohjan koko ajattelutavalle, joka vaatii myös organisaation kulttuurin muuttamista. Esimerkiksi Lean-johtamisessa mittareita ei ensisijaisesti käytetä ihmisten valvontaan ja kontrollointiin, vaan mittarit ovat tavoitteita ja työkaluja joilla tiimit ja työntekijät voivat arvioida omaa suorituskykyään.

Tutkimuksen perusteella kohdeorganisaatiolle esitettävät, toiminnalliseen kehittämiseen liittyvät asiat perustuvan kohdeorganisaation nykyiseen toimintatapaan. Tällä hetkellä ohjelmistotuotanto perustuu vesiputousmalliin, mutta muutos kohti ketteriä menetelmiä on aloitettu. Tutkimuksen toiminnalliset muutosehdotukset perustuvat Lean-ohjelmistokehityksen ’*optimoi kokonaisuus’*-periaatteeseen, missä Lean toteutetaan koko arvovirtaan ja samalla pyritään poistamaan hukkaa eli turhaa työtä. Kehitettävät asiat ovat:

- **Jatkuvan parantamiseen ajattelutapa**. Lean-ajattelun myötä kohdeorganisaation tulisi ottaa käyttöön kalenteriaikataulutettu ’jatkuvan parantamisen’ -palaveri, jonka voisi nähdä vastaavan Scrum-menetelmästä tuttua retrospektiivipalaveria (jälkipuintipalaveri). Ilman toiminnan jatkuvaan parantamiseen liittyvää ajattelutapaa, organisaation siirtyminen Lean-menetelmiin tai yleensä ketteriin menetelmiin on haastavaa.

- **Asiakkaan kokeman arvo määrittäminen ja Kano-malli**. Asiakkaiden tarpeiden tunnistamiseen ja toiminnallisten ominaisuuksien arviointiin, sekä näiden suhdetta asiakastyytyväisyyteen ja asiakkaan kokemaan arvoon. Tätä voidaan tutkia ja analysoida Kano-mallin avulla. Tuloksissa kohdeorganisaation kannattaa keskittyä enemmän tuotteen arvoa välittömästi nostaviin ja asiakkaita positiivisesti yllättäviin ominaisuuksiin. Näidenkin ominaisuuksien arvo laskee ajan kuluessa.

- **Visuaalinen prosessin arvontuottokyvyn mittari**. Lean-menetelmien kannalta prosessissa virtaavien tehtävien läpimenoaikojen mittaaminen ja parantaminen on oleellinen asia. Kun organisaatio muuttaa ja kehittää prosessia, tämä mittari osoittaa trendiviivalla, mihin suuntaa prosessin arvon-

tuottokyky on kokonaisuudessaan menossa. Mittarin avulla tiimit ja työntekijät voivat arvioida omaa suorituskykyään.

- **Arvovirran kuvaaminen**. Prosessin arvovirran kuvaaminen on tärkeä vaihe, sillä se luo pohjan kaikille muille Lean-työkaluille. Tutkimuksen perusteella kohdeorganisaatiota suositellaan tekemään arvovirtakarttoja prosessissa olevan hukan tunnistamiseksi ja turhien toimintojen poistamiseksi. Prosessin arvovirralla pirtäisi omistaja, joka raportoi ja hoitaa arvovirtaan liittyvät ongelmakohdat.

- **Kanban-menetelmän käyttöönotto ja prosessin virtauksen hallinta**. Suuri osa nykyisistä ohjelmistotuotannon ongelmista aiheutuu koko prosessin arvovirran hallintatavasta, joka on jäänyt vähemmälle huomiolle. Tutkimuksen perusteella ratkaisu näiden ongelmien hallintaan on Kanban-menetelmä.

7 YHTEENVETO JA POHDINTAA

Tutkimuksen yhteenveto voidaan aloittaa miettimällä vastaus seuraaviin kysymyksiin: mitä, miksi, miten ja kenelle? Lopuksi on kirjoitettu tutkimukseen liittyvää pohdintaa.

Mitä. Tässä työssä tutkittiin Lean-ajattelua, jonka alkuperä on 70 vuotta sitten syntyneessä Toyotan tuotantojärjestelmässä ja Toyotan tavasta valmistaa autoja. Terminä Lean-ajattelu ja sen viisi toimialasta riippumatonta periaatetta tulivat yleiseen tietoisuuteen Womacin ja Jonesin toimesta vasta vuonna 1990. Lean-ajattelun ydinajatuksena on maksimoida asiakkaalle luotava arvo ja minimoida lisäarvoa tuottamaton toiminta eli hukka. Toiminnan parantaminen tapahtuu jatkuvalla arvonlisäysprosessin kehittämisellä, läpi koko arvoketun. Lean-ajattelua hyödynnetään sekä valmistavassa tuotannossa että tuotekehityksessä. Tämän työn osalta Lean-ajattelua tutkittiin erityisesti ohjelmistotuotannon näkulmasta. Työssä tutkittiin myös Kanban-menetelmää, joka Lean-työkaluna liittyy oleellisesti arvovirtojen hallintaan ja ohjaukseen. Lopuksi tutkittiin kohdeorganisaatiota ja miten sen toimintaa voidaan parantaa Lean- ja Kanban-menetelmillä.

Miksi. Ohjelmistotuotanto on nuori tieteenala verrattuna muuhun teolliseen tuotantoon. Erilaisia menetelmiä ja tekniikoita on kehitelty runsaasti, mutta ohjelmistotuotannon käytännöt eivät ole vakiintuneet. Kehitetyt menetelmät ovat yleensä liittyneet uusiin ohjelmointikieliin tai tapoihin määritellä ja kuvata asioita. Ohjelmistotuotannon ongelmat kuitenkin liittyvät yleensä itse tuotantoprosessiin oli siinä käytettävät sisäiset menetelmät kuinka hyviä tahansa.

Perinteisten ohjelmistomenetelmien (esim. vesiputousmalli) lisäksi ohjelmistotuotanto on yli 10 vuotta hakenut uutta suuntaa ketterien ohjelmistomenetelmien muodossa. Ketteryys ei kuitenkaan ole toiminut kokonaisvaltaisen liiketoiminnan ja yrityksen pitkän tähtäimen tavoitteiden kanssa. Ketterät menetelmät eivät myöskään helposti skaalaudu yhden tiimin projektista usean tiimin moniprojektiympäristöksi.

Nykyään ohjelmistotuotannon ongelmiin on haettu apua japanilaisesta autoteollisuudesta peräisin olevalla Lean-ajattelulla. Lean-ajatteluun perustuva tuotan-

nonohjaus valmistavassa tuotannossa ja sen tuotekehityksessä ovat todistaneet toimivuutensa. Ketterien menetelmien toiseen sukupolveen kuuluvaa Lean-ajattelua soveltamalla ohjelmistotuotannosta on saatu järkevämpää, mikä on johtanut laadun ja prosessin paranemiseen. Lean-periaatteet voivat oikein käytettynä auttaa koko ohjelmistotuotannon hallintaan liittyvissä kysymyksissä. Kanban-menetelmä tarjoaa vähän vastustusta aiheuttavan lähestymistavan muuttaa nykyistä prosessia ja se on todettu olevan hyvä tapa tuoda Lean-ajattelua organisaatioon.

Tutkimustyön tavoite oli miettiä teoriapohjalta kohdeorganisaation kehittämistä edellä mainittujen menetelmien perusteella.

Miten. Tämä opinnäytetyö on tyypiltään tapaustutkimus. Työ suoritettiin kvalitatiivisena eli laadullisena tutkimuksena, jotta ymmärrettäisiin tutkimukseen liittyvää viitekehystä: Toyotan valmistamisen filosofia, Lean-ajattelu, Kanban-menetelmä, perinteinen ohjelmistotuotanto ja ketterä ohjelmistotuotanto.

Tapaustutkimuksen aineisto hankinna lähtökohtana oli tutkimustehtävä. Tutkimuksen primaariaineistona on käytetty välitöntä tietoa tutkimuskohteena olevista Lean- ja Kanban-menetelmistä. Tutkimuksen teoriaosuus aloitettiin tutustumalla Mary ja Tom Poppendieckin (2003 – 2013) kirjoittamaan Lean-ohjelmistotuotannon kirjasarjaan, missä teollisen tuotannon Lean-ajattelun periaatteet on muunnettu ohjelmistotuotantoon. Tutkimustyö eteni Toyotan Lean-periaatteisiin tutustumiseen John Stewartin (2012), Jeffrey K. Likerin (2006) ja Likerin ja Convisin (2012) kirjojen kautta. Lean-tuotekehityksen taustalla oleviin periaatteisiin ja teoriaan käytettiin Donald G. Reinertsenin (2009) ja Allen C. Wardin (2007) teoksia, jotka antoivat hyvät pohjatiedot myös Kanban-menetelmälle. Kanban osuuden primaariaineistona käytettiin David J. Andersonin (2010) kirjaa.

Tutkimuksen sekundaariaineistona käytettiin muiden tekemiä empiirisiä tutkimuksia. Tutkimusaineistoa löytyi väitöskirjojen muodossa sekä Lean-menetelmästä että Kanban-menetelmästä. Sekundaariaineisto laajensi tutkimusta kohti Lean-menetelmien ja ketterien menetelmien yhdistämisestä.

Kenelle. Tutkimustyö on tehty kohdeorganisaatiolle, jonka ohjelmistotuotannon toimintatapa perustuu perinteiseen vesiputousmalliin. Toimintaa ollaan kuitenkin

muuttamassa kohti ketteriä menetelmiä. Esitetyt toiminnalliset kehitysehdotukset perustuvat lean-ohjelmistokehityksen ”*optimoi kokonaisuus”*-periaatteeseen. Kehityskohteet ovat: Lean-ajattelun periaatteet, asiakkaan kokeman arvon määrittäminen, prosessin kyky tuottaa arvoa ja tämän mittaaminen, arvovirran kuvaaminen ja prosessin virtauksen hallinta Kanban-menetelmän avulla

Pohdintaa. Tutkimustyön aihealue oli haastavampi kuin aluksi osasi odottaa. Osittain siksi, että aihealue sisälsi tutkimuksen tekijälle paljon uutta asiaa. Mitä pidemmälle työn tekeminen eteni, sitä enemmän löytyi uutta aineistoa ja tutkimuksia liittyen tutkimustyöhön. Tämä tietysti aiheutti muutoksia ja uusia näkökulmia tutkimuksen tekemiseen.

Ohjelmistotuotannon Lean-ajattelu ja siihen perustuvat periaatteet ovat enemmänkin ajattelutapa, jota käytetään koko ohjelmistoprosessin parantamiseen. Lean-ohjelmistokehitykselle ei ole olemassa yhtä oikeaa lähestymistapaa tai prosessia. On vain olemassa Lean-ajattelu ja siitä johdettuja periaatteita, jotka on ymmärrettävä, kun räätälöidään omaa Lean-ohjelmistoprosessia.

Valmistavassa tuotannossa hukan lähde ja sen esiintyminen voidaan havaita tarkkailemalla fyysisen materiaalin virtausta, sekä koneen tai työntekijän toimintaa. Ohjelmistotuotannon työtehtävät ja työvaiheet ovat enemmänkin aineettomia ja siten vaikeuttaa hukan tunnistamista ja prosessin parantamista. Työt ovat informaatiota, eikä asiaa ajatella aineellisena varastona, kuten valmistavassa tuotannossa. Kuitenkin tämä mittaamaton jono aiheuttaa suuren osan tuotekehityksen ongelmista ja vaikuttaa ennen kaikkea heikentävästi taloudelliseen suorituskykyyn.

Lean-työkalujen, kuten Kanbanin-menetelmän käyttö kuitenkin visualisoi prosessin, ohjaa töiden virtausta prosessissa ja tuoden samalla ongelmat (hukan) reaaliaikaisesti esille. Työkalujen tarkoitus onkin tuoda ongelmakohtia näkyvästi esille, joita työntekijät ja esimiehet yhdessä ratkaisevat. Muutosvaiheen alussa tapahtuva toiminnan pysäyttäminen ongelmanratkaisun ajaksi voidaan kokea hyvinkin turhauttavaksi. Lean-ajattelu kuitenkin helpottaa ongelmien juurisyiden löytämistä.

Lean-ajatteluun käyttöönottoon liittyvä ongelma on hyvinkin samanlainen, kuin ketterissä menetelmissä nyt meneillään oleva ongelma eli skaalautuvuus. Ajatteluta-pa ja menetelmät eivät ole mukana kokonaisvaltaisen liiketoiminnan ja yrityksen

pitkän tähtäimen tavoitteiden kanssa. Myynti, tuotehallinto tai asiakas eivät toiminnassaan käytä Lean-ajattelua tai ketteriä menetelmiä. Kohdeorganisaatiossa tuotekehitysosaston on vaikea olla yksin virtaviivainen tai ketterä ja asiakaskin pitää ensin opettaa virtaviivaisen ketteräksi. Todellista tietoa siitä, mitä Lean-käytäntöjä ja työkaluja on käytännössä sovellettu, on edelleen aika niukasti olemassa.

Kohdeorganisaation visiossa kerrottiin seuraavaa: *Tarkoituksena on auttaa asiakkaita vähentämään manuaalisia prosesseja, virtaviivaistamaan heidän liiketoimintaa ja saavuttamaan mittavia tuloksia liiketoiminnan kaikilla osaalueilla*. Jos kohdeorganisaatiossa omaksutaan Lean-ajattelu tai oikeastaan Lean- ja Agile-ajattelu kokonaisuudessaan, myös organisaation oma ohjelmistoprosessi virtaviivaistuu ja alkaa olla vision mukainen. Tutkimusten mukaan, Lean-menetelmillä ohjelmistokehitys tulee saavuttamaan suuria ja ainutlaatuisia tuloksia. Ja kaikki tapahtuu vain uuden ajattelutavan johtamisella.

Lean-johtaminen eli Lean-Management voisi olla luonnollinen jatkotutkimuksen aihe tälle työlle. Lean-johtamisella kaikki Lean-periaatteet saadaan hallitusti käyttöön.

LÄHTEET

Aalto-Yliopisto. 2012. Ketteristä menetelmistä tukea yritysten projektinhallintaan [Verkkojulkaisu]. [Viitattu 23.4.2014]. Saatavana: http://web.aalto.fi/fi/current/news/2012-02-27/.

Agile Manifesto Org. 2001. Manifesto for Agile Software Development. [Verkkojulkaisu]. [Viitattu 30.12.2012]. Saatavana: http://agilemanifesto.org/iso/fi.

Anderson ,D. J. 2010. Kanban. Successful Evolutionary Change for Your Technology Business. Sequim, Yhdysvallat, Blue Hole Press.

Black Swan Farming. 2013. Black Swan Farming using Cost of Delay.[Verkkojulkaisu]. [Viitattu 05.03.2014]. Saatavana: https://docs.google.com/document/d/1COg4UbmPWJTnQop1BJ8CLTiBlYKbMZAgErH8bIWEd8/pub

Boeg, J. 2011. Priming Kanban. [Verkkojulkaisu]. [Viitattu 05.03.2014]. Saatavana: http://www.infoq.com/minibooks/priming-kanban-jesper-boeg

Cohn, M. 2006. Agile Estimating and Planning. Westford, Massachusetts, Yhdysvallat, Pearson Education.

Cloud Software Finland. 2012. Lean In Cloud. Lean Thinking Principles for Cloud Software. [Verkkojulkaisu]. [Viitattu 03.03.2014]. Saatavana: https://www.cloudsoftwareprogram.org/results/deliverables-and-other-reports/i/27000/1941/d2-1-1-lean-in-cloud-lean-thinking-principles-for-cloud-software.

Crisp. 2014. Kanban. [Verkkojulkaisu]. [Viitattu 18.04.2014 Saatavana: http://www.crisp.se/gratis-material-och-guider/kanban.

Denning, S. 2010. The Leader’s Guide to Radical Management : reinventing the workplace for the 21st century. San Francisco, Yhdysvallat, Jossey-Bass.

Helsingin Sanomat. 2008. Toyotan tulos painumassa tappiolle ensimmäisen kerran 70 vuoteen. [Verkkojulkaisu]. [Viitattu 27.01.2014]. Saatavana: http://www.hs.fi/autot/a1371449934985

Helsingin Yliopisto. 2014. Ohjelmistoprosessit ja ohjelmistojen laatu. [Verkkojulkaisu]. [Viitattu 24.04.2014]. Saatavana: http://www.cs.helsinki.fi/u/paakki/Laatu-14-Luentokalvot-1.pdf

Huuhka, M. 2010. Luovan asiantuntijaorganisaation johtamien. Helsinki, Talentum Media Oy.

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

Haikala, I. & Märijärvi, J. 2004. Ohjelmistotuotanto. Helsinki, Talentum Media Oy.

Ikonen, M. 2011. Lean Thinking in Software Development - impacts of Kanban on Projects. [Verkkojulkaisu]. [Viitattu 12.03.2014]. Saatavana: https://helda.helsinki.fi/handle/10138/28453.

Kniberg, H. 2011. Lean from the Trenches – An example of Kanban in a large software project. [Verkkojulkaisu]. [Viitattu 12.04.2014] Saatavana: http://www.crisp.se/file-uploads/Lean-from-the-trenches.pdf.

Kniberg, H. & Skarin, M. 2010. Kanban and Scrum – making the most of both. C4Media Inc, Yhdysvallat, Publisher of InfoQ.com.

Larman, C. & Vodde, B. 2009. Lean Primer. Versio 1.5. [Verkkojulkaisu]. [Viitattu 30.12.2012]. Saatavana:http://www.leanprimer.com.

LeanAgileChange. 2014. Cost of Delay. [Verkkojulkaisu]. [Viitattu 25.04.2014]. Saatavana: http://leanagilechange.com/leanagilewiki/index.php?title=Cost_of_Delay.

Lean Enterprise Institute. 2009. What is Lean? .[Verkkojulkaisu]. [Viitattu 30.12.2012]. Saatavana: http://www.lean.org/WhatsLean.

Liker, J. K. 2006. *Toyotan tapaan*. Suomentaja Marko Niemi. Helsinki, Readme.fi.

Liker, J. K. & Convis, G. L. 2012. *Toyotan tapa lean johtamiseen*. Suomentaja Marko Niemi. Helsinki, Readme.fi.

Neilimo, K. & Uusi-Rauva, E. 2010. Johdon laskentatoimi. Helsinki, Edita Prima Oy.

Net Objectives. 2012a. Mindsets: Waterfall,1st & 2nd Generation Agile. [Verkkojulkaisu]. [Viitattu 27.01.2014]. Saatavana: http://www.netobjectives.com/blogs/mindsets-waterfall-1st-2nd-generation-agile.

Net Objectives. 2012b. Why Agile May Not Work for You, But Lean-Agile Will [Verkkojulkaisu]. [Viitattu 27.01.2014]. Saatavana: http://www.netobjectives.com/blogs/why-agile-may-not-work-you-lean-agile-will.

Oivo, M. 2014. Lean & Agile. [Verkkojulkaisu]. [Viitattu 15.4.2014]. Saatavana: https://tapahtumat.tekes.fi/uploads/a01a0390/Markku_Oivo-9436.pdf.

Pichler, R. 2010. Agile Product Management with Scrum: Creating Products That Customers Love. Westford, Massachusetts, Yhdysvallat, Addison-Wesley

Poppendieck, M. & Poppendieck, T. 2003. Lean Software Development. 16. uud. p. Crawfordsville, Indiana, Yhdysvallat, Addison-Wesley.

Poppendieck, M. & Poppendieck, T. 2007. Implementing Lean Software Development. 8. uud. p. Stoughton, Massachusetts, Yhdysvallat, Addison-Wesley.

Poppendieck, M. & Poppendieck, T. 2010. Leading Lean Software Development. 3. uud. p. Crawfordsville, Indiana, Yhdysvallat, Addison-Wesley.

Poppendieck, M. & Poppendieck, T. 2013. Lean is mindset – Ask the right questions. Westford, Massachusetts, Yhdysvallat, Addison-Wesley

Reinertsen, D. G. 2009. The Principles of Product Development Flow: Second generation Lead Product Development. Yhdysvallat, Celeritas Publishing.

Rodriguez, P. 2013. Combining Lean thinking and Agile Software Development. How do software-intensive companies use them in practice?. [Verkkojulkaisu]. [Viitattu 30.01.2014]. Saatavana: http: //www.hallinto.oulu.fi/viestin/ vaitos13/ rodriguez.html.

Stewart, J. 2012. The Toyota Kaizen Continuum: A Practical Guide to Implementing Lean. Yhdysvallat, CRC Press, Taylor & Francis Group.

Toyota Motor Manufacturing Kentucky. 2012. Toyota. [Verkkosivu]. [Viitattu 30.12.2012]. Saatavana: http://www.toyotageorgetown.com/history.asp.

VersionOne. 2013. 7th Annual State of Agile. [Verkkojulkaisu]. [Viitattu 14.12.2013]. Saatavana: http://www.versionone.com/pdf/7th-Annual-State-of-Agile-Development-Survey.pdf.

VersionOne. 2014a. 8th Annual State of Agile. [Verkkojulkaisu]. [Viitattu 14.12.2013]. Saatavana: http://stateofagile.versionone.com/thankyou-for-requesting-the-state-of-agile.

VersionOne. 2014b. Kanban for Agile teams.[Verkkojulkaisu]. [Viitattu 6.4.2014]. Saatavana: http://pm.versionone.com/whitepaper_kanbanagileteams.html.

VTT. 2008. Lean tuotekehitys. [Verkkojulkaisu]. [Viitattu 14.12.2012]. Saatavana: http://www.vtt.fi/proj/leanver/files/lean_tuotekehitys.pdf.

Ward, A. C. 2007. Lean Product and Process Development.Yhdysvallat, The Lean Enterprise Institute Inc.